MULTIMEDIA AND ANIMATION

BRANCH: CS/IT

YEAR:FIRST

SEM:SECOND

MULTIMEDIA AND ANIMATION

BY NAVNEET SOLANKI (LECTURER -IT)AND PRASHANT KUMAR(LECTURER--CS)

**UNIT-3 COMPRESSION /DECOMPRESSION AND FILE FORMATS**

**Need of compression**

# Compression reduces the size of an application or document for storage or transmission. Compressed files are smaller, download faster, and easier to transport. Decompression or expansion restores the document or application to its original size.

# Why Compression

Text data which is ASCII Files is transferred at 100kbps( 50 pages of text) and most LANS easily handle this. However for images, video, audio which are large data objects requiring large storage space. In order to manage large multimedia data objects need to be compressed to reduce file size and at the receiver end it needs to be reconstructed, decompressed . These Multimedia elements need to be stored, retrieved, transmitted , displayed and hence compression techniques are required. The compression and decompression techniques are being used for variety of applications like facsimile, printer, document storage and retrieval, teleconferencing, multimedia messaging systems. Compression principles try to eliminate redundancies (Example : In B/W picture we can use a code mechanism to store repeated white pixels and in absence it will be treated as black. Compression techniques attempt to reduce horizontal and vertical redundancies in CCITT standards Huge Data Demands: The way the humans are communicating is increasing huge demand of bandwidth. As the technology is developing people seek screens which are larger with more resolution, there is need of high density TV. On a LAN , WAN, MAN, Internet depending on the bandwidth available appropriate techniques are required for reliable communications. Hence there is need for techniques of compression and decompression. Hence CCITT ( Consultative Committee for Telephone and Telegraph) has standardized compression and decompression techniques.

**Compression**is useful because it helps reduce the consumption of expensive resources, such as hard disk space or transmission bandwidth. On the downside, compressed data must be decompressed to be used, and this extra processing may be detrimental to some applications. For instance, a compression scheme for video may require expensive hardware for the video to be decompressed fast enough to be viewed as it is being decompressed (the option of decompressing the video in full before watching it may be inconvenient, and requires storage space for the decompressed video). The design of data compression schemes therefore involves trade-offs among various factors, including the degree of compression, the amount of distortion introduced (if using a lossy compression scheme), and the computational resources required to compress and uncompress the data. Compression was one of the main drivers for the growth of information during the past two decades.

**TYPES OF COMPRESSION**

# Lossless versus lossy compression

Lossless compression algorithms usually exploit statistical redundancy in such a way as to represent the sender's data more concisely without error. Lossless compression is possible because most real-world data has statistical redundancy. For example, in English text, the letter 'e' is much more common than the letter 'z', and the probability that the letter 'q' will be followed by the letter 'z' is very small. Another kind of compression, called lossy data compression or perceptual coding, is possible if some loss of fidelity is acceptable. Generally, a lossy data compression will be guided by research on how people perceive the data in question. For example, the human eye is more sensitive to subtle variations in luminance than it is to variations in color. JPEG image compression works in part by "rounding off" some of this less-important information. Lossy data compression provides a way to obtain the best fidelity for a given amount of compression.

## Lossy

Lossy image compression is used in digital cameras, to increase storage capacities with minimal degradation of picture quality. Similarly, DVDs use the lossy MPEG-2 Video codec for video compression.

In lossy audio compression, methods of psychoacoustics are used to remove non-audible (or less audible) components of the signal. Compression of human speech is often performed with even more specialized techniques, so that "speech compression" or "voice coding" is sometimes distinguished as a separate discipline from "audio compression". Different audio and speech compression standards are listed under audio codecs. Voice compression is used in Internet telephony for example, while audio compression is used for CD ripping and is decoded by audio players.

## Lossless

The Lempel–Ziv (LZ) compression methods are among the most popular algorithms for lossless storage. DEFLATE is a variation on LZ which is optimized for decompression speed and compression ratio, but compression can be slow. DEFLATE is used in PKZIP, gzip and PNG. LZW (Lempel–Ziv–Welch) is used in GIF images. Also noteworthy are the LZR (LZ–Renau) methods, which serve as the basis of the Zip method. LZ methods utilize a table-based compression model where table entries are substituted for repeated strings of data. For most LZ methods, this table is generated dynamically from earlier data in the input. The table itself is often Huffman encoded (e.g. SHRI, LZX). A current LZ-based coding scheme that performs well is LZX, used in Microsoft's CAB format.

The very best modern lossless compressors use probabilistic models, such as prediction by partial matching. The Burrows–Wheeler transform can also be viewed as an indirect form of statistical modelling.

In a further refinement of these techniques, statistical predictions can be coupled to an algorithm called arithmetic coding. Arithmetic coding, invented by Jorma Rissanen, and turned into a practical method by Witten, Neal, and Cleary, achieves superior compression to the better-known Huffman algorithm, and lends itself especially well to adaptive data compression tasks where the predictions are strongly context-dependent. Arithmetic coding is used in the bilevel image-compression standard JBIG, and the document-compression standard DjVu. The text entry system, Dasher, is an inverse-arithmetic-coder.

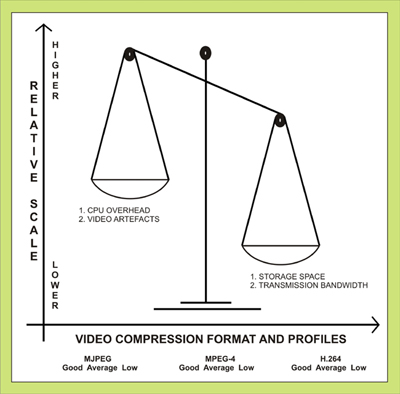
**VIDEO COMPRESSION TECHNIQUES**

Video Compression Standards – Pros & Cons

Video Compression is the term used to define a method for reducing the data used to encode digital video content. This reduction in data translates to benefits such as smaller storage requirements and lower transmission bandwidth requirements, for a clip of video content.

Video compression typically involves an elision of information not considered critical to the viewing of the video content, and an effective video compression codec (format) is one that delivers the benefits mentioned above: without a significant degradation in the visual experience of the video content, post-compression, and without requiring significant hardware overhead to achieve the compression. Even within a particular video compression codec, there are various levels of compression that can be applied (so called profiles); and the more aggressive the compression, the greater the savings in storage space and transmission bandwidth, but the lower the quality of the compressed video [as manifested in visual artefacts – blockiness, pixelated edges, blurring, rings – that appear in the video] and the greater the computing power required.

**Motion JPEG (M-JPEG):** Originally developed by JPEG (Joint Photographic Experts Group) for multimedia PC applications, M-JPEG is now used by many digital video capture devices. In M-JPEG, each video frame is separately compressed as a JPEG image.**MPEG-4 Part 2:** Developed by MPEG (Motion Pictures Expert Group), it is a successor MPEG standards such as MPEG-1 and MPEG-2, and offers higher compression factors.**H.264 (MPEG-4 Part 10 or MPEG-4 AVC):** This is a standard jointly developed and maintained by ITU-T (Telecommunication Standardisation Sector) and MPEG, and hence the fact that it is referred by multiple acronyms.



There are two approaches to achieving video compression, viz. intra-frame and inter-frame. Intra-frame compression uses the current video frame for compression: essentially image compression. Inter-frame compression uses one or more preceding and/or succeeding frames in a sequence, to compress the contents of the current frame. An example of intra-frame compression is the Motion JPEG (M-JPEG) standard. The MPEG-1 (CD, VCD), MPEG-2 (DVD), MPEG-4, and H.264 standards are examples of inter-frame compression.The popular video compression standards in the IP video surveillance market are M-JPEG, MPEG-4, and H.264.

**Comparison of Video Compression Standards**

There is a constant improvement in video compression factors, thanks to new techniques and technology, and some new formats in the horizon are H.265 and VP8.

* H.265 is still in the process of being formulated, and aims to achieve a 25% improvement in the compression factor while lowering computational overhead by 50%: for the same perceived video quality.
* VP8 is a codec from On2 Technologies (which recently agreed to be acquired by Google), who claims that the codec brings bandwidth savings and uses less data than H.264: to the extent of 40%. There is currently a fight over the standard to be chosen for Web video (fuelled by the upcoming HTML5 standard), and VP8 is slugging it out with H.264 and Ogg Theora.

**MPEG**

Short for ***M****oving****P****icture****E****xperts****G****roup*, and pronounced m-peg, is a working group of the [ISO](https://www.webopedia.com/TERM/I/ISO.html). The term also refers to the family of [digital video](https://www.webopedia.com/TERM/D/digital_video.html) compression standards and file formats developed by the group. MPEG generally produces better-quality video than competing formats, such as [Video for Windows](https://www.webopedia.com/TERM/V/Video_for_Windows.html), [Indeo](https://www.webopedia.com/TERM/I/Indeo.html) and [QuickTime](https://www.webopedia.com/TERM/Q/QuickTime.html). MPEG files previously on PCs needed [hardware](https://www.webopedia.com/TERM/H/hardware.html) [decoders](https://www.webopedia.com/TERM/D/decoder.html) (codecs) for MPEG processing. Today, however, PCs can use [software](https://www.webopedia.com/TERM/S/software.html)-only [codecs](https://www.webopedia.com/TERM/C/codec.html)including products from RealNetworks, QuickTime or Windows Media Player.

MPEG algorithms compress data to form small bits that can be easily transmitted and then decompressed. MPEG achieves its high compression rate by storing only the changes from one [frame](https://www.webopedia.com/TERM/F/frame.html) to another, instead of each entire frame. The [video](https://www.webopedia.com/TERM/V/video.html) information is then [encoded](https://www.webopedia.com/TERM/E/encoding.html) using a technique called ***D****iscrete****C****osine****T****ransform* ([DCT](https://www.webopedia.com/TERM/D/DCT.html)). MPEG uses a type of [lossy compression](https://www.webopedia.com/TERM/L/lossy_compression.html), since some data is removed. But the diminishment of data is generally imperceptible to the human eye.  
  
The major MPEG standards include the following;

* **MPEG-1**: The most common implementations of the MPEG-1 standard provide a video resolution of 352-by-240 at 30 frames per second (fps). This produces video quality slightly below the quality of conventional VCR videos.
* **MPEG-2**: Offers resolutions of 720x480 and 1280x720 at 60 fps, with full CD-quality audio. This is sufficient for all the major TV standards, including NTSC, and even HDTV. MPEG-2 is used by DVD-ROMs. MPEG-2 can compress a 2 hour video into a few gigabytes. While decompressing an MPEG-2 data stream requires only modest computing power, encoding video in MPEG-2 format requires significantly more processing power.
* **MPEG-3**: Was designed for HDTV but was abandoned in place of using MPEG-2 for HDTV.
* **MPEG-4**: A graphics and video compression algorithm standard that is based on MPEG-1 and MPEG-2 and Apple QuickTime technology. Wavelet-based MPEG-4 files are smaller than JPEG or QuickTime files, so they are designed to transmit video and images over a narrower bandwidth and can mix video with text, graphics and 2-D and 3-D animation layers. MPEG-4 was standardized in October 1998 in the ISO/IEC document 14496. *See*[*MPEG-4*](https://www.webopedia.com/TERM/M/MPEG_4.html).
* **MPEG-7:** Formally called the *Multimedia Content Description Interface*, MPEG-7 provides a tool set for completely describing multimedia content. MPEG-7 is designed to be generic and not targeted to a specific application.
* **MPEG-21:** Includes a *Rights Expression Language* ([REL](https://www.webopedia.com/TERM/R/REL.html)) and a Rights Data Dictionary. Unlike other MPEG standards that describe compression coding methods, MPEG-21 describes a standard that defines the description of content and also processes for accessing, searching, storing and protecting the copyrights of content. *See*[*MPEG-21*](https://www.webopedia.com/TERM/M/MPEG_21.html).

**H.261**

**H.261** is an [ITU-T](https://en.wikipedia.org/wiki/ITU-T) [video compression standard](https://en.wikipedia.org/wiki/Video_compression_standard), first ratified in November 1988.[[1]](https://en.wikipedia.org/wiki/H.261#cite_note-1)[[2]](https://en.wikipedia.org/wiki/H.261#cite_note-H2611988-2) It is the first member of the H.26x family of video coding standards in the domain of the ITU-T [Video Coding Experts Group](https://en.wikipedia.org/wiki/Video_Coding_Experts_Group) ([VCEG](https://en.wikipedia.org/wiki/VCEG), then Specialists Group on Coding for Visual Telephony), and was developed with a number of companies, including [Hitachi](https://en.wikipedia.org/wiki/Hitachi), [PictureTel](https://en.wikipedia.org/wiki/PictureTel), [NTT](https://en.wikipedia.org/wiki/Nippon_Telegraph_and_Telephone), [BT](https://en.wikipedia.org/wiki/BT_plc) and [Toshiba](https://en.wikipedia.org/wiki/Toshiba). It was the first video coding standard that was useful in practical terms.

H.261 was originally designed for transmission over [ISDN](https://en.wikipedia.org/wiki/Integrated_Services_Digital_Network) lines on which data rates are multiples of 64 kbit/s. The coding algorithm was designed to be able to operate at video bit rates between 40 kbit/s and 2 Mbit/s. The standard supports two video frame sizes: [CIF](https://en.wikipedia.org/wiki/Common_Intermediate_Format) (352×288 luma with 176×144 chroma) and QCIF (176×144 with 88×72 chroma) using a [4:2:0](https://en.wikipedia.org/wiki/4:2:0) sampling scheme. It also has a backward-compatible trick for sending still images with 704×576 luma resolution and 352×288 chroma resolution (which was added in a later revision in 1993).

The [discrete cosine transform](https://en.wikipedia.org/wiki/Discrete_cosine_transform) (DCT), a form of [lossy compression](https://en.wikipedia.org/wiki/Lossy_compression), was first proposed by [Nasir Ahmed](https://en.wikipedia.org/wiki/N._Ahmed) in 1972.[[3]](https://en.wikipedia.org/wiki/H.261#cite_note-Ahmed-3) Ahmed developed a working algorithm with T. Natarajan and [K. R. Rao](https://en.wikipedia.org/wiki/K._R._Rao) in 1973,[[3]](https://en.wikipedia.org/wiki/H.261#cite_note-Ahmed-3) and published it in 1974.[[4]](https://en.wikipedia.org/wiki/H.261#cite_note-pubDCT-4)[[5]](https://en.wikipedia.org/wiki/H.261#cite_note-pubRaoYip-5) DCT would later become the basis for H.261.[[6]](https://en.wikipedia.org/wiki/H.261#cite_note-Ghanbari-6)

The first digital [video coding standard](https://en.wikipedia.org/wiki/Video_coding_standard) was [H.120](https://en.wikipedia.org/wiki/H.120), created by the [CCITT](https://en.wikipedia.org/wiki/ITU-T) (now ITU-T) in 1984.[[7]](https://en.wikipedia.org/wiki/H.261#cite_note-history-7) H.120 was not usable in practice, as its performance was too poor.[[7]](https://en.wikipedia.org/wiki/H.261#cite_note-history-7) H.120 was based on [differential pulse-code modulation](https://en.wikipedia.org/wiki/Differential_pulse-code_modulation) (DPCM), which had inefficient compression. During the late 1980s, a number of companies began experimenting with the much more efficient DCT compression for video coding, with the CCITT receiving 14 proposals for DCT-based video compression formats, in contrast to a single proposal based on [vector quantization](https://en.wikipedia.org/wiki/Vector_quantization) (VQ) compression. The H.261 standard was subsequently developed based on DCT compression.[[6]](https://en.wikipedia.org/wiki/H.261#cite_note-Ghanbari-6)

H.261 was developed by the [CCITT](https://en.wikipedia.org/wiki/CCITT) Study Group XV Specialists Group on Coding for Visual Telephony (which later became part of ITU-T SG16), chaired by Sakae Okubo of [NTT](https://en.wikipedia.org/wiki/Nippon_Telegraph_and_Telephone).[[8]](https://en.wikipedia.org/wiki/H.261#cite_note-Okubo-8) A number of companies were involved in its development, including [Hitachi](https://en.wikipedia.org/wiki/Hitachi), [PictureTel](https://en.wikipedia.org/wiki/PictureTel), [NTT](https://en.wikipedia.org/wiki/Nippon_Telegraph_and_Telephone), [BT](https://en.wikipedia.org/wiki/BT_plc), and [Toshiba](https://en.wikipedia.org/wiki/Toshiba), among others.[[9]](https://en.wikipedia.org/wiki/H.261#cite_note-h261-patents-9) Since H.261, DCT compression has been adopted by all the major video coding standards that followed.[[6]](https://en.wikipedia.org/wiki/H.261#cite_note-Ghanbari-6)

Whilst H.261 was preceded in 1984 by H.120 (which also underwent a revision in 1988 of some historic importance) as a digital video coding standard, H.261 was the first truly practical digital video coding standard (in terms of product support in significant quantities). In fact, all subsequent international video coding standards ([MPEG-1 Part 2](https://en.wikipedia.org/wiki/MPEG-1#Part_2:_Video), [H.262/MPEG-2 Part 2](https://en.wikipedia.org/wiki/H.262/MPEG-2_Part_2), [H.263](https://en.wikipedia.org/wiki/H.263), [MPEG-4 Part 2](https://en.wikipedia.org/wiki/MPEG-4_Part_2), [H.264/MPEG-4 Part 10](https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC), and [HEVC](https://en.wikipedia.org/wiki/HEVC)) have been based closely on the H.261 design. Additionally, the methods used by the H.261 development committee to collaboratively develop the standard have remained the basic operating process for subsequent standardization work in the field.[[8]](https://en.wikipedia.org/wiki/H.261#cite_note-Okubo-8)

Although H.261 was first approved as a standard in 1988, the first version was missing some significant elements necessary to make it a complete [interoperability](https://en.wikipedia.org/wiki/Interoperability) specification. Various parts of it were marked as "Under Study".[[2]](https://en.wikipedia.org/wiki/H.261#cite_note-H2611988-2) It was later revised in 1990 to add the remaining necessary aspects,[[10]](https://en.wikipedia.org/wiki/H.261#cite_note-H2611990-10) and was then revised again in 1993.[[11]](https://en.wikipedia.org/wiki/H.261#cite_note-H2611993-11) The 1993 revision added an Annex D entitled "Still image transmission", which provided a backward-compatible way to send [still images](https://en.wikipedia.org/wiki/Still_image) with 704×576 luma resolution and 352×288 chroma resolution by using a staggered 2:1 [subsampling](https://en.wikipedia.org/wiki/Decimation_(signal_processing)) horizontally and vertically to separate the picture into four sub-pictures that were sent sequentially.[[11]](https://en.wikipedia.org/wiki/H.261#cite_note-H2611993-11)

## H.261 design

The basic processing unit of the design is called a [macroblock](https://en.wikipedia.org/wiki/Macroblock), and H.261 was the first standard in which the macroblock concept appeared. Each macroblock consists of a 16×16 array of [luma](https://en.wikipedia.org/wiki/Luminance_(video)) samples and two corresponding 8×8 arrays of [chroma](https://en.wikipedia.org/wiki/Chrominance) samples, using [4:2:0 sampling](https://en.wikipedia.org/wiki/Chroma_subsampling) and a [YCbCr](https://en.wikipedia.org/wiki/YCbCr) [color space](https://en.wikipedia.org/wiki/Color_space). The coding algorithm uses a hybrid of [motion-compensated](https://en.wikipedia.org/wiki/Motion_compensation) [inter-picture prediction](https://en.wikipedia.org/wiki/Inter_prediction) and spatial [transform coding](https://en.wikipedia.org/wiki/Transform_coding) with [scalar quantization](https://en.wikipedia.org/wiki/Scalar_quantization), zig-zag scanning and [entropy encoding](https://en.wikipedia.org/wiki/Entropy_encoding).

The inter-picture prediction reduces temporal redundancy, with [motion vectors](https://en.wikipedia.org/wiki/Motion_vector) used to compensate for motion. Whilst only integer-valued motion vectors are supported in H.261, a blurring filter can be applied to the prediction signal – partially mitigating the lack of fractional-sample motion vector precision. Transform coding using an 8×8 [discrete cosine transform](https://en.wikipedia.org/wiki/Discrete_cosine_transform) (DCT) reduces the spatial redundancy. The DCT that is widely used in this regard was introduced by [N. Ahmed](https://en.wikipedia.org/wiki/N._Ahmed), T. Natarajan and [K. R. Rao](https://en.wikipedia.org/wiki/K._R._Rao) in 1974.[[12]](https://en.wikipedia.org/wiki/H.261#cite_note-12) Scalar quantization is then applied to round the transform coefficients to the appropriate precision determined by a step size control parameter, and the quantized transform coefficients are zig-zag scanned and entropy-coded (using a "[run](https://en.wikipedia.org/wiki/Run-length_encoding)-level" [variable-length code](https://en.wikipedia.org/wiki/Variable-length_code)) to remove statistical redundancy.

The H.261 standard actually only specifies how to decode the video. Encoder designers were left free to design their own encoding algorithms (such as their own [motion estimation](https://en.wikipedia.org/wiki/Motion_estimation) algorithms), as long as their output was constrained properly to allow it to be decoded by any decoder made according to the standard. Encoders are also left free to perform any pre-processing they want to their input video, and decoders are allowed to perform any post-processing they want to their decoded video prior to display. One effective post-processing technique that became a key element of the best H.261-based systems is called [deblocking](https://en.wikipedia.org/wiki/Deblocking) filtering. This reduces the appearance of block-shaped artifacts caused by the block-based [motion compensation](https://en.wikipedia.org/wiki/Motion_compensation) and spatial transform parts of the design. Indeed, blocking artifacts are probably a familiar phenomenon to almost everyone who has watched digital video. Deblocking filtering has since become an integral part of the more recent standards [H.264](https://en.wikipedia.org/wiki/H.264) and [HEVC](https://en.wikipedia.org/wiki/HEVC) (although even when using these newer standards, additional post-processing is still allowed and can enhance visual quality if performed well).

Design refinements introduced in later standardization efforts have resulted in significant improvements in compression capability relative to the H.261 design. This has resulted in H.261 becoming essentially obsolete, although it is still used as a backward-compatibility mode in some video-conferencing systems (such as [H.323](https://en.wikipedia.org/wiki/H.323)) and for some types of internet video. However, H.261 remains a major historical milestone in the field of video coding development.

**H.263**

**H.263** is a [video compression standard](https://en.wikipedia.org/wiki/Video_compression_standard) originally designed as a low-bit-rate compressed format for [videoconferencing](https://en.wikipedia.org/wiki/Videoconferencing). It was standardized by the [ITU-T](https://en.wikipedia.org/wiki/ITU-T) [Video Coding Experts Group](https://en.wikipedia.org/wiki/Video_Coding_Experts_Group) (VCEG) in a project ending in 1995/1996. It is a member of the H.26x family of video coding standards in the domain of the ITU-T.

Like previous [H.26x](https://en.wikipedia.org/wiki/H.26x) standards, H.263 is based on [discrete cosine transform](https://en.wikipedia.org/wiki/Discrete_cosine_transform) (DCT) [video compression](https://en.wikipedia.org/wiki/Video_compression).[[1]](https://en.wikipedia.org/wiki/H.263#cite_note-1) H.263 was later extended to add various additional enhanced features in 1998 and 2000. Smaller additions were also made in 1997 and 2001, and a unified edition was produced in 2005.

The H.263 standard was first designed to be utilized in [H.324](https://en.wikipedia.org/wiki/H.324) based systems ([PSTN](https://en.wikipedia.org/wiki/Public_Switched_Telephone_Network) and other [circuit-switched](https://en.wikipedia.org/wiki/Circuit_switching) network [videoconferencing](https://en.wikipedia.org/wiki/Videoconferencing) and [videotelephony](https://en.wikipedia.org/wiki/Videotelephony)), but it also found use in [H.323](https://en.wikipedia.org/wiki/H.323) ([RTP](https://en.wikipedia.org/wiki/Real-time_Transport_Protocol)/IP-based videoconferencing), [H.320](https://en.wikipedia.org/wiki/H.320) ([ISDN](https://en.wikipedia.org/wiki/Integrated_Services_Digital_Network)-based videoconferencing, where it was the most widely used video compression standard),[[2]](https://en.wikipedia.org/wiki/H.263#cite_note-2) [RTSP](https://en.wikipedia.org/wiki/RTSP) ([streaming media](https://en.wikipedia.org/wiki/Streaming_media)) and [SIP](https://en.wikipedia.org/wiki/Session_Initiation_Protocol) (IP-based videoconferencing) solutions.

H.263 is a required video coding format in [ETSI](https://en.wikipedia.org/wiki/ETSI) [3GPP](https://en.wikipedia.org/wiki/3GPP) technical specifications for [IP Multimedia Subsystem](https://en.wikipedia.org/wiki/IP_Multimedia_Subsystem) (IMS), [Multimedia Messaging Service](https://en.wikipedia.org/wiki/Multimedia_Messaging_Service) (MMS) and Transparent end-to-end Packet-switched Streaming Service (PSS).[[3]](https://en.wikipedia.org/wiki/H.263#cite_note-3)[[4]](https://en.wikipedia.org/wiki/H.263#cite_note-4)[[5]](https://en.wikipedia.org/wiki/H.263#cite_note-5) In 3GPP specifications, H.263 video is usually used in [3GP](https://en.wikipedia.org/wiki/3GP) [container format](https://en.wikipedia.org/wiki/Container_format_(digital)).

H.263 also found many applications on the internet: much [Flash Video](https://en.wikipedia.org/wiki/Flash_Video) content (as used on sites such as [YouTube](https://en.wikipedia.org/wiki/YouTube), [Google Video](https://en.wikipedia.org/wiki/Google_Video), and [MySpace](https://en.wikipedia.org/wiki/MySpace)) used to be encoded in [Sorenson Spark](https://en.wikipedia.org/wiki/Sorenson_Spark#Sorenson_Spark) format (an incomplete implementation of H.263[[6]](https://en.wikipedia.org/wiki/H.263#cite_note-incomplete-h263-6)[[7]](https://en.wikipedia.org/wiki/H.263#cite_note-almost-7)[[8]](https://en.wikipedia.org/wiki/H.263#cite_note-8)). The original version of the [RealVideo](https://en.wikipedia.org/wiki/RealVideo) codec was based on H.263 until the release of RealVideo 8.[[9]](https://en.wikipedia.org/wiki/H.263#cite_note-9)

H.263 was developed as an evolutionary improvement based on experience from [H.261](https://en.wikipedia.org/wiki/H.261) and [H.262](https://en.wikipedia.org/wiki/H.262) (aka [MPEG-2 Video](https://en.wikipedia.org/wiki/MPEG-2_Video)), the previous ITU-T standards for video compression, and the [MPEG-1](https://en.wikipedia.org/wiki/MPEG-1) standard developed in ISO/IEC. Its first version was completed in 1995 and provided a suitable replacement for [H.261](https://en.wikipedia.org/wiki/H.261) at all bit rates. It was further enhanced in projects known as H.263v2 (also known as H.263+ or H.263 1998) and H.263v3 (also known as H.263++ or H.263 2000). It was also used as the basis for the development of [MPEG-4 Part 2](https://en.wikipedia.org/wiki/MPEG-4_Part_2). MPEG-4 Part 2 is H.263 compatible in the sense that basic "baseline" H.263 bitstreams are correctly decoded by an MPEG-4 Video decoder.[[10]](https://en.wikipedia.org/wiki/H.263#cite_note-end_of_the_ride-10)[[13]](https://en.wikipedia.org/wiki/H.263#cite_note-inside_MPEG-4_part_B-13)

The next enhanced format developed by ITU-T [VCEG](https://en.wikipedia.org/wiki/VCEG) (in partnership with [MPEG](https://en.wikipedia.org/wiki/MPEG)) after H.263 was the [H.264](https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC) standard, also known as AVC and [MPEG-4](https://en.wikipedia.org/wiki/MPEG-4) part 10. As H.264 provides a significant improvement in capability beyond H.263, the H.263 standard is now considered a legacy design. Most new videoconferencing products now include H.264 as well as H.263 and [H.261](https://en.wikipedia.org/wiki/H.261) capabilities. An even-newer standard format, [HEVC](https://en.wikipedia.org/wiki/HEVC), has also been developed by VCEG and MPEG, and has begun to emerge in some applications.

**FILE FORMAT**

In a computer, a file format is the layout of a [file](https://whatis.techtarget.com/definition/file) in terms of how the data within the file is organized. A particular file format is often indicated as part of a file's name by a file name [extension](https://whatis.techtarget.com/definition/extension) ([suffix](https://whatis.techtarget.com/definition/suffix)). Conventionally, the extension is separated by a period from the name and contains three or four letters that identify the format.

Application programs must be able to recognize and possibly access data within the file. Sometimes a program may not care whether the file has the appropriate extension name because it can examine the bits in the file to see whether the format (layout) is one it recognizes. Other times, the end user must specify what program should be used to open and work with file.

WhatIs.com has a resource to help you look up file extensions manually: [Every File Format in the World](https://whatis.techtarget.com/file-extensions)

There are as many different file formats as there are different programs to process the files. A few of the more common file formats are:

* Word documents (.doc)
* Web text pages (.htm or .html)
* Web page images (.gif and .jpg)
* Adobe Postcript files (.ps)
* Adobe Acrobat files (.pdf)
* Executable programs (.exe)
* Multimedia files (.mp3 and others)

There are hundreds of different file extensions and file types used with computers, and you can find a complete list on our [computer files and file extensions](https://www.computerhope.com/dosext.htm) page. However, it would be impossible for most people to memorize all file extensions and their associated programs. Below is a list of the most common file extensions, broken into categories by type of files.

## Audio file formats by file extensions

There are several [audio](https://www.computerhope.com/jargon/a/audio.htm) file formats, standards, and file extensions used today. Below is a list of the most common audio file extensions.

* **.aif** - AIF audio file
* **.cda** - CD audio track file
* **.mid** or **.midi** - [MIDI](https://www.computerhope.com/jargon/m/midi.htm) audio file.
* **.mp3** - [MP3](https://www.computerhope.com/jargon/m/mp3.htm) audio file
* **.mpa** - [MPEG-2](https://www.computerhope.com/jargon/m/mpeg2.htm) audio file
* **.ogg** - Ogg Vorbis audio file
* **.wav** - [WAV](https://www.computerhope.com/jargon/w/wav.htm) file
* **.wma** - [WMA](https://www.computerhope.com/jargon/w/wma.htm) audio file
* **.wpl** - Windows Media Player playlist

## Compressed file extensions

Most computer users are familiar with the .zip compressed files, but there are other types of [compressed files](https://www.computerhope.com/jargon/c/compfile.htm). Below is a list of the most common compressed file extensions.

* **.7z** - [7-Zip](https://www.computerhope.com/jargon/num/7zip.htm) compressed file
* **.arj** - ARJ compressed file
* **.deb** - [Debian](https://www.computerhope.com/jargon/d/debian.htm) software package file
* **.pkg** - [Package](https://www.computerhope.com/jargon/p/package.htm) file
* **.rar** - [RAR](https://www.computerhope.com/jargon/r/rar.htm) file
* **.rpm** - [Red Hat Package Manager](https://www.computerhope.com/jargon/r/rpm.htm)
* **.tar.gz** - [Tarball](https://www.computerhope.com/jargon/t/tarball.htm) compressed file
* **.z** - Z compressed file
* **.zip** - [Zip](https://www.computerhope.com/jargon/z/zip.htm) compressed file

## Disc and media file extensions

When making an [image](https://www.computerhope.com/jargon/i/image.htm) of a [disc](https://www.computerhope.com/jargon/d/disc.htm) or other [media](https://www.computerhope.com/jargon/m/media.htm), all of the contained files are saved to an image file. Below are the most common disc image file extensions.

* **.bin** - Binary disc image
* **.dmg** - macOS X disk image
* **.iso** - [ISO](https://www.computerhope.com/jargon/i/iso.htm) disc image
* **.toast** - Toast disc image
* **.vcd** - Virtual CD

## Data and database file extensions

A [data](https://www.computerhope.com/jargon/d/data.htm) file could be any file, but for the purpose of this list, we've listed the most common data files that relate to data used for a database, errors, information, importing, and exporting.

* **.csv** - [Comma separated value](https://www.computerhope.com/jargon/c/csv.htm) file
* **.dat** - Data file
* **.db** or **.dbf** - [Database](https://www.computerhope.com/jargon/d/database.htm) file
* **.log** - [Log file](https://www.computerhope.com/jargon/l/log.htm)
* **.mdb** - [Microsoft Access](https://www.computerhope.com/jargon/a/access.htm) database file
* **.sav** - Save file (e.g., game save file)
* **.sql** - [SQL](https://www.computerhope.com/jargon/s/sql.htm) database file
* **.tar** - Linux / Unix [tarball](https://www.computerhope.com/jargon/t/tarball.htm) file archive
* **.xml** - [XML](https://www.computerhope.com/jargon/x/xml.htm) file

## E-mail file extensions

Below is a list of the most common file extensions for [e-mail](https://www.computerhope.com/jargon/e/email.htm) and related files.

* **.email** - Outlook Express e-mail message file.
* **.eml** - E-mail message file from multiple e-mail clients, including [Gmail](https://www.computerhope.com/jargon/g/gmail.htm).
* **.emlx** - Apple Mail e-mail file.
* **.msg** - [Microsoft Outlook](https://www.computerhope.com/jargon/o/outlook.htm) [e-mail message](https://www.computerhope.com/jargon/m/msg.htm) file.
* **.oft** - Microsoft Outlook e-mail template file.
* **.ost** - Microsoft Outlook offline e-mail storage file.
* **.pst** - [Microsoft Outlook e-mail storage file.](https://www.computerhope.com/jargon/p/pst.htm)
* **.vcf** - E-mail [contact](https://www.computerhope.com/jargon/c/contact.htm) file.

## Executable file extensions

The most common [executable file](https://www.computerhope.com/jargon/e/execfile.htm) are files ending with the .exe file extension. However, other files can also be run by themselves or with the aid of an [interpreter](https://www.computerhope.com/jargon/i/interpre.htm).

* **.apk** - [Android](https://www.computerhope.com/jargon/a/android.htm) package file
* **.bat** - [Batch file](https://www.computerhope.com/jargon/b/batchfil.htm)
* **.bin** - [Binary file](https://www.computerhope.com/jargon/b/bin.htm)
* **.cgi** or **.pl** - [Perl](https://www.computerhope.com/jargon/p/perl.htm) script file
* **.com** - [MS-DOS](https://www.computerhope.com/jargon/m/msdos.htm) command file
* **.exe** - [Executable file](https://www.computerhope.com/jargon/e/execfile.htm)
* **.gadget** - Windows gadget
* **.jar** - [Java Archive file](https://www.computerhope.com/jargon/j/jar.htm)
* **.msi** - Windows installer package
* **.py** - [Python](https://www.computerhope.com/jargon/p/python.htm) file
* **.wsf** - Windows Script File

## Font file extensions

Below are the most common file extensions used with [fonts](https://www.computerhope.com/jargon/f/font.htm).

* **.fnt** - Windows font file
* **.fon** - Generic font file
* **.otf** - Open type font file
* **.ttf** - [TrueType](https://www.computerhope.com/jargon/t/truetype.htm) font file

## Image file formats by file extension

There are many different image types and image file extensions that can be used when creating and saving images on the computer. Below is a list of the most common image file extensions.

* **.ai** - [Adobe Illustrator](https://www.computerhope.com/jargon/a/adobe-illustrator.htm) file
* **.bmp** - [Bitmap image](https://www.computerhope.com/jargon/b/bitmap.htm)
* **.gif** - [GIF](https://www.computerhope.com/jargon/g/gif.htm) image
* **.ico** - [Icon](https://www.computerhope.com/jargon/i/icon.htm) file
* **.jpeg** or **.jpg** - [JPEG](https://www.computerhope.com/jargon/j/jpeg.htm) image
* **.png** - [PNG](https://www.computerhope.com/jargon/p/png.htm) image
* **.ps** - [PostScript](https://www.computerhope.com/jargon/p/postscri.htm) file
* **.psd** - PSD image
* **.svg** - Scalable Vector Graphics file
* **.tif** or **.tiff** - [TIFF](https://www.computerhope.com/jargon/t/tiff.htm) image

## Internet related file extensions

The [Internet](https://www.computerhope.com/jargon/i/internet.htm) is the most used resource on the computer and because [web servers](https://www.computerhope.com/jargon/w/webserve.htm) supply the files, there are many different file extensions utilized. Below are a list of the most common file extensions on the web.

**Note**

[Image file extensions](https://www.computerhope.com/issues/ch001789.htm#image), [video file extensions](https://www.computerhope.com/issues/ch001789.htm#video), and other extensions listed on this page could also be included with these file extensions mentioned below.

* **.asp** and **.aspx** - [Active Server Page](https://www.computerhope.com/jargon/a/asp.htm) file
* **.cer** - Internet security [certificate](https://www.computerhope.com/jargon/c/certific.htm)
* **.cfm** - ColdFusion Markup file
* **.cgi** or **.pl** - [Perl](https://www.computerhope.com/jargon/p/perl.htm) script file
* **.css** - [Cascading Style Sheet](https://www.computerhope.com/jargon/c/css.htm) file
* **.htm** and **.html** - [HTML](https://www.computerhope.com/jargon/h/html.htm) file
* **.js** - [JavaScript](https://www.computerhope.com/jargon/j/javascript.htm) file
* **.jsp** - [Java Server Page](https://www.computerhope.com/jargon/j/jsp.htm) file
* **.part** - Partially [downloaded](https://www.computerhope.com/jargon/d/download.htm) file
* **.php** - [PHP](https://www.computerhope.com/jargon/p/php.htm) file
* **.py** - [Python](https://www.computerhope.com/jargon/p/python.htm) file
* **.rss** - [RSS](https://www.computerhope.com/jargon/r/rss.htm) file
* **.xhtml** - [XHTML](https://www.computerhope.com/jargon/x/xhtml.htm) file

## Presentation file formats by file extension

Today, there are a few programs that can create a presentation. Below is a list of the most common file extensions associated with [presentation programs](https://www.computerhope.com/jargon/p/presenta.htm).

* **.key** - Keynote presentation
* **.odp** - OpenOffice Impress presentation file
* **.pps** - [PowerPoint](https://www.computerhope.com/jargon/p/powerpoi.htm) slide show
* **.ppt** - PowerPoint presentation
* **.pptx** - PowerPoint Open XML presentation

## Programming files by file extensions

Many file extensions are used for programs before they are compiled, as well as programming scripts. Below is a list of the most common file extensions associated with programming.

**Note**

Many of the [Internet related file extensions](https://www.computerhope.com/issues/ch001789.htm#internet) could also be included with these file extensions mentioned below.

* **.c** - [C](https://www.computerhope.com/jargon/c/c.htm) and [C++](https://www.computerhope.com/jargon/c/cplus.htm) source code file
* **.class** - [Java](https://www.computerhope.com/jargon/j/java.htm) class file
* **.cpp** - [C++](https://www.computerhope.com/jargon/c/cplus.htm) source code file
* **.cs** - [Visual C#](https://www.computerhope.com/jargon/c/csharp.htm) source code file
* **.h** - C, C++, and Objective-C header file
* **.java** - Java Source code file
* **.pl** - [Perl](https://www.computerhope.com/jargon/p/perl.htm) script file.
* **.sh** - [Bash](https://www.computerhope.com/unix/ubash.htm) shell script
* **.swift** - Swift source code file
* **.vb** - [Visual Basic](https://www.computerhope.com/jargon/v/vb.htm) file

## Spreadsheet file formats by file extension

Below are the most common file extensions used to save [spreadsheet](https://www.computerhope.com/jargon/s/spreadsheet.htm) files to a computer.

* **.ods** - [OpenOffice](https://www.computerhope.com/jargon/o/openoffi.htm) Calc spreadsheet file
* **.xls** - Microsoft [Excel](https://www.computerhope.com/jargon/e/excel.htm) file
* **.xlsm** - Microsoft Excel file with [macros](https://www.computerhope.com/jargon/m/macro.htm)
* **.xlsx** - Microsoft Excel Open XML spreadsheet file

## System related file formats and file extensions

Like all other programs, your [operating system](https://www.computerhope.com/jargon/o/os.htm) uses files and has file extensions that are more common than others. Below is a list of the most common file extensions used on operating systems.

**Note**

The executable file extensions and all other files could also be included in this list.

* **.bak** - Backup file
* **.cab** - Windows [Cabinet](https://www.computerhope.com/jargon/c/cab.htm) file
* **.cfg** - Configuration file
* **.cpl** - Windows [Control panel](https://www.computerhope.com/jargon/c/controlp.htm) file
* **.cur** - Windows [cursor](https://www.computerhope.com/jargon/c/cursor.htm) file
* **.dll** - [DLL](https://www.computerhope.com/jargon/d/dll.htm) file
* **.dmp** - Dump file
* **.drv** - [Device driver](https://www.computerhope.com/jargon/d/driver.htm) file
* **.icns** - [macOS](https://www.computerhope.com/jargon/m/macos.htm) X icon resource file
* **.ico** - [Icon](https://www.computerhope.com/jargon/i/icon.htm) file
* **.ini** - [Initialization file](https://www.computerhope.com/jargon/i/inifile.htm)
* **.lnk** - Windows [shortcut](https://www.computerhope.com/jargon/s/shortcut.htm) file
* **.msi** - Windows installer package
* **.sys** - Windows system file
* **.tmp** - [Temporary file](https://www.computerhope.com/jargon/t/tempfile.htm)

## Video file formats by file extension

Today, several file types are associated with video files to add different types of [compression](https://www.computerhope.com/jargon/c/compress.htm), compatibility, and [DRM](https://www.computerhope.com/jargon/d/drm.htm) to video files. Below is a list of the most commonly found video file extensions.

* **.3g2** - 3GPP2 multimedia file
* **.3gp** - [3GPP](https://www.computerhope.com/jargon/num/3gpp.htm) multimedia file
* **.avi** - [AVI](https://www.computerhope.com/jargon/a/avi.htm) file
* **.flv** - [Adobe Flash file](https://www.computerhope.com/jargon/f/flash.htm)
* **.h264** - [H.264](https://www.computerhope.com/jargon/h/h264.htm) video file
* **.m4v** - Apple MP4 video file
* **.mkv** - [Matroska Multimedia Container](https://www.computerhope.com/jargon/m/mkv.htm)
* **.mov** - [Apple](https://www.computerhope.com/comp/apple.htm) [QuickTime](https://www.computerhope.com/jargon/q/quicktim.htm) movie file
* **.mp4** - [MPEG4](https://www.computerhope.com/jargon/m/mpeg4.htm) video file
* **.mpg** or **.mpeg** - [MPEG](https://www.computerhope.com/jargon/m/mpeg.htm) video file
* **.rm** - [RealMedia](https://www.computerhope.com/comp/real.htm) file
* **.swf** - [Shockwave](https://www.computerhope.com/jargon/s/shockwav.htm) flash file
* **.vob** - [DVD Video Object](https://www.computerhope.com/jargon/v/vob-converter.htm)
* **.wmv** - [Windows Media Video](https://www.computerhope.com/jargon/w/wmv.htm) file

## Word processor and text file formats by file extension

Creating [text files](https://www.computerhope.com/jargon/t/textfile.htm) and using a [word processor](https://www.computerhope.com/jargon/w/word-processor.htm) is one of the most common tasks on a computer. Below is the most common file extensions used with text files and documents.

* **.doc** and **.docx** - [Microsoft Word](https://www.computerhope.com/jargon/m/microsoft-word.htm) file
* **.odt** - [OpenOffice](https://www.computerhope.com/jargon/o/openoffi.htm) Writer document file
* **.pdf** - [PDF](https://www.computerhope.com/jargon/p/pdf.htm) file
* **.rtf** - [Rich Text Format](https://www.computerhope.com/jargon/r/rtf.htm)
* **.tex** - A LaTeX document file
* **.txt** - Plain text file
* **.wpd** - WordPerfect document

**HISTORY OF RIF**

**Resource Interchange File Format** (**RIFF**) is a generic file [container format](https://en.wikipedia.org/wiki/Container_format_(digital)) for storing data in tagged [chunks](https://en.wikipedia.org/wiki/Chunk_(information)).[[1]](https://en.wikipedia.org/wiki/Resource_Interchange_File_Format#cite_note-1) It is primarily used to store [multimedia](https://en.wikipedia.org/wiki/Multimedia) such as sound and video, though it may also be used to store any arbitrary data.[[2]](https://en.wikipedia.org/wiki/Resource_Interchange_File_Format#cite_note-LoC-2)

The Microsoft implementation is mostly known through container formats like [AVI](https://en.wikipedia.org/wiki/Audio_Video_Interleave), [ANI](https://en.wikipedia.org/wiki/ANI_(animation_file_format)) and [WAV](https://en.wikipedia.org/wiki/WAV), which use RIFF as their basis

## History

RIFF was introduced in 1991 by [Microsoft](https://en.wikipedia.org/wiki/Microsoft) and [IBM](https://en.wikipedia.org/wiki/International_Business_Machines), and was presented by Microsoft as the default format for [Windows 3.1](https://en.wikipedia.org/wiki/Windows_3.1x) multimedia files. It is based on [Electronic Arts](https://en.wikipedia.org/wiki/Electronic_Arts)' [Interchange File Format](https://en.wikipedia.org/wiki/Interchange_File_Format), introduced in 1985 on the [Commodore Amiga](https://en.wikipedia.org/wiki/Commodore_Amiga), the only difference being that multi-[byte](https://en.wikipedia.org/wiki/Byte) integers are in [little-endian](https://en.wikipedia.org/wiki/Endianness) format, native to the [80x86](https://en.wikipedia.org/wiki/80x86) processor series used in IBM PCs, rather than the big-endian format native to the [68k](https://en.wikipedia.org/wiki/68k) processor series used in [Amiga](https://en.wikipedia.org/wiki/Amiga) and [Apple Macintosh](https://en.wikipedia.org/wiki/Apple_Macintosh) computers, where IFF files were heavily used. A RIFX format, using big-endian format, was also introduced.

In 2010 Google introduced the [WebP](https://en.wikipedia.org/wiki/WebP) picture format, which uses RIFF as a container.[[4]](https://en.wikipedia.org/wiki/Resource_Interchange_File_Format#cite_note-4)

## Explanation

RIFF files consist entirely of "[chunks](https://en.wikipedia.org/wiki/Chunk_(information))". The overall format is identical to [IFF](https://en.wikipedia.org/wiki/Interchange_File_Format), except for the endianness as previously stated, and the different meaning of the chunk names.

All chunks have the following format:

* 4 bytes: an [ASCII](https://en.wikipedia.org/wiki/ASCII) identifier for this chunk (examples are "fmt " and "data"; note the space in "fmt ").
* 4 bytes: an unsigned, little-endian 32-[bit](https://en.wikipedia.org/wiki/Bit) integer with the length of this chunk (except this field itself and the chunk identifier).
* variable-sized field: the chunk data itself, of the size given in the previous field.
* a pad byte, if the chunk's length is not even.

Two chunk identifiers, "RIFF" and "LIST", introduce a chunk that can contain subchunks. The RIFF and LIST chunk data (appearing after the identifier and length) have the following format:

* 4 bytes: an ASCII identifier for this particular RIFF or LIST chunk (for RIFF in the typical case, these 4 bytes describe the content of the entire file, such as "AVI " or "WAVE").
* rest of data: subchunks.

The file itself consists of one RIFF chunk, which then can contain further subchunks: hence, the first four bytes of a correctly formatted RIFF file will spell out "R", "I", "F", "F".

More information about the RIFF format can be found in the [Interchange File Format](https://en.wikipedia.org/wiki/Interchange_File_Format) article.

[RF64](https://en.wikipedia.org/wiki/RF64) is a multichannel file format based on RIFF specification, developed by the [European Broadcasting Union](https://en.wikipedia.org/wiki/European_Broadcasting_Union). It is [BWF](https://en.wikipedia.org/wiki/Broadcast_Wave_Format)-compatible and allows file sizes to exceed 4 [gigabytes](https://en.wikipedia.org/wiki/Gigabyte). It does so by providing a "ds64" chunk with a 64-bit (8-byte) size.

## Use of the INFO chunk

The optional INFO chunk allows RIFF files to be "tagged" with information falling into a number of predefined categories, such as copyright ("ICOP"), comments ("ICMT"), artist ("IART"), in a standardised way. These details can be read from a RIFF file even if the rest of the file format is unrecognized. The standard also allows the use of user-defined fields. Programmers intending to use non-standard fields should bear in mind that the same non-standard subchunk ID may be used by different applications in different (and potentially incompatible) ways.

## Compatibility issues

### Initial difficulties with MIDI files

In line with their policy of using .RIFF for all Windows 3.1 "multimedia" files, Microsoft introduced a new variant on the existing [MIDI file](https://en.wikipedia.org/wiki/MIDI) format used for storing song information to be played on electronic musical instruments. Microsoft's "new" MIDI file format consisted of a standard MIDI file enclosed in a RIFF "wrapper", and had the file extension [.RMI](https://en.wikipedia.org/wiki/Musical_Instrument_Digital_Interface#RIFF-RMID). Since the existing MIDI file format already supported embedded "tagging" information, the advantages to the user of having a new format were not obvious.

The MIDI Manufacturers Association have since embraced the RIFF-based MIDI file format, and used it as the basis of an "extended midifile" that also includes instrument data in "[DLS](https://en.wikipedia.org/wiki/DLS_format)" format, embedded within the same .RMI file.

### INFO chunk placement problems

For cataloguing purposes, the optimal position for the INFO chunk is near the beginning of the file. However, since the INFO chunk is optional, it is often omitted from the detailed specifications of individual file formats, leading to some confusion over the correct position for this chunk within a file.

When dealing with large media files, the expansion or contraction of the INFO chunk during tag-editing can result in the following "data" section of the file having to be read and rewritten back to disk to accommodate the new header size. Since media files can be gigabytes in size, this is a potentially disk-intensive process. One workaround is to "pad out" the leading INFO chunk using dummy data (using a "dummy chunk" or "pad chunk") when the file is created. Later editing can then expand or contract the "dummy" field to keep the total size of the file header constant: an intelligently written piece of software can then overwrite just the file header when tagging data is changed, without modifying or moving the main body of the file.

Some programs have tried to address the problem by placing the INFO chunk at the end of a media file, after the main body of the file. This has resulted in two different conventions for chunk placement, with the attendant risk that some combinations of software can cause a file's INFO data to be ignored or permanently overwritten during editing. More sophisticated programs will take into account the possibility of "unexpected" chunk placement in files and respond accordingly. For instance, when the audio-editing program [Audacity](https://en.wikipedia.org/wiki/Audacity_(audio_editor)) encounters a .WAV file with end-placed INFO data, it will correctly identify and read the data, but on saving, will relocate the INFO chunk back to the file header.

Although [CorelDRAW](https://en.wikipedia.org/wiki/CorelDRAW) 10 nominally uses a RIFF file structure, the program's initial release placed the INFO chunk at the end, so that any embedded preview bitmap would not be displayed under Windows' file manager by default. A "patch" utility supplied with the program fixes this problem.

## RIFF info tags

RIFF information tags are found in WAV audio and AVI video files. Tags which are part of the [Exif](https://en.wikipedia.org/wiki/Exif) 2.2 specification (Tag ID's beginning with "I") have an underlined tag name in the HTML version of this documentation. Other tags are found in AVI files generated by [Sony Vegas](https://en.wikipedia.org/wiki/Sony_Vegas) video editing software.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tag ID** | **Tag name** | **Writable** | **Values / notes** |
| DTIM | DateTimeOriginal | N | ICC Profile "dtim" format values |
| TAPE | TapeName | N |  |

### Converting DTIM time to normal time

The field consists of two values (v[0] and v[1]) separated with a space (0x20). Sample code:

*// time in seconds - "concatenate" date & time elements with a decimal point delimiter*

TimeInSeconds = (v[0] \* (2^32) + v[1]) \* 10^(-7);

*// shift basis from Jan 1, 1601 to Unix epoch Jan 1, 1970 (369 years & leap days)*

UnixTimeStamp = TimeInSeconds - 134774 \* 24 \* 3600;

## Some common RIFF file types

* [WAV](https://en.wikipedia.org/wiki/WAV) (Windows audio)
* [AVI](https://en.wikipedia.org/wiki/Audio_Video_Interleave) (Windows audiovisual)
* [RMI](https://en.wikipedia.org/wiki/Musical_Instrument_Digital_Interface#Standard_files) (Windows "RIFF MIDIfile")
* [CDR](https://en.wikipedia.org/wiki/CorelDRAW) (CorelDRAW vector graphics file)
* [ANI](https://en.wikipedia.org/wiki/ANI_(file_format)) (Animated Windows cursors)
* [PAL](https://en.wikipedia.org/wiki/Palette_(computing)) (Palette)
* [DLS](https://en.wikipedia.org/wiki/DLS_format) (Downloadable Sounds)
* [WebP](https://en.wikipedia.org/wiki/WebP) (An image format developed by Google)
* [XMA](https://en.wikipedia.org/wiki/XMA_(audio_format)) (Microsoft [Xbox 360](https://en.wikipedia.org/wiki/Xbox_360) console audio format based on [WMA Pro](https://en.wikipedia.org/wiki/Windows_Media_Audio))

**TIFF**

The TIFF file format was born out of a specific need, and has since been developed and adopted to the point that it serves as a solution to several. Its history begins back in the 1980s, when the personal computer was still in the early stages of its rise into an essential part of our everyday lives.

Although there may not have been a computer in every household yet, many businesses were starting to integrate their use into their operations in some way or another. Printers, of course, were already ubiquitous, but in time, it became clear that there was going to be a significant need for physical, hard copies to be transferred **on** to the computer. The TIFF file format was developed with this in mind: a standard format to be used in conjunction with document scanners.

As technology quickly evolved, the TIFF format underwent several revisions to accommodate grayscale, and then full color. The format’s flexibility as an image format that’s ideal for maintaining document formatting has seen it adopted across multiple industries as the format of choice. For as much as computers have changed in the past 30 years, the versatility of TIFF has allowed it to flourish as a reliable format that’s still commonly used today.

## Why Use TIFF?

There are a number of reasons why businesses choose TIFF as a primary format for document management and digital record keeping. Some of the key points include:

* TIFF is a widely-adopted file format, so it’s viewable on virtually any computer.
* Because it is an image format, the formatting of any documents or records is always maintained across all devices, operating systems, screen sizes, etc.
* TIFF has a variety of compression options, so the outputted file can be compressed in size while still maintaining a high-quality appearance.
* TIFF is a secure document format – links and hidden data cannot be embedded in it, making it a secure choice for companies dealing with customer records or sensitive information.
* Because of TIFF’s high-quality output, they are somewhat more difficult to alter than other image formats, adding another layer of security.
* You are able to merge multiple pages into one TIFF file.

## Who Uses TIFF?

Organizations in a wide variety of industries use the TIFF file format in their daily operations.

* Because of its high quality, TIFF is a preferred format for those in the publishing profession;
* Industries with a need to store substantial client documents/records, like the medical profession;
* Businesses looking for a more secure alternative to PDF format documents;
* Businesses whose daily operations require a lot of fax transmissions.

## How to Convert to TIFF:

Depending on the size, scope and needs of your business and organization, Peernet offers a variety of products that could be right for you. Our [**TIFF Image Printer**](https://www.peernet.com/conversion-software/pdf-to-tiff-converter/) offers fast and easy conversions from a wide range of formats to TIFF, and is as simple to use as printing a document.

AVI

What is a AVI File?

AVI means Audio Video Interleave, a popular video container format developed by Microsoft back in 1992. This file format can hold different types of video and audio streams for synchronized playback. It became the most used video file format for its simplicity and compatibility across different platforms. AVI file format is commonly used for containing movies and TV shows in small size with great quality. AVI files also can contain lossless video which are great for editing in VideoStudio Pro.

How to open AVI files with VideoStudio

1. Launch VideoStudio
2. Choose File > Open
3. Find the AVI file you wish to open
4. Select the File(s)
5. Edit & Save Your File!

Open 3GP files in Windows

*Compatible with:*

Windows 10

Windows 9

Windows 8

VideoStudio Opens AVI Files and many more file formats

We designed VideoStudio to open a wide range of file format, including all of the following:

* [mkv](https://www.videostudiopro.com/en/pages/mkv-file)
* [swf](https://www.videostudiopro.com/en/pages/swf-file)
* [3gp](https://www.videostudiopro.com/en/pages/3gp-file)
* [avi](https://www.videostudiopro.com/en/pages/avi-file)
* [wmv](https://www.videostudiopro.com/en/pages/wmv-file)
* [mov](https://www.videostudiopro.com/en/pages/mov-file)
* [webm](https://www.videostudiopro.com/en/pages/webm-file)
* [mxf](https://www.videostudiopro.com/en/pages/mxf-file)
* [m4v](https://www.videostudiopro.com/en/pages/m4v-file)
* [mod](https://www.videostudiopro.com/en/pages/mod-file)
* [m2ts](https://www.videostudiopro.com/en/pages/m2ts-file)

**JPEG OBJECTIVES,ARCHITECTURE DCT ENCODING QUANTIZATION AND STASTICAL ENCODING**

**JPEG** ([/ˈdʒeɪpɛɡ/](https://en.wikipedia.org/wiki/Help:IPA/English) [*JAY-peg*](https://en.wikipedia.org/wiki/Help:Pronunciation_respelling_key))[[2]](https://en.wikipedia.org/wiki/JPEG#cite_note-2) is a commonly used method of [lossy compression](https://en.wikipedia.org/wiki/Lossy_compression) for [digital images](https://en.wikipedia.org/wiki/Digital_image), particularly for those images produced by [digital photography](https://en.wikipedia.org/wiki/Digital_photography). The degree of compression can be adjusted, allowing a selectable tradeoff between storage size and [image quality](https://en.wikipedia.org/wiki/Image_quality). JPEG typically achieves 10:1 compression with little perceptible loss in image quality.[[3]](https://en.wikipedia.org/wiki/JPEG#cite_note-3) Since its introduction in 1992, JPEG has been the most widely used [image compression](https://en.wikipedia.org/wiki/Image_compression) standard in the world,[[4]](https://en.wikipedia.org/wiki/JPEG#cite_note-Hudson-4)[[5]](https://en.wikipedia.org/wiki/JPEG#cite_note-5) and the most widely used digital [image format](https://en.wikipedia.org/wiki/Image_format), with several billion JPEG images produced every day as of 2015.[[6]](https://en.wikipedia.org/wiki/JPEG#cite_note-6)

The term "JPEG" is an initialism/acronym for the [Joint Photographic Experts Group](https://en.wikipedia.org/wiki/Joint_Photographic_Experts_Group), which created the standard in 1992. The basis for JPEG is the [discrete cosine transform](https://en.wikipedia.org/wiki/Discrete_cosine_transform) (DCT),[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1) a lossy image compression technique that was first proposed by [Nasir Ahmed](https://en.wikipedia.org/wiki/N._Ahmed) in 1972.[[7]](https://en.wikipedia.org/wiki/JPEG#cite_note-Ahmed-7) JPEG was largely responsible for the proliferation of digital images and [digital photos](https://en.wikipedia.org/wiki/Digital_photo) across the Internet, and later [social media](https://en.wikipedia.org/wiki/Social_media).[[8]](https://en.wikipedia.org/wiki/JPEG#cite_note-8)

JPEG compression is used in a number of [image file formats](https://en.wikipedia.org/wiki/Image_file_formats). JPEG/[Exif](https://en.wikipedia.org/wiki/Exif) is the most common image format used by [digital cameras](https://en.wikipedia.org/wiki/Digital_camera) and other photographic image capture devices; along with JPEG/[JFIF](https://en.wikipedia.org/wiki/JFIF), it is the most common format for storing and transmitting [photographic images](https://en.wikipedia.org/wiki/Photographic_image) on the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web).[[9]](https://en.wikipedia.org/wiki/JPEG#cite_note-9) These format variations are often not distinguished, and are simply called JPEG.

The [MIME media type](https://en.wikipedia.org/wiki/Internet_media_type) for JPEG is *image/jpeg*, except in older [Internet Explorer](https://en.wikipedia.org/wiki/Internet_Explorer) versions, which provides a MIME type of *image/pjpeg* when uploading JPEG images.[[10]](https://en.wikipedia.org/wiki/JPEG#cite_note-10) JPEG files usually have a [filename extension](https://en.wikipedia.org/wiki/Filename_extension) of .jpg or .jpeg. JPEG/JFIF supports a maximum image size of 65,535×65,535 pixels,[[11]](https://en.wikipedia.org/wiki/JPEG#cite_note-11) hence up to 4 gigapixels for an [aspect ratio](https://en.wikipedia.org/wiki/Aspect_ratio_(image)) of 1:1. In 2000, the JPEG group introduced a format intended to be a successor, [JPEG 2000](https://en.wikipedia.org/wiki/JPEG_2000), but it was unable to replace the original JPEG as the dominant image standard

The original JPEG specification published in 1992 implements processes from various earlier [research papers](https://en.wikipedia.org/wiki/Research_papers) and [patents](https://en.wikipedia.org/wiki/Patents) cited by the [CCITT](https://en.wikipedia.org/wiki/CCITT) (now [ITU-T](https://en.wikipedia.org/wiki/ITU-T)) and Joint Photographic Experts Group.[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1) The main basis for JPEG's lossy compression algorithm is the discrete cosine transform (DCT),[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1)[[13]](https://en.wikipedia.org/wiki/JPEG#cite_note-Heise-13) which was first proposed by [Nasir Ahmed](https://en.wikipedia.org/wiki/N._Ahmed) as an [image compression](https://en.wikipedia.org/wiki/Image_compression) technique in 1972.[[7]](https://en.wikipedia.org/wiki/JPEG#cite_note-Ahmed-7)[[13]](https://en.wikipedia.org/wiki/JPEG#cite_note-Heise-13) Ahmed developed a practical DCT algorithm with T. Natarajan of [Kansas State University](https://en.wikipedia.org/wiki/Kansas_State_University) and [K. R. Rao](https://en.wikipedia.org/wiki/K._R._Rao) of the [University of Texas](https://en.wikipedia.org/wiki/University_of_Texas) in 1973.[[7]](https://en.wikipedia.org/wiki/JPEG#cite_note-Ahmed-7) Their seminal 1974 paper[[14]](https://en.wikipedia.org/wiki/JPEG#cite_note-Ahmed_Discrete-14) is cited in the JPEG specification, along with several later research papers that did further work on DCT, including a 1977 paper by Wen-Hsiung Chen, C.H. Smith and S.C. Fralick that described a fast DCT algorithm,[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1)[[15]](https://en.wikipedia.org/wiki/JPEG#cite_note-Chen-15) as well as a 1978 paper by N.J. Narasinha and S.C. Fralick, and a 1984 paper by B.G. Lee.[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1) The specification also cites a 1984 paper by Wen-Hsiung Chen and W.K. Pratt as an influence on its [quantization](https://en.wikipedia.org/wiki/Quantization_(image_processing)) algorithm,[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1)[[16]](https://en.wikipedia.org/wiki/JPEG#cite_note-Chen1984-16) and [David A. Huffman](https://en.wikipedia.org/wiki/David_A._Huffman)'s 1952 paper for its [Huffman coding](https://en.wikipedia.org/wiki/Huffman_coding) algorithm.[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1)

The JPEG specification cites patents from several companies. The following patents provided the basis for its [arithmetic coding](https://en.wikipedia.org/wiki/Arithmetic_coding) algorithm.[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1)

* [IBM](https://en.wikipedia.org/wiki/IBM)
  + [U.S. Patent 4,652,856](https://www.google.com/patents/US4652856) – February 4, 1986 – Kottappuram M. A. Mohiuddin and Jorma J. Rissanen – Multiplication-free multi-alphabet arithmetic code
  + [U.S. Patent 4,905,297](https://www.google.com/patents/US4905297) – February 27, 1990 – G. Langdon, J.L. Mitchell, W.B. Pennebaker, and Jorma J. Rissanen – Arithmetic coding encoder and decoder system
  + [U.S. Patent 4,935,882](https://www.google.com/patents/US4935882) – June 19, 1990 – W.B. Pennebaker and J.L. Mitchell – Probability adaptation for arithmetic coders
* [Mitsubishi Electric](https://en.wikipedia.org/wiki/Mitsubishi_Electric)
  + [JP H02202267](https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=JPH02202267) ([1021672](https://patents.google.com/patent/JPH02202267A)) – January 21, 1989 – Toshihiro Kimura, Shigenori Kino, Fumitaka Ono, Masayuki Yoshida – Coding system
  + [JP H03247123](https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=JPH03247123) ([2-46275](https://patents.google.com/patent/JPH0834434B2/en)) – February 26, 1990 – Fumitaka Ono, Tomohiro Kimura, Masayuki Yoshida, and Shigenori Kino – Coding apparatus and coding method

The JPEG specification also cites three other patents from IBM. Other companies cited as patent holders include [AT&T](https://en.wikipedia.org/wiki/AT%26T) (two patents) and [Canon Inc.](https://en.wikipedia.org/wiki/Canon_Inc.)[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1) Absent from the list is [U.S. Patent 4,698,672](https://www.google.com/patents/US4698672), filed by [Compression Labs](https://en.wikipedia.org/wiki/Compression_Labs,_Inc.)' Wen-Hsiung Chen and Daniel J. Klenke in October 1986. The patent describes a DCT-based image compression algorithm, and would later be a cause of controversy in 2002 (see [*Patent controversy*](https://en.wikipedia.org/wiki/JPEG#Patent_controversy) below).[[17]](https://en.wikipedia.org/wiki/JPEG#cite_note-Lemos-17) However, the JPEG specification did cite two earlier research papers by Wen-Hsiung Chen, published in 1977 and 1984.[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1)

### JPEG standard

"JPEG" stands for Joint Photographic Experts Group, the name of the committee that created the JPEG standard and also other still picture coding standards. The "Joint" stood for [ISO](https://en.wikipedia.org/wiki/ISO) TC97 WG8 and CCITT SGVIII. Founded in 1986, the group developed the JPEG standard during the late 1980s. Among several [transform coding](https://en.wikipedia.org/wiki/Transform_coding) techniques they examined, they selected the [discrete cosine transform](https://en.wikipedia.org/wiki/Discrete_cosine_transform) (DCT), as it was by far the most efficient practical compression technique. The group published the JPEG standard in 1992.[[4]](https://en.wikipedia.org/wiki/JPEG#cite_note-Hudson-4)

In 1987, ISO TC 97 became ISO/IEC JTC1 and, in 1992, CCITT became ITU-T. Currently on the JTC1 side, JPEG is one of two sub-groups of [ISO](https://en.wikipedia.org/wiki/International_Organization_for_Standardization)/[IEC](https://en.wikipedia.org/wiki/International_Electrotechnical_Commission) [Joint Technical Committee 1](https://en.wikipedia.org/wiki/ISO/IEC_JTC1), Subcommittee 29, Working Group 1 ([ISO/IEC JTC 1/SC 29](https://en.wikipedia.org/wiki/ISO/IEC_JTC_1/SC_29)/WG 1) – titled as *Coding of still pictures*.[[18]](https://en.wikipedia.org/wiki/JPEG#cite_note-18)[[19]](https://en.wikipedia.org/wiki/JPEG#cite_note-ISO/IEC-19)[[20]](https://en.wikipedia.org/wiki/JPEG#cite_note-20) On the ITU-T side, ITU-T SG16 is the respective body. The original JPEG Group was organized in 1986,[[21]](https://en.wikipedia.org/wiki/JPEG#cite_note-jpeg.org-21) issuing the first JPEG standard in 1992, which was approved in September 1992 as [**ITU-T**](https://en.wikipedia.org/wiki/ITU-T)**Recommendation T.81**[[22]](https://en.wikipedia.org/wiki/JPEG#cite_note-22) and, in 1994, as **ISO/IEC 10918-1**.

The JPEG standard specifies the [codec](https://en.wikipedia.org/wiki/Codec), which defines how an image is compressed into a stream of [bytes](https://en.wikipedia.org/wiki/Byte) and decompressed back into an image, but not the file format used to contain that stream.[[23]](https://en.wikipedia.org/wiki/JPEG#cite_note-23) The Exif and JFIF standards define the commonly used file formats for interchange of JPEG-compressed images.

JPEG standards are formally named as *Information technology – Digital compression and coding of continuous-tone still images*. ISO/IEC 10918 consists of the following parts:

| **Digital compression and coding of continuous-tone still images – Parts**[[19]](https://en.wikipedia.org/wiki/JPEG#cite_note-ISO/IEC-19)[[21]](https://en.wikipedia.org/wiki/JPEG#cite_note-jpeg.org-21)[[24]](https://en.wikipedia.org/wiki/JPEG#cite_note-24) | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Part** | **ISO/IEC standard** | **ITU-T Rec.** | **First public release date** | **Latest amendment** | **Title** | **Description** |
| Part 1 | [ISO/IEC 10918-1:1994](http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=18902) | [T.81 (09/92)](http://www.itu.int/rec/T-REC-T.81) | Sep 18, 1992 |  | Requirements and guidelines |  |
| Part 2 | [ISO/IEC 10918-2:1995](http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=20689) | [T.83 (11/94)](http://www.itu.int/rec/T-REC-T.83) | Nov 11, 1994 |  | Compliance testing | Rules and checks for software conformance (to Part 1). |
| Part 3 | [ISO/IEC 10918-3:1997](http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=25037) | [T.84 (07/96)](http://www.itu.int/rec/T-REC-T.84) | Jul 3, 1996 | Apr 1, 1999 | Extensions | Set of extensions to improve the Part 1, including the **Still Picture Interchange File Format** (SPIFF).[[25]](https://en.wikipedia.org/wiki/JPEG#cite_note-25) |
| Part 4 | [ISO/IEC 10918-4:1999](http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=25431) | [T.86 (06/98)](http://www.itu.int/rec/T-REC-T.86) | Jun 18, 1998 | Jun 29, 2012 | Registration of JPEG profiles, SPIFF profiles, SPIFF tags, SPIFF colour spaces, APPn markers, SPIFF compression types and Registration Authorities (REGAUT) | methods for registering some of the parameters used to extend JPEG |
| Part 5 | [ISO/IEC 10918-5:2013](http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=54989) | [T.871 (05/11)](http://www.itu.int/rec/T-REC-T.871) | May 14, 2011 |  | JPEG File Interchange Format (JFIF) | A popular format which has been the de facto file format for images encoded by the JPEG standard. In 2009, the JPEG Committee formally established an Ad Hoc Group to standardize JFIF as JPEG Part 5.[[26]](https://en.wikipedia.org/wiki/JPEG#cite_note-26) |
| Part 6 | [ISO/IEC 10918-6:2013](http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=59634) | [T.872 (06/12)](http://www.itu.int/rec/T-REC-T.872) | Jun 2012 |  | Application to printing systems | Specifies a subset of features and application tools for the interchange of images encoded according to the ISO/IEC 10918-1 for printing. |

[Ecma International](https://en.wikipedia.org/wiki/Ecma_International) TR/98 specifies the JPEG File Interchange Format (JFIF); the first edition was published in June 2009.[[27]](https://en.wikipedia.org/wiki/JPEG#cite_note-27)

### Patent controversy

In 2002, [Forgent Networks](https://en.wikipedia.org/wiki/Forgent_Networks) asserted that it owned and would enforce patent rights on the JPEG technology, arising from a patent that had been filed on October 27, 1986, and granted on October 6, 1987: [U.S. Patent 4,698,672](https://www.google.com/patents/US4698672) by Compression Labs' Wen-Hsiung Chen and Daniel J. Klenke.[[17]](https://en.wikipedia.org/wiki/JPEG#cite_note-Lemos-17)[[28]](https://en.wikipedia.org/wiki/JPEG#cite_note-28) While Forgent did not own Compression Labs at the time, Chen later sold Compression Labs to Forgent, before Chen went on to work for [Cisco](https://en.wikipedia.org/wiki/Cisco). This led to Forgent acquiring ownership over the patent.[[17]](https://en.wikipedia.org/wiki/JPEG#cite_note-Lemos-17) Forgent's 2002 announcement created a furor reminiscent of [Unisys](https://en.wikipedia.org/wiki/Unisys)' attempts to assert its rights over the GIF image compression standard.

The JPEG committee investigated the patent claims in 2002 and were of the opinion that they were invalidated by [prior art](https://en.wikipedia.org/wiki/Prior_art),[[29]](https://en.wikipedia.org/wiki/JPEG#cite_note-29) a view shared by various experts.[[17]](https://en.wikipedia.org/wiki/JPEG#cite_note-Lemos-17)[[30]](https://en.wikipedia.org/wiki/JPEG#cite_note-30) The patent describes an image compression algorithm based on the discrete cosine transform (DCT),[[17]](https://en.wikipedia.org/wiki/JPEG#cite_note-Lemos-17) a lossy image compression technique that originated from a 1974 paper by Nasir Ahmed, T. Natarajan and [K. R. Rao](https://en.wikipedia.org/wiki/K._R._Rao).[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1)[[13]](https://en.wikipedia.org/wiki/JPEG#cite_note-Heise-13)[[14]](https://en.wikipedia.org/wiki/JPEG#cite_note-Ahmed_Discrete-14) Wen-Hsiung Chen further developed their DCT technique, describing a fast DCT algorithm in a 1977 paper with C.H. Smith and S.C. Fralick.[[15]](https://en.wikipedia.org/wiki/JPEG#cite_note-Chen-15)[[17]](https://en.wikipedia.org/wiki/JPEG#cite_note-Lemos-17) The 1992 JPEG specification cites both the 1974 Ahmed paper and the 1977 Chen paper for its DCT algorithm, as well as a 1984 paper by Chen and W.K. Pratt for its [quantization](https://en.wikipedia.org/wiki/Quantization_(signal_processing)) algorithm.[[1]](https://en.wikipedia.org/wiki/JPEG#cite_note-w3-1)[[16]](https://en.wikipedia.org/wiki/JPEG#cite_note-Chen1984-16) Compression Labs was founded by Chen, and was the first company to commercialize DCT technology.[[31]](https://en.wikipedia.org/wiki/JPEG#cite_note-31) By the time Chen had filed his patent for a DCT-based image compression algorithm with Klenke in 1986, most of what would later become the JPEG standard had already been formulated in prior literature.[[17]](https://en.wikipedia.org/wiki/JPEG#cite_note-Lemos-17) JPEG representative Richard Clark also claimed that Chen himself sat in one of the JPEG committees, but Forgent denied this claim.[[17]](https://en.wikipedia.org/wiki/JPEG#cite_note-Lemos-17)

Between 2002 and 2004, Forgent was able to obtain about US$105 million by licensing their patent to some 30 companies. In April 2004, Forgent sued 31 other companies to enforce further license payments. In July of the same year, a consortium of 21 large computer companies filed a countersuit, with the goal of invalidating the patent. In addition, Microsoft launched a separate lawsuit against Forgent in April 2005.[[32]](https://en.wikipedia.org/wiki/JPEG#cite_note-32) In February 2006, the [United States Patent and Trademark Office](https://en.wikipedia.org/wiki/United_States_Patent_and_Trademark_Office) agreed to re-examine Forgent's JPEG patent at the request of the [Public Patent Foundation](https://en.wikipedia.org/wiki/Public_Patent_Foundation).[[33]](https://en.wikipedia.org/wiki/JPEG#cite_note-33) On May 26, 2006 the USPTO found the patent invalid based on prior art. The USPTO also found that Forgent knew about the prior art, yet it intentionally avoided telling the Patent Office. This makes any appeal to reinstate the patent highly unlikely to succeed.[[34]](https://en.wikipedia.org/wiki/JPEG#cite_note-34)

Forgent also possesses a similar patent granted by the [European Patent Office](https://en.wikipedia.org/wiki/European_Patent_Office) in 1994, though it is unclear how enforceable it is.[[35]](https://en.wikipedia.org/wiki/JPEG#cite_note-35)

As of October 27, 2006, the U.S. patent's 20-year term appears to have expired, and in November 2006, Forgent agreed to abandon enforcement of patent claims against use of the JPEG standard.[[36]](https://en.wikipedia.org/wiki/JPEG#cite_note-36)

The JPEG committee has as one of its explicit goals that their standards (in particular their baseline methods) be implementable without payment of license fees, and they have secured appropriate license rights for their JPEG 2000 standard from over 20 large organizations.

Beginning in August 2007, another company, Global Patent Holdings, LLC claimed that its patent ([U.S. Patent 5,253,341](https://www.google.com/patents/US5253341)) issued in 1993, is infringed by the downloading of JPEG images on either a website or through e-mail. If not invalidated, this patent could apply to any website that displays JPEG images. The patent was under reexamination by the U.S. Patent and Trademark Office from 2000–2007; in July 2007, the Patent Office revoked all of the original claims of the patent but found that an additional claim proposed by Global Patent Holdings (claim 17) was valid.[[37]](https://en.wikipedia.org/wiki/JPEG#cite_note-37) Global Patent Holdings then filed a number of lawsuits based on claim 17 of its patent.

In its first two lawsuits following the reexamination, both filed in Chicago, Illinois, Global Patent Holdings sued the [Green Bay Packers](https://en.wikipedia.org/wiki/Green_Bay_Packers), [CDW](https://en.wikipedia.org/wiki/CDW), [Motorola](https://en.wikipedia.org/wiki/Motorola), [Apple](https://en.wikipedia.org/wiki/Apple_Inc.), [Orbitz](https://en.wikipedia.org/wiki/Orbitz), [Officemax](https://en.wikipedia.org/wiki/Officemax), [Caterpillar](https://en.wikipedia.org/wiki/Caterpillar_Inc.), [Kraft](https://en.wikipedia.org/wiki/Kraft) and [Peapod](https://en.wikipedia.org/wiki/Peapod) as defendants. A third lawsuit was filed on December 5, 2007 in South Florida against [ADT Security Services](https://en.wikipedia.org/wiki/ADT_Security_Services), [AutoNation](https://en.wikipedia.org/wiki/AutoNation), [Florida Crystals](https://en.wikipedia.org/wiki/Fanjul_brothers) Corp., HearUSA, [MovieTickets.com](https://en.wikipedia.org/wiki/MovieTickets.com), [Ocwen Financial Corp.](https://en.wikipedia.org/wiki/Ocwen_Financial_Corp.) and [Tire Kingdom](https://en.wikipedia.org/wiki/Tire_Kingdom), and a fourth lawsuit on January 8, 2008 in South Florida against the [Boca Raton Resort & Club](https://en.wikipedia.org/wiki/Boca_Raton_Resort_%26_Club). A fifth lawsuit was filed against Global Patent Holdings in Nevada. That lawsuit was filed by [Zappos.com](https://en.wikipedia.org/wiki/Zappos.com), Inc., which was allegedly threatened by Global Patent Holdings, and sought a judicial declaration that the '341 patent is invalid and not infringed.

Global Patent Holdings had also used the '341 patent to sue or threaten outspoken critics of broad software patents, including Gregory Aharonian[[38]](https://en.wikipedia.org/wiki/JPEG#cite_note-38) and the anonymous operator of a website blog known as the "[Patent Troll Tracker](https://en.wikipedia.org/wiki/Patent_Troll_Tracker)."[[39]](https://en.wikipedia.org/wiki/JPEG#cite_note-39) On December 21, 2007, patent lawyer Vernon Francissen of Chicago asked the U.S. Patent and Trademark Office to reexamine the sole remaining claim of the '341 patent on the basis of new prior art.[[40]](https://en.wikipedia.org/wiki/JPEG#cite_note-40)

On March 5, 2008, the U.S. Patent and Trademark Office agreed to reexamine the '341 patent, finding that the new prior art raised substantial new questions regarding the patent's validity.[[41]](https://en.wikipedia.org/wiki/JPEG#cite_note-41) In light of the reexamination, the accused infringers in four of the five pending lawsuits have filed motions to suspend (stay) their cases until completion of the U.S. Patent and Trademark Office's review of the '341 patent. On April 23, 2008, a judge presiding over the two lawsuits in Chicago, Illinois granted the motions in those cases.[[42]](https://en.wikipedia.org/wiki/JPEG#cite_note-42) On July 22, 2008, the Patent Office issued the first "Office Action" of the second reexamination, finding the claim invalid based on nineteen separate grounds.[[43]](https://en.wikipedia.org/wiki/JPEG#cite_note-43) On Nov. 24, 2009, a Reexamination Certificate was issued cancelling all claims.

Beginning in 2011 and continuing as of early 2013, an entity known as Princeton Digital Image Corporation,[[44]](https://en.wikipedia.org/wiki/JPEG#cite_note-44) based in Eastern Texas, began suing large numbers of companies for alleged infringement of [U.S. Patent 4,813,056](https://www.google.com/patents/US4813056). Princeton claims that the JPEG image compression standard infringes the '056 patent and has sued large numbers of websites, retailers, camera and device manufacturers and resellers. The patent was originally owned and assigned to General Electric. The patent expired in December 2007, but Princeton has sued large numbers of companies for "past infringement" of this patent. (Under U.S. patent laws, a patent owner can sue for "past infringement" up to six years before the filing of a lawsuit, so Princeton could theoretically have continued suing companies until December 2013.) As of March 2013, Princeton had suits pending in New York and Delaware against more than 55 companies. General Electric's involvement in the suit is unknown, although court records indicate that it assigned the patent to Princeton in 2009 and retains certain rights in the patent.[[45]](https://en.wikipedia.org/wiki/JPEG#cite_note-45)

## Typical usage

The JPEG compression algorithm operates at its best on photographs and paintings of realistic scenes with smooth variations of tone and color. For web usage, where reducing the amount of data used for an image is important for responsive presentation, JPEG's compression benefits make JPEG popular. JPEG/Exif is also the most common format saved by digital cameras.

However, JPEG is not well suited for line drawings and other textual or iconic graphics, where the sharp contrasts between adjacent pixels can cause noticeable artifacts. Such images are better saved in a [lossless graphics format](https://en.wikipedia.org/wiki/Lossless_data_compression#Graphics) such as [TIFF](https://en.wikipedia.org/wiki/TIFF), [GIF](https://en.wikipedia.org/wiki/Graphics_Interchange_Format), [PNG](https://en.wikipedia.org/wiki/Portable_Network_Graphics), or a [raw image format](https://en.wikipedia.org/wiki/Raw_image_format). The JPEG standard includes a lossless coding mode, but that mode is not supported in most products.

As the typical use of JPEG is a lossy compression method, which reduces the image fidelity, it is inappropriate for exact reproduction of imaging data (such as some scientific and medical imaging applications and certain technical [image processing](https://en.wikipedia.org/wiki/Image_processing) work).

JPEG is also not well suited to files that will undergo multiple edits, as some image quality is lost each time the image is recompressed, particularly if the image is cropped or shifted, or if encoding parameters are changed – see [digital generation loss](https://en.wikipedia.org/wiki/Digital_generation_loss) for details. To prevent image information loss during sequential and repetitive editing, the first edit can be saved in a lossless format, subsequently edited in that format, then finally published as JPEG for distribution.

## JPEG compression

JPEG uses a lossy form of compression based on the [discrete cosine transform](https://en.wikipedia.org/wiki/Discrete_cosine_transform) (DCT). This mathematical operation converts each frame/field of the video source from the spatial (2D) domain into the [frequency domain](https://en.wikipedia.org/wiki/Frequency_domain) (a.k.a. transform domain). A perceptual model based loosely on the human psychovisual system discards high-frequency information, i.e. sharp transitions in intensity, and [color hue](https://en.wikipedia.org/w/index.php?title=Color_hue&action=edit&redlink=1). In the transform domain, the process of reducing information is called quantization. In simpler terms, quantization is a method for optimally reducing a large number scale (with different occurrences of each number) into a smaller one, and the transform-domain is a convenient representation of the image because the high-frequency coefficients, which contribute less to the overall picture than other coefficients, are characteristically small-values with high compressibility. The quantized coefficients are then sequenced and losslessly packed into the output [bitstream](https://en.wikipedia.org/wiki/Bitstream). Nearly all software implementations of JPEG permit user control over the [compression ratio](https://en.wikipedia.org/wiki/Compression_ratio) (as well as other optional parameters), allowing the user to trade off picture-quality for smaller file size. In embedded applications (such as miniDV, which uses a similar DCT-compression scheme), the parameters are pre-selected and fixed for the application.

The compression method is usually lossy, meaning that some original image information is lost and cannot be restored, possibly affecting image quality. There is an optional [lossless](https://en.wikipedia.org/wiki/Lossless_JPEG) mode defined in the JPEG standard. However, this mode is not widely supported in products.

There is also an [interlaced](https://en.wikipedia.org/wiki/Interlace_(bitmaps)) *progressive* JPEG format, in which data is compressed in multiple passes of progressively higher detail. This is ideal for large images that will be displayed while downloading over a slow connection, allowing a reasonable preview after receiving only a portion of the data. However, support for progressive JPEGs is not universal. When progressive JPEGs are received by programs that do not support them (such as versions of [Internet Explorer](https://en.wikipedia.org/wiki/Internet_Explorer_for_Mac) before [Windows 7](https://en.wikipedia.org/wiki/Windows_7))[[46]](https://en.wikipedia.org/wiki/JPEG#cite_note-46) the software displays the image only after it has been completely downloaded.

There are also many [medical imaging](https://en.wikipedia.org/wiki/Medical_imaging), traffic and camera applications that create and process 12-bit JPEG images both grayscale and color. 12-bit JPEG format is included in an Extended part of the JPEG specification. The libjpeg codec supports 12-bit JPEG and there even exists a high-performance version.[[47]](https://en.wikipedia.org/wiki/JPEG#cite_note-47)

### Lossless editing

A number of alterations to a JPEG image can be performed losslessly (that is, without recompression and the associated quality loss) as long as the image size is a multiple of 1 MCU block (Minimum Coded Unit) (usually 16 pixels in both directions, for 4:2:0 [chroma subsampling](https://en.wikipedia.org/wiki/Chroma_subsampling)). Utilities that implement this include:

* [jpegtran](https://en.wikipedia.org/wiki/Jpegtran) and its GUI, Jpegcrop.
* [IrfanView](https://en.wikipedia.org/wiki/IrfanView) using "JPG Lossless Crop (PlugIn)" and "JPG Lossless Rotation (PlugIn)", which require installing the JPG\_TRANSFORM plugin.
* [FastStone Image Viewer](https://en.wikipedia.org/wiki/FastStone_Image_Viewer) using "Lossless Crop to File" and "JPEG Lossless Rotate".
* [XnViewMP](https://en.wikipedia.org/wiki/XnViewMP) using "JPEG lossless transformations".
* [ACDSee](https://en.wikipedia.org/wiki/ACDSee) supports lossless rotation (but not lossless cropping) with its "Force lossless JPEG operations" option.

Blocks can be rotated in 90-degree increments, flipped in the horizontal, vertical and diagonal axes and moved about in the image. Not all blocks from the original image need to be used in the modified one.

The top and left edge of a JPEG image must lie on an 8 × 8 pixel block boundary, but the bottom and right edge need not do so. This limits the possible **lossless crop** operations, and also prevents flips and rotations of an image whose bottom or right edge does not lie on a block boundary for all channels (because the edge would end up on top or left, where – as aforementioned – a block boundary is obligatory).

Rotations where the image width and height not a multiple of 8 or 16 (depending upon the chroma subsampling), are not lossless. Rotating such an image causes the blocks to be recomputed which results in loss of quality.[[48]](https://en.wikipedia.org/wiki/JPEG#cite_note-48)

When using lossless cropping, if the bottom or right side of the crop region is not on a block boundary, then the rest of the data from the partially used blocks will still be present in the cropped file and can be recovered. It is also possible to transform between baseline and progressive formats without any loss of quality, since the only difference is the order in which the coefficients are placed in the file.

Furthermore, several JPEG images can be losslessly joined together, as long as they were saved with the same quality and the edges coincide with block boundaries.

## JPEG files[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=8)]

The [file format](https://en.wikipedia.org/wiki/File_format) known as "JPEG Interchange Format" (JIF) is specified in Annex B of the standard. However, this "pure" file format is rarely used, primarily because of the difficulty of programming encoders and decoders that fully implement all aspects of the standard and because of certain shortcomings of the standard:

* Color space definition
* Component sub-sampling registration
* Pixel aspect ratio definition.

Several additional standards have evolved to address these issues. The first of these, released in 1992, was the [JPEG File Interchange Format](https://en.wikipedia.org/wiki/JPEG_File_Interchange_Format) (or JFIF), followed in recent years by [Exchangeable image file format](https://en.wikipedia.org/wiki/Exchangeable_image_file_format) (Exif) and [ICC](https://en.wikipedia.org/wiki/International_Color_Consortium) [color profiles](https://en.wikipedia.org/wiki/ICC_Profile). Both of these formats use the actual JIF byte layout, consisting of different *markers*, but in addition, employ one of the JIF standard's extension points, namely the *application markers*: JFIF uses APP0, while Exif uses APP1. Within these segments of the file that were left for future use in the JIF standard and are not read by it, these standards add specific metadata.

Thus, in some ways, JFIF is a cut-down version of the JIF standard in that it specifies certain constraints (such as not allowing all the different encoding modes), while in other ways, it is an extension of JIF due to the added metadata. The documentation for the original JFIF standard states:[[49]](https://en.wikipedia.org/wiki/JPEG#cite_note-49)

*JPEG File Interchange Format is a minimal file format which enables JPEG bitstreams to be exchanged between a wide variety of platforms and applications. This minimal format does not include any of the advanced features found in the TIFF JPEG specification or any application specific file format. Nor should it, for the only purpose of this simplified format is to allow the exchange of JPEG compressed images.*

Image files that employ JPEG compression are commonly called "JPEG files", and are stored in variants of the JIF image format. Most image capture devices (such as digital cameras) that output JPEG are actually creating files in the Exif format, the format that the camera industry has standardized on for metadata interchange. On the other hand, since the Exif standard does not allow color profiles, most image editing software stores JPEG in JFIF format, and also includes the APP1 segment from the Exif file to include the metadata in an almost-compliant way; the JFIF standard is interpreted somewhat flexibly.[[50]](https://en.wikipedia.org/wiki/JPEG#cite_note-50)

Strictly speaking, the JFIF and Exif standards are incompatible, because each specifies that its marker segment (APP0 or APP1, respectively) appear first. In practice, most JPEG files contain a JFIF marker segment that precedes the Exif header. This allows older readers to correctly handle the older format JFIF segment, while newer readers also decode the following Exif segment, being less strict about requiring it to appear first.

### JPEG filename extensions[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=9)]

The most common filename extensions for files employing JPEG compression are **.jpg** and **.jpeg**, though .jpe, .jfif and .jif are also used. It is also possible for JPEG data to be embedded in other file types – [TIFF](https://en.wikipedia.org/wiki/Tagged_Image_File_Format) encoded files often embed a JPEG image as a [thumbnail](https://en.wikipedia.org/wiki/Thumbnail) of the main image; and MP3 files can contain a JPEG of [cover art](https://en.wikipedia.org/wiki/Cover_art) in the [ID3v2](https://en.wikipedia.org/wiki/ID3v2) tag.

### Color profile[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=10)]

Many JPEG files embed an [ICC color profile](https://en.wikipedia.org/wiki/ICC_profile) ([color space](https://en.wikipedia.org/wiki/Color_space)). Commonly used color profiles include [sRGB](https://en.wikipedia.org/wiki/SRGB_color_space) and [Adobe RGB](https://en.wikipedia.org/wiki/Adobe_RGB_color_space). Because these color spaces use a non-linear transformation, the [dynamic range](https://en.wikipedia.org/wiki/Dynamic_range) of an 8-bit JPEG file is about 11 [stops](https://en.wikipedia.org/wiki/F-number); see [gamma curve](https://en.wikipedia.org/wiki/Gamma_curve).

## Syntax and structure[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=11)]

A JPEG image consists of a sequence of *segments*, each beginning with a *marker*, each of which begins with a 0xFF byte, followed by a byte indicating what kind of marker it is. Some markers consist of just those two bytes; others are followed by two bytes (high then low), indicating the length of marker-specific payload data that follows. (The length includes the two bytes for the length, but not the two bytes for the marker.) Some markers are followed by [entropy-coded](https://en.wikipedia.org/wiki/Entropy_encoding) data; the length of such a marker does not include the entropy-coded data. Note that consecutive 0xFF bytes are used as fill bytes for [padding](https://en.wikipedia.org/wiki/Data_padding) purposes, although this fill byte padding should only ever take place for markers immediately following entropy-coded scan data (see JPEG specification section B.1.1.2 and E.1.2 for details; specifically "In all cases where markers are appended after the compressed data, optional 0xFF fill bytes may precede the marker").

Within the entropy-coded data, after any 0xFF byte, a 0x00 byte is inserted by the encoder before the next byte, so that there does not appear to be a marker where none is intended, preventing framing errors. Decoders must skip this 0x00 byte. This technique, called [byte stuffing](https://en.wikipedia.org/wiki/Byte_stuffing) (see JPEG specification section F.1.2.3), is only applied to the entropy-coded data, not to marker payload data. Note however that entropy-coded data has a few markers of its own; specifically the Reset markers (0xD0 through 0xD7), which are used to isolate independent chunks of entropy-coded data to allow parallel decoding, and encoders are free to insert these Reset markers at regular intervals (although not all encoders do this).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Common JPEG markers**[[51]](https://en.wikipedia.org/wiki/JPEG#cite_note-51) | | | | |
| **Short name** | **Bytes** | **Payload** | **Name** | **Comments** |
| **SOI** | 0xFF, 0xD8 | *none* | Start Of Image |  |
| **SOF0** | 0xFF, 0xC0 | *variable size* | Start Of Frame (baseline DCT) | Indicates that this is a baseline DCT-based JPEG, and specifies the width, height, number of components, and component subsampling (e.g., 4:2:0). |
| **SOF2** | 0xFF, 0xC2 | *variable size* | Start Of Frame (progressive DCT) | Indicates that this is a progressive DCT-based JPEG, and specifies the width, height, number of components, and component subsampling (e.g., 4:2:0). |
| **DHT** | 0xFF, 0xC4 | *variable size* | Define Huffman Table(s) | Specifies one or more Huffman tables. |
| **DQT** | 0xFF, 0xDB | *variable size* | Define Quantization Table(s) | Specifies one or more quantization tables. |
| **DRI** | 0xFF, 0xDD | 4 bytes | Define Restart Interval | Specifies the interval between RST*n* markers, in Minimum Coded Units (MCUs). This marker is followed by two bytes indicating the fixed size so it can be treated like any other variable size segment. |
| **SOS** | 0xFF, 0xDA | *variable size* | Start Of Scan | Begins a top-to-bottom scan of the image. In baseline DCT JPEG images, there is generally a single scan. Progressive DCT JPEG images usually contain multiple scans. This marker specifies which slice of data it will contain, and is immediately followed by entropy-coded data. |
| **RST*n*** | 0xFF, 0xD*n* (*n*=0..7) | *none* | Restart | Inserted every *r* macroblocks, where *r* is the restart interval set by a DRI marker. Not used if there was no DRI marker. The low three bits of the marker code cycle in value from 0 to 7. |
| **APP*n*** | 0xFF, 0xE*n* | *variable size* | Application-specific | For example, an Exif JPEG file uses an APP1 marker to store metadata, laid out in a structure based closely on [TIFF](https://en.wikipedia.org/wiki/TIFF). |
| **COM** | 0xFF, 0xFE | *variable size* | Comment | Contains a text comment. |
| **EOI** | 0xFF, 0xD9 | *none* | End Of Image |  |

There are other *Start Of Frame* markers that introduce other kinds of JPEG encodings.

Since several vendors might use the same APP*n* marker type, application-specific markers often begin with a standard or vendor name (e.g., "Exif" or "Adobe") or some other identifying string.

At a restart marker, block-to-block predictor variables are reset, and the bitstream is synchronized to a byte boundary. Restart markers provide means for recovery after bitstream error, such as transmission over an unreliable network or file corruption. Since the runs of macroblocks between restart markers may be independently decoded, these runs may be decoded in parallel.

## JPEG codec example

Although a JPEG file can be encoded in various ways, most commonly it is done with JFIF encoding. The encoding process consists of several steps:

1. The representation of the colors in the image is converted from [RGB](https://en.wikipedia.org/wiki/RGB_color_model) to [Y′CBCR](https://en.wikipedia.org/wiki/YCbCr), consisting of one [luma](https://en.wikipedia.org/wiki/Luma_(video)) component (Y'), representing brightness, and two [chroma](https://en.wikipedia.org/wiki/Chrominance) components, (CB and CR), representing color. This step is sometimes skipped.
2. The resolution of the chroma data is reduced, usually by a factor of 2 or 3. This reflects the fact that the eye is less sensitive to fine color details than to fine brightness details.
3. The image is split into blocks of 8×8 pixels, and for each block, each of the Y, CB, and CR data undergoes the discrete cosine transform (DCT). A DCT is similar to a [Fourier transform](https://en.wikipedia.org/wiki/Fourier_transform) in the sense that it produces a kind of spatial frequency spectrum.
4. The amplitudes of the frequency components are quantized. Human vision is much more sensitive to small variations in color or brightness over large areas than to the strength of high-frequency brightness variations. Therefore, the magnitudes of the high-frequency components are stored with a lower accuracy than the low-frequency components. The quality setting of the encoder (for example 50 or 95 on a scale of 0–100 in the Independent JPEG Group's library[[52]](https://en.wikipedia.org/wiki/JPEG#cite_note-52)) affects to what extent the resolution of each frequency component is reduced. If an excessively low quality setting is used, the high-frequency components are discarded altogether.
5. The resulting data for all 8×8 blocks is further compressed with a lossless algorithm, a variant of [Huffman encoding](https://en.wikipedia.org/wiki/Huffman_encoding).

The decoding process reverses these steps, except the *quantization* because it is irreversible. In the remainder of this section, the encoding and decoding processes are described in more detail.

### Encoding[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=13)]

Many of the options in the JPEG standard are not commonly used, and as mentioned above, most image software uses the simpler JFIF format when creating a JPEG file, which among other things specifies the encoding method. Here is a brief description of one of the more common methods of encoding when applied to an input that has 24 [bits per pixel](https://en.wikipedia.org/wiki/Bits_per_pixel) (eight each of red, green, and blue). This particular option is a [lossy data compression](https://en.wikipedia.org/wiki/Lossy_data_compression) method.

#### Color space transformation

First, the image should be converted from RGB into a different color space called [Y′CBCR](https://en.wikipedia.org/wiki/YCbCr) (or, informally, YCbCr). It has three components Y', CB and CR: the Y' component represents the brightness of a pixel, and the CB and CR components represent the [chrominance](https://en.wikipedia.org/wiki/Chrominance) (split into blue and red components). This is basically the same color space as used by [digital color television](https://en.wikipedia.org/wiki/Digital_television) as well as digital video including [video DVDs](https://en.wikipedia.org/wiki/DVD_Video), and is similar to the way color is represented in analog [PAL](https://en.wikipedia.org/wiki/PAL) video and [MAC](https://en.wikipedia.org/wiki/Multiplexed_Analogue_Components) (but not by analog [NTSC](https://en.wikipedia.org/wiki/NTSC), which uses the [YIQ](https://en.wikipedia.org/wiki/YIQ) color space). The Y′CBCR color space conversion allows greater compression without a significant effect on perceptual image quality (or greater perceptual image quality for the same compression). The compression is more efficient because the brightness information, which is more important to the eventual perceptual quality of the image, is confined to a single channel. This more closely corresponds to the perception of color in the human visual system. The color transformation also improves compression by statistical [decorrelation](https://en.wikipedia.org/wiki/Decorrelation).

A particular conversion to Y′CBCR is specified in the JFIF standard, and should be performed for the resulting JPEG file to have maximum compatibility. However, some JPEG implementations in "highest quality" mode do not apply this step and instead keep the color information in the RGB color model,[[53]](https://en.wikipedia.org/wiki/JPEG#cite_note-53) where the image is stored in separate channels for red, green and blue brightness components. This results in less efficient compression, and would not likely be used when file size is especially important.

#### Downsampling

Due to the densities of color- and brightness-sensitive receptors in the human eye, humans can see considerably more fine detail in the brightness of an image (the Y' component) than in the hue and color saturation of an image (the Cb and Cr components). Using this knowledge, encoders can be designed to compress images more efficiently.

The transformation into the [Y′CBCR color model](https://en.wikipedia.org/wiki/YCbCr) enables the next usual step, which is to reduce the spatial resolution of the Cb and Cr components (called "[downsampling](https://en.wikipedia.org/wiki/Downsampling)" or "chroma subsampling"). The ratios at which the downsampling is ordinarily done for JPEG images are [4:4:4](https://en.wikipedia.org/wiki/YUV_4:4:4) (no downsampling), [4:2:2](https://en.wikipedia.org/wiki/YUV_4:2:2) (reduction by a factor of 2 in the horizontal direction), or (most commonly) [4:2:0](https://en.wikipedia.org/wiki/YUV_4:2:0) (reduction by a factor of 2 in both the horizontal and vertical directions). For the rest of the compression process, Y', Cb and Cr are processed separately and in a very similar manner.

#### Block splitting

After [subsampling](https://en.wikipedia.org/wiki/Chroma_Subsampling), each [channel](https://en.wikipedia.org/wiki/Channel_(digital_image)) must be split into 8×8 blocks. Depending on chroma subsampling, this yields Minimum Coded Unit (MCU) blocks of size 8×8 (4:4:4 – no subsampling), 16×8 (4:2:2), or most commonly 16×16 (4:2:0). In [video compression](https://en.wikipedia.org/wiki/Video_compression) MCUs are called [macroblocks](https://en.wikipedia.org/wiki/Macroblock).

If the data for a channel does not represent an integer number of blocks then the encoder must fill the remaining area of the incomplete blocks with some form of dummy data. Filling the edges with a fixed color (for example, black) can create [ringing artifacts](https://en.wikipedia.org/wiki/Ringing_artifact) along the visible part of the border; repeating the edge pixels is a common technique that reduces (but does not necessarily completely eliminate) such artifacts, and more sophisticated border filling techniques can also be applied.

#### Discrete cosine transform[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=17)]

[](https://en.wikipedia.org/wiki/File:JPEG_example_subimage.svg)

The 8×8 sub-image shown in 8-bit grayscale

Next, each 8×8 block of each component (Y, Cb, Cr) is converted to a [frequency-domain](https://en.wikipedia.org/wiki/Frequency-domain) representation, using a normalized, two-dimensional type-II discrete cosine transform (DCT), see Citation 1 in [discrete cosine transform](https://en.wikipedia.org/wiki/Discrete_cosine_transform#Citations). The DCT is sometimes referred to as "type-II DCT" in the context of a family of transforms as in [discrete cosine transform](https://en.wikipedia.org/wiki/Discrete_cosine_transform#DCT-II), and the corresponding inverse (IDCT) is denoted as "type-III DCT".

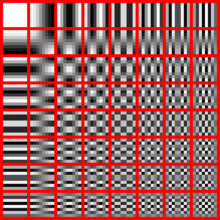
As an example, one such 8×8 8-bit subimage might be:

{\displaystyle \left[{\begin{array}{rrrrrrrr}52&55&61&66&70&61&64&73\\63&59&55&90&109&85&69&72\\62&59&68&113&144&104&66&73\\63&58&71&122&154&106&70&69\\67&61&68&104&126&88&68&70\\79&65&60&70&77&68&58&75\\85&71&64&59&55&61&65&83\\87&79&69&68&65&76&78&94\end{array}}\right].}

Before computing the DCT of the 8×8 block, its values are shifted from a positive range to one centered on zero. For an 8-bit image, each entry in the original block falls in the range {\displaystyle [0,255]}. The midpoint of the range (in this case, the value 128) is subtracted from each entry to produce a data range that is centered on zero, so that the modified range is {\displaystyle [-128,127]}. This step reduces the dynamic range requirements in the DCT processing stage that follows.

This step results in the following values:

{\displaystyle g={\begin{array}{c}x\\\longrightarrow \\\left[{\begin{array}{rrrrrrrr}-76&-73&-67&-62&-58&-67&-64&-55\\-65&-69&-73&-38&-19&-43&-59&-56\\-66&-69&-60&-15&16&-24&-62&-55\\-65&-70&-57&-6&26&-22&-58&-59\\-61&-67&-60&-24&-2&-40&-60&-58\\-49&-63&-68&-58&-51&-60&-70&-53\\-43&-57&-64&-69&-73&-67&-63&-45\\-41&-49&-59&-60&-63&-52&-50&-34\end{array}}\right]\end{array}}{\Bigg \downarrow }y.}

[](https://en.wikipedia.org/wiki/File:Dctjpeg.png)

The DCT transforms an 8×8 block of input values to a [linear combination](https://en.wikipedia.org/wiki/Linear_combination) of these 64 patterns. The patterns are referred to as the two-dimensional DCT *basis functions*, and the output values are referred to as *transform coefficients*. The horizontal index is {\displaystyle u} and the vertical index is {\displaystyle v}.

The next step is to take the two-dimensional DCT, which is given by:

{\displaystyle \ G\_{u,v}={\frac {1}{4}}\alpha (u)\alpha (v)\sum \_{x=0}^{7}\sum \_{y=0}^{7}g\_{x,y}\cos \left[{\frac {(2x+1)u\pi }{16}}\right]\cos \left[{\frac {(2y+1)v\pi }{16}}\right]}

where

* {\displaystyle \ u} is the horizontal [spatial frequency](https://en.wikipedia.org/wiki/Spatial_frequency), for the integers {\displaystyle \ 0\leq u<8}.
* {\displaystyle \ v} is the vertical spatial frequency, for the integers {\displaystyle \ 0\leq v<8}.
* {\displaystyle \alpha (u)={\begin{cases}{\frac {1}{\sqrt {2}}},&{\mbox{if }}u=0\\1,&{\mbox{otherwise}}\end{cases}}} is a normalizing scale factor to make the transformation [orthonormal](https://en.wikipedia.org/wiki/Orthonormal)
* {\displaystyle \ g\_{x,y}} is the pixel value at coordinates {\displaystyle \ (x,y)}
* {\displaystyle \ G\_{u,v}} is the DCT coefficient at coordinates {\displaystyle \ (u,v).}

If we perform this transformation on our matrix above, we get the following (rounded to the nearest two digits beyond the decimal point):

{\displaystyle G={\begin{array}{c}u\\\longrightarrow \\\left[{\begin{array}{rrrrrrrr}-415.38&-30.19&-61.20&27.24&56.12&-20.10&-2.39&0.46\\4.47&-21.86&-60.76&10.25&13.15&-7.09&-8.54&4.88\\-46.83&7.37&77.13&-24.56&-28.91&9.93&5.42&-5.65\\-48.53&12.07&34.10&-14.76&-10.24&6.30&1.83&1.95\\12.12&-6.55&-13.20&-3.95&-1.87&1.75&-2.79&3.14\\-7.73&2.91&2.38&-5.94&-2.38&0.94&4.30&1.85\\-1.03&0.18&0.42&-2.42&-0.88&-3.02&4.12&-0.66\\-0.17&0.14&-1.07&-4.19&-1.17&-0.10&0.50&1.68\end{array}}\right]\end{array}}{\Bigg \downarrow }v.}

Note the top-left corner entry with the rather large magnitude. This is the DC coefficient (also called the constant component), which defines the basic hue for the entire block. The remaining 63 coefficients are the AC coefficients (also called the alternating components).[[54]](https://en.wikipedia.org/wiki/JPEG#cite_note-54) The advantage of the DCT is its tendency to aggregate most of the signal in one corner of the result, as may be seen above. The quantization step to follow accentuates this effect while simultaneously reducing the overall size of the DCT coefficients, resulting in a signal that is easy to compress efficiently in the entropy stage.

The DCT temporarily increases the bit-depth of the data, since the DCT coefficients of an 8-bit/component image take up to 11 or more bits (depending on fidelity of the DCT calculation) to store. This may force the codec to temporarily use 16-bit numbers to hold these coefficients, doubling the size of the image representation at this point; these values are typically reduced back to 8-bit values by the quantization step. The temporary increase in size at this stage is not a performance concern for most JPEG implementations, since typically only a very small part of the image is stored in full DCT form at any given time during the image encoding or decoding process.

#### Quantization

The human eye is good at seeing small differences in [brightness](https://en.wikipedia.org/wiki/Brightness) over a relatively large area, but not so good at distinguishing the exact strength of a high frequency brightness variation. This allows one to greatly reduce the amount of information in the high frequency components. This is done by simply dividing each component in the frequency domain by a constant for that component, and then rounding to the nearest integer. This rounding operation is the only lossy operation in the whole process (other than chroma subsampling) if the DCT computation is performed with sufficiently high precision. As a result of this, it is typically the case that many of the higher frequency components are rounded to zero, and many of the rest become small positive or negative numbers, which take many fewer bits to represent.

The elements in the [quantization matrix](https://en.wikipedia.org/wiki/Quantization_matrix) control the compression ratio, with larger values producing greater compression. A typical quantization matrix (for a quality of 50% as specified in the original JPEG Standard), is as follows:

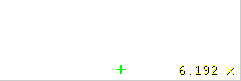
{\displaystyle Q={\begin{bmatrix}16&11&10&16&24&40&51&61\\12&12&14&19&26&58&60&55\\14&13&16&24&40&57&69&56\\14&17&22&29&51&87&80&62\\18&22&37&56&68&109&103&77\\24&35&55&64&81&104&113&92\\49&64&78&87&103&121&120&101\\72&92&95&98&112&100&103&99\end{bmatrix}}.}

The quantized DCT coefficients are computed with

{\displaystyle B\_{j,k}=\mathrm {round} \left({\frac {G\_{j,k}}{Q\_{j,k}}}\right){\mbox{ for }}j=0,1,2,\ldots ,7;k=0,1,2,\ldots ,7}

where {\displaystyle G} is the unquantized DCT coefficients; {\displaystyle Q} is the quantization matrix above; and {\displaystyle B} is the quantized DCT coefficients.

Using this quantization matrix with the DCT coefficient matrix from above results in:

[](https://en.wikipedia.org/wiki/File:Idct-animation.gif)

Left: a final image is built up from a series of basis functions. Right: each of the DCT basis functions that comprise the image, and the corresponding weighting coefficient. Middle: the basis function, after multiplication by the coefficient: this component is added to the final image. For clarity, the 8×8 macroblock in this example is magnified by 10x using bilinear interpolation.

{\displaystyle B=\left[{\begin{array}{rrrrrrrr}-26&-3&-6&2&2&-1&0&0\\0&-2&-4&1&1&0&0&0\\-3&1&5&-1&-1&0&0&0\\-3&1&2&-1&0&0&0&0\\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\end{array}}\right].}

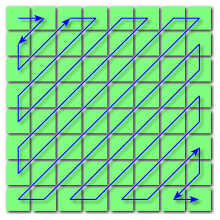
For example, using −415 (the DC coefficient) and rounding to the nearest integer

{\displaystyle \mathrm {round} \left({\frac {-415.37}{16}}\right)=\mathrm {round} \left(-25.96\right)=-26.}

Notice that most of the higher-frequency elements of the sub-block (i.e., those with an *x* or *y* spatial frequency greater than 4) are quantized into zero values.

#### Entropy coding

*Main article:*[*Entropy encoding*](https://en.wikipedia.org/wiki/Entropy_encoding)

[](https://en.wikipedia.org/wiki/File:JPEG_ZigZag.svg)

Zigzag ordering of JPEG image components

Entropy coding is a special form of [lossless data compression](https://en.wikipedia.org/wiki/Lossless_data_compression). It involves arranging the image components in a "[zigzag](https://en.wikipedia.org/wiki/Zigzag)" order employing [run-length encoding](https://en.wikipedia.org/wiki/Run-length_encoding) (RLE) algorithm that groups similar frequencies together, inserting length coding zeros, and then using Huffman coding on what is left.

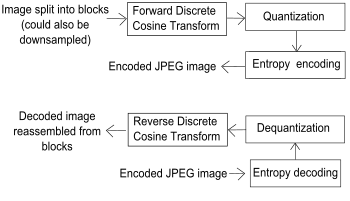
The JPEG standard also allows, but does not require, decoders to support the use of arithmetic coding, which is mathematically superior to Huffman coding. However, this feature has rarely been used, as it was historically covered by patents requiring royalty-bearing licenses, and because it is slower to encode and decode compared to Huffman coding. Arithmetic coding typically makes files about 5–7% smaller.

The previous quantized DC coefficient is used to predict the current quantized DC coefficient. The difference between the two is encoded rather than the actual value. The encoding of the 63 quantized AC coefficients does not use such prediction differencing.

The zigzag sequence for the above quantized coefficients are shown below. (The format shown is just for ease of understanding/viewing.)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| −26 |  |  |  |  |  |  |  |
| −3 | 0 |  |  |  |  |  |  |
| −3 | −2 | −6 |  |  |  |  |  |
| 2 | −4 | 1 | −3 |  |  |  |  |
| 1 | 1 | 5 | 1 | 2 |  |  |  |
| −1 | 1 | −1 | 2 | 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 |  |  |  |  |  |  |  |

If the *i*-th block is represented by {\displaystyle B\_{i}} and positions within each block are represented by {\displaystyle (p,q)} where {\displaystyle p=0,1,...,7} and {\displaystyle q=0,1,...,7}, then any coefficient in the DCT image can be represented as {\displaystyle B\_{i}(p,q)}. Thus, in the above scheme, the order of encoding pixels (for the *i*-th block) is {\displaystyle B\_{i}(0,0)}, {\displaystyle B\_{i}(0,1)}, {\displaystyle B\_{i}(1,0)}, {\displaystyle B\_{i}(2,0)}, {\displaystyle B\_{i}(1,1)}, {\displaystyle B\_{i}(0,2)}, {\displaystyle B\_{i}(0,3)}, {\displaystyle B\_{i}(1,2)} and so on.

[](https://en.wikipedia.org/wiki/File:JPEG_process.svg)

*Baseline sequential* JPEG encoding and decoding processes

This encoding mode is called baseline *sequential* encoding. Baseline JPEG also supports *progressive* encoding. While sequential encoding encodes coefficients of a single block at a time (in a zigzag manner), progressive encoding encodes similar-positioned batch of coefficients of all blocks in one go (called a *scan*), followed by the next batch of coefficients of all blocks, and so on. For example, if the image is divided into N 8×8 blocks {\displaystyle B\_{0},B\_{1},B\_{2},...,B\_{n-1}}, then a 3-scan progressive encoding encodes DC component, {\displaystyle B\_{i}(0,0)} for all blocks, i.e., for all {\displaystyle i=0,1,2,...,N-1}, in first scan. This is followed by the second scan which encoding a few more components (assuming four more components, they are {\displaystyle B\_{i}(0,1)} to {\displaystyle B\_{i}(1,1)}, still in a zigzag manner) coefficients of all blocks (so the sequence is: {\displaystyle B\_{0}(0,1),B\_{0}(1,0),B\_{0}(2,0),B\_{0}(1,1),B\_{1}(0,1),B\_{1}(1,0),...,B\_{N}(2,0),B\_{N}(1,1)}), followed by all the remained coefficients of all blocks in the last scan.

Once all similar-positioned coefficients have been encoded, the next position to be encoded is the one occurring next in the zigzag traversal as indicated in the figure above. It has been found that *baseline progressive* JPEG encoding usually gives better compression as compared to *baseline sequential* JPEG due to the ability to use different Huffman tables (see below) tailored for different frequencies on each "scan" or "pass" (which includes similar-positioned coefficients), though the difference is not too large.

In the rest of the article, it is assumed that the coefficient pattern generated is due to sequential mode.

In order to encode the above generated coefficient pattern, JPEG uses Huffman encoding. The JPEG standard provides general-purpose Huffman tables; encoders may also choose to generate Huffman tables optimized for the actual frequency distributions in images being encoded.

The process of encoding the zig-zag quantized data begins with a run-length encoding explained below, where:

* *x* is the non-zero, quantized AC coefficient.
* *RUNLENGTH* is the number of zeroes that came before this non-zero AC coefficient.
* *SIZE* is the number of bits required to represent *x*.
* *AMPLITUDE* is the bit-representation of *x*.

The run-length encoding works by examining each non-zero AC coefficient *x* and determining how many zeroes came before the previous AC coefficient. With this information, two symbols are created:

|  |  |
| --- | --- |
| **Symbol 1** | **Symbol 2** |
| (RUNLENGTH, SIZE) | (AMPLITUDE) |

Both *RUNLENGTH* and *SIZE* rest on the same byte, meaning that each only contains four bits of information. The higher bits deal with the number of zeroes, while the lower bits denote the number of bits necessary to encode the value of *x*.

This has the immediate implication of *Symbol 1* being only able store information regarding the first 15 zeroes preceding the non-zero AC coefficient. However, JPEG defines two special Huffman code words. One is for ending the sequence prematurely when the remaining coefficients are zero (called "End-of-Block" or "EOB"), and another when the run of zeroes goes beyond 15 before reaching a non-zero AC coefficient. In such a case where 16 zeroes are encountered before a given non-zero AC coefficient, *Symbol 1* is encoded "specially" as: (15, 0)(0).

The overall process continues until "EOB" – denoted by (0, 0) – is reached.

With this in mind, the sequence from earlier becomes:

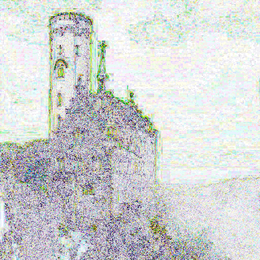
(0, 2)(-3);(1, 2)(-3);(0, 1)(-2);(0, 2)(-6);(0, 1)(2);(0, 1)(-4);(0, 1)(1);(0, 2)(-3);(0, 1)(1);(0, 1)(1);

(0, 2)(5);(0, 1)(1);(0, 1)(2);(0, 1)(-1);(0, 1)(1);(0, 1)(-1);(0, 1)(2);(5, 1)(-1);(0, 1)(-1);(0, 0);

(The first value in the matrix, −26, is the DC coefficient; it is not encoded the same way. See above.)

From here, frequency calculations are made based on occurrences of the coefficients. In our example block, most of the quantized coefficients are small numbers that are not preceded immediately by a zero coefficient. These more-frequent cases will be represented by shorter code words.

### Compression ratio and artifacts[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=20)]

[](https://en.wikipedia.org/wiki/File:Lichtenstein_jpeg_difference.png)

This image shows the pixels that are different between a non-compressed image and the same image JPEG compressed with a quality setting of 50. Darker means a larger difference. Note especially the changes occurring near sharp edges and having a block-like shape.

[](https://en.wikipedia.org/wiki/File:Lichtenstein_img_processing_test.png)

The original image

[](https://en.wikipedia.org/wiki/File:Jpegvergroessert.jpg)

The compressed 8×8 squares are visible in the scaled-up picture, together with other visual artifacts of the [lossy compression](https://en.wikipedia.org/wiki/Lossy_compression).

The resulting compression ratio can be varied according to need by being more or less aggressive in the divisors used in the quantization phase. Ten to one compression usually results in an image that cannot be distinguished by eye from the original. A compression ratio of 100:1 is usually possible, but will look distinctly [artifacted](https://en.wikipedia.org/wiki/Compression_artifact) compared to the original. The appropriate level of compression depends on the use to which the image will be put.

Those who use the World Wide Web may be familiar with the irregularities known as compression artifacts that appear in JPEG images, which may take the form of noise around contrasting edges (especially curves and corners), or "blocky" images. These are due to the quantization step of the JPEG algorithm. They are especially noticeable around sharp corners between contrasting colors (text is a good example, as it contains many such corners). The analogous artifacts in [MPEG](https://en.wikipedia.org/wiki/MPEG) video are referred to as [*mosquito noise*](https://en.wikipedia.org/wiki/Mosquito_noise)*,* as the resulting "edge busyness" and spurious dots, which change over time, resemble mosquitoes swarming around the object.[[55]](https://en.wikipedia.org/wiki/JPEG#cite_note-Dinh_and_Patry-55)[[56]](https://en.wikipedia.org/wiki/JPEG#cite_note-56)

These artifacts can be reduced by choosing a lower level of compression; they may be completely avoided by saving an image using a lossless file format, though this will result in a larger file size. The images created with [ray-tracing](https://en.wikipedia.org/wiki/Ray_tracing_(graphics)) programs have noticeable blocky shapes on the terrain. Certain low-intensity compression artifacts might be acceptable when simply viewing the images, but can be emphasized if the image is subsequently processed, usually resulting in unacceptable quality. Consider the example below, demonstrating the effect of lossy compression on an [edge detection](https://en.wikipedia.org/wiki/Edge_detection) processing step.

|  |  |  |
| --- | --- | --- |
| **Image** | **Lossless compression** | **Lossy compression** |
| **Original** | [Lossless-circle.png](https://en.wikipedia.org/wiki/File:Lossless-circle.png) | [Lossy-circle.jpg](https://en.wikipedia.org/wiki/File:Lossy-circle.jpg) |
| **Processed by** [**Canny edge detector**](https://en.wikipedia.org/wiki/Canny_edge_detector) | [Lossless-circle-canny.png](https://en.wikipedia.org/wiki/File:Lossless-circle-canny.png) | [Lossy-circle-canny.png](https://en.wikipedia.org/wiki/File:Lossy-circle-canny.png) |

Some programs allow the user to vary the amount by which individual blocks are compressed. Stronger compression is applied to areas of the image that show fewer artifacts. This way it is possible to manually reduce JPEG file size with less loss of quality.

Since the quantization stage *always* results in a loss of information, JPEG standard is always a lossy compression codec. (Information is lost both in quantizing and rounding of the floating-point numbers.) Even if the quantization matrix is a [matrix of ones](https://en.wikipedia.org/wiki/Matrix_of_ones), information will still be lost in the rounding step.

### Decoding

Decoding to display the image consists of doing all the above in reverse.

Taking the DCT coefficient matrix (after adding the difference of the DC coefficient back in)

{\displaystyle \left[{\begin{array}{rrrrrrrr}-26&-3&-6&2&2&-1&0&0\\0&-2&-4&1&1&0&0&0\\-3&1&5&-1&-1&0&0&0\\-3&1&2&-1&0&0&0&0\\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\end{array}}\right]}

and taking the [entry-for-entry product](https://en.wikipedia.org/wiki/Hadamard_product_(matrices)) with the quantization matrix from above results in

{\displaystyle \left[{\begin{array}{rrrrrrrr}-416&-33&-60&32&48&-40&0&0\\0&-24&-56&19&26&0&0&0\\-42&13&80&-24&-40&0&0&0\\-42&17&44&-29&0&0&0&0\\18&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\end{array}}\right]}

which closely resembles the original DCT coefficient matrix for the top-left portion.

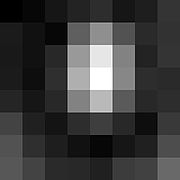
The next step is to take the two-dimensional inverse DCT (a 2D type-III DCT), which is given by:

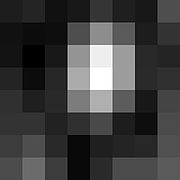
{\displaystyle f\_{x,y}={\frac {1}{4}}\sum \_{u=0}^{7}\sum \_{v=0}^{7}\alpha (u)\alpha (v)F\_{u,v}\cos \left[{\frac {(2x+1)u\pi }{16}}\right]\cos \left[{\frac {(2y+1)v\pi }{16}}\right]}

where

* {\displaystyle \ x} is the pixel row, for the integers {\displaystyle \ 0\leq x<8}.
* {\displaystyle \ y} is the pixel column, for the integers {\displaystyle \ 0\leq y<8}.
* {\displaystyle \ \alpha (u)} is defined as above, for the integers {\displaystyle \ 0\leq u<8}.
* {\displaystyle \ F\_{u,v}} is the reconstructed approximate coefficient at coordinates {\displaystyle \ (u,v).}
* {\displaystyle \ f\_{x,y}} is the reconstructed pixel value at coordinates {\displaystyle \ (x,y)}

Rounding the output to integer values (since the original had integer values) results in an image with values (still shifted down by 128)

[](https://en.wikipedia.org/wiki/File:JPEG_example_image.jpg)

[](https://en.wikipedia.org/wiki/File:JPEG_example_image_decompressed.jpg)

Slight differences are noticeable between the original (top) and decompressed image (bottom), which is most readily seen in the bottom-left corner.

{\displaystyle \left[{\begin{array}{rrrrrrrr}-66&-63&-71&-68&-56&-65&-68&-46\\-71&-73&-72&-46&-20&-41&-66&-57\\-70&-78&-68&-17&20&-14&-61&-63\\-63&-73&-62&-8&27&-14&-60&-58\\-58&-65&-61&-27&-6&-40&-68&-50\\-57&-57&-64&-58&-48&-66&-72&-47\\-53&-46&-61&-74&-65&-63&-62&-45\\-47&-34&-53&-74&-60&-47&-47&-41\end{array}}\right]}

and adding 128 to each entry

{\displaystyle \left[{\begin{array}{rrrrrrrr}62&65&57&60&72&63&60&82\\57&55&56&82&108&87&62&71\\58&50&60&111&148&114&67&65\\65&55&66&120&155&114&68&70\\70&63&67&101&122&88&60&78\\71&71&64&70&80&62&56&81\\75&82&67&54&63&65&66&83\\81&94&75&54&68&81&81&87\end{array}}\right].}

This is the decompressed subimage. In general, the decompression process may produce values outside the original input range of {\displaystyle [0,255]}. If this occurs, the decoder needs to clip the output values so as to keep them within that range to prevent overflow when storing the decompressed image with the original bit depth.

The decompressed subimage can be compared to the original subimage (also see images to the right) by taking the difference (original − uncompressed) results in the following error values:

{\displaystyle \left[{\begin{array}{rrrrrrrr}-10&-10&4&6&-2&-2&4&-9\\6&4&-1&8&1&-2&7&1\\4&9&8&2&-4&-10&-1&8\\-2&3&5&2&-1&-8&2&-1\\-3&-2&1&3&4&0&8&-8\\8&-6&-4&-0&-3&6&2&-6\\10&-11&-3&5&-8&-4&-1&-0\\6&-15&-6&14&-3&-5&-3&7\end{array}}\right]}

with an average absolute error of about 5 values per pixels (i.e., {\displaystyle {\frac {1}{64}}\sum \_{x=0}^{7}\sum \_{y=0}^{7}|e(x,y)|=4.8750}).

The error is most noticeable in the bottom-left corner where the bottom-left pixel becomes darker than the pixel to its immediate right.

### Required precision

The encoding description in the JPEG standard does not fix the precision needed for the output compressed image. However, the JPEG standard (and the similar MPEG standards) includes some precision requirements for the *de*coding, including all parts of the decoding process (variable length decoding, inverse DCT, dequantization, renormalization of outputs); the output from the reference algorithm must not exceed:

* a maximum of one bit of difference for each pixel component
* low mean square error over each 8×8-pixel block
* very low mean error over each 8×8-pixel block
* very low mean square error over the whole image
* extremely low mean error over the whole image

These assertions are tested on a large set of randomized input images, to handle the worst cases. The former IEEE 1180–1990 standard contained some similar precision requirements. The precision has a consequence on the implementation of decoders, and it is critical because some encoding processes (notably used for encoding sequences of images like MPEG) need to be able to construct, on the encoder side, a reference decoded image. In order to support 8-bit precision per pixel component output, dequantization and inverse DCT transforms are typically implemented with at least 14-bit precision in optimized decoders.

## Effects of JPEG compression[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=23)]

JPEG compression artifacts blend well into photographs with detailed non-uniform textures, allowing higher compression ratios. Notice how a higher compression ratio first affects the high-frequency textures in the upper-left corner of the image, and how the contrasting lines become more fuzzy. The very high compression ratio severely affects the quality of the image, although the overall colors and image form are still recognizable. However, the precision of colors suffer less (for a human eye) than the precision of contours (based on luminance). This justifies the fact that images should be first transformed in a color model separating the luminance from the chromatic information, before subsampling the chromatic planes (which may also use lower quality quantization) in order to preserve the precision of the luminance plane with more information bits.

### Sample photographs[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=24)]

[](https://en.wikipedia.org/wiki/File:Visual_impact_of_a_jpeg_compression_on_Photoshop.jpg)

Visual impact of a jpeg compression in Photoshop on a picture of 4480x4480 pixels

For information, the uncompressed 24-bit RGB bitmap image below (73,242 pixels) would require 219,726 bytes (excluding all other information headers). The filesizes indicated below include the internal JPEG information headers and some [metadata](https://en.wikipedia.org/wiki/Metadata). For highest quality images (Q=100), about 8.25 bits per color pixel is required[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]. On grayscale images, a minimum of 6.5 bits per pixel is enough (a comparable Q=100 quality color information requires about 25% more encoded bits). The highest quality image below (Q=100) is encoded at nine bits per color pixel, the medium quality image (Q=25) uses one bit per color pixel. For most applications, the quality factor should not go below 0.75 bit per pixel (Q=12.5), as demonstrated by the low quality image. The image at lowest quality uses only 0.13 bit per pixel, and displays very poor color. This is useful when the image will be displayed in a significantly scaled-down size. A method for creating better quantization matrices for a given image quality using [PSNR](https://en.wikipedia.org/wiki/PSNR) instead of the Q factor is described in Minguillón & Pujol (2001).[[57]](https://en.wikipedia.org/wiki/JPEG#cite_note-57)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Image** | **Quality** | **Size (bytes)** | **Compression ratio** | **Comment** |
| [JPEG example JPG RIP 100.jpg](https://en.wikipedia.org/wiki/File:JPEG_example_JPG_RIP_100.jpg) | Highest quality (Q = 100) | 81,447 | 2.7:1 | Extremely minor artifacts |
| [JPEG example JPG RIP 050.jpg](https://en.wikipedia.org/wiki/File:JPEG_example_JPG_RIP_050.jpg) | High quality (Q = 50) | 14,679 | 15:1 | Initial signs of subimage artifacts |
| [JPEG example JPG RIP 025.jpg](https://en.wikipedia.org/wiki/File:JPEG_example_JPG_RIP_025.jpg) | Medium quality (Q = 25) | 9,407 | 23:1 | Stronger artifacts; loss of high frequency information |
| [JPEG example JPG RIP 010.jpg](https://en.wikipedia.org/wiki/File:JPEG_example_JPG_RIP_010.jpg) | Low quality (Q = 10) | 4,787 | 46:1 | Severe high frequency loss leads to obvious artifacts on subimage boundaries ("macroblocking") |
| [JPEG example JPG RIP 001.jpg](https://en.wikipedia.org/wiki/File:JPEG_example_JPG_RIP_001.jpg) | Lowest quality (Q = 1) | 1,523 | 144:1 | Extreme loss of color and detail; the leaves are nearly unrecognizable. |
| **Note: The above images are not**[**IEEE**](https://en.wikipedia.org/wiki/Institute_of_Electrical_and_Electronics_Engineers)**/**[**CCIR**](https://en.wikipedia.org/wiki/ITU-R)**/**[**EBU**](https://en.wikipedia.org/wiki/European_Broadcasting_Union)[**test images**](https://en.wikipedia.org/wiki/Standard_test_image)**, and the encoder settings are not specified or available.** | | | | |

The medium quality photo uses only 4.3% of the storage space required for the uncompressed image, but has little noticeable loss of detail or visible artifacts. However, once a certain threshold of compression is passed, compressed images show increasingly visible defects. See the article on [rate–distortion theory](https://en.wikipedia.org/wiki/Rate%E2%80%93distortion_theory) for a mathematical explanation of this threshold effect. A particular limitation of JPEG in this regard is its non-overlapped 8×8 block transform structure. More modern designs such as JPEG 2000 and [JPEG XR](https://en.wikipedia.org/wiki/JPEG_XR) exhibit a more graceful degradation of quality as the bit usage decreases – by using transforms with a larger spatial extent for the lower frequency coefficients and by using overlapping transform basis functions.

## Lossless further compression[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=25)]

From 2004 to 2008, new research emerged on ways to further compress the data contained in JPEG images without modifying the represented image This has applications in scenarios where the original image is only available in JPEG format, and its size needs to be reduced for archiving or transmission. Standard general-purpose compression tools cannot significantly compress JPEG files.

Typically, such schemes take advantage of improvements to the naive scheme for coding DCT coefficients, which fails to take into account:

* Correlations between magnitudes of adjacent coefficients in the same block;
* Correlations between magnitudes of the same coefficient in adjacent blocks;
* Correlations between magnitudes of the same coefficient/block in different channels;
* The DC coefficients when taken together resemble a downscale version of the original image multiplied by a scaling factor. Well-known schemes for [lossless coding of continuous-tone images](https://en.wikipedia.org/wiki/Lossless_compression#Graphics) can be applied, achieving somewhat better compression than the [Huffman coded](https://en.wikipedia.org/wiki/Huffman_code) [DPCM](https://en.wikipedia.org/wiki/DPCM) used in JPEG.

Some standard but rarely used options already exist in JPEG to improve the efficiency of coding DCT coefficients: the arithmetic coding option, and the progressive coding option (which produces lower bitrates because values for each coefficient are coded independently, and each coefficient has a significantly different distribution). Modern methods have improved on these techniques by reordering coefficients to group coefficients of larger magnitude together;[[58]](https://en.wikipedia.org/wiki/JPEG#cite_note-Bauermann-58) using adjacent coefficients and blocks to predict new coefficient values;[[60]](https://en.wikipedia.org/wiki/JPEG#cite_note-Stirner-60) dividing blocks or coefficients up among a small number of independently coded models based on their statistics and adjacent values;[[59]](https://en.wikipedia.org/wiki/JPEG#cite_note-Ponomarenko-59)[[60]](https://en.wikipedia.org/wiki/JPEG#cite_note-Stirner-60) and most recently, by decoding blocks, predicting subsequent blocks in the spatial domain, and then encoding these to generate predictions for DCT coefficients.[[61]](https://en.wikipedia.org/wiki/JPEG#cite_note-Matsuda-61)

Typically, such methods can compress existing JPEG files between 15 and 25 percent, and for JPEGs compressed at low-quality settings, can produce improvements of up to 65%.[[60]](https://en.wikipedia.org/wiki/JPEG#cite_note-Stirner-60)[[61]](https://en.wikipedia.org/wiki/JPEG#cite_note-Matsuda-61)

A freely available tool called packJPG is based on the 2007 paper "Improved Redundancy Reduction for JPEG Files."[[62]](https://en.wikipedia.org/wiki/JPEG#cite_note-62) A 2016 paper titled "JPEG on steroids" using [ISO libjpeg](https://en.wikipedia.org/wiki/JPEG_XT#Reference_software) shows that current techniques, lossy or not, can make JPEG nearly as efficient as [JPEG XR](https://en.wikipedia.org/wiki/JPEG_XR).[[63]](https://en.wikipedia.org/wiki/JPEG#cite_note-63) [JPEG XL](https://en.wikipedia.org/wiki/JPEG#JPEG_XL) is a new file format that promises to losslessly re-encode a JPEG with efficient back-conversion to JPEG.

## Derived formats[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=26)]

### For stereoscopic 3D[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=27)]

#### JPEG Stereoscopic[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=28)]

[](https://en.wikipedia.org/wiki/File:JPS-sample.jpg)

An example of a stereoscopic .JPS file

JPS is a stereoscopic JPEG image used for creating 3D effects from 2D images. It contains two static images, one for the left eye and one for the right eye; encoded as two side-by-side images in a single JPG file. JPEG Stereoscopic (JPS, extension .jps) is a JPEG-based format for [stereoscopic](https://en.wikipedia.org/wiki/Stereoscopy) images.[[64]](https://en.wikipedia.org/wiki/JPEG#cite_note-64)[[65]](https://en.wikipedia.org/wiki/JPEG#cite_note-65) It has a range of configurations stored in the JPEG APP3 marker field, but usually contains one image of double width, representing two images of identical size in cross-eyed (i.e. left frame on the right half of the image and vice versa) side-by-side arrangement. This file format can be viewed as a JPEG without any special software, or can be processed for rendering in other modes.

#### JPEG Multi-Picture Format[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=29)]

JPEG Multi-Picture Format (MPO, extension .mpo) is a JPEG-based format for storing multiple images in a single file. It contains two or more JPEG files concatenated together.[[66]](https://en.wikipedia.org/wiki/JPEG#cite_note-66)[[67]](https://en.wikipedia.org/wiki/JPEG#cite_note-67) It also defines a JPEG APP2 marker segment for image description. Various devices use it to store 3D images, such as [Fujifilm FinePix Real 3D W1](https://en.wikipedia.org/wiki/Fujifilm_FinePix_Real_3D_W1), [HTC Evo 3D](https://en.wikipedia.org/wiki/HTC_Evo_3D), JVC GY-HMZ1U AVCHD/MVC extension camcorder, [Nintendo 3DS](https://en.wikipedia.org/wiki/Nintendo_3DS), [Sony PlayStation 3](https://en.wikipedia.org/wiki/Sony_PlayStation_3),[[68]](https://en.wikipedia.org/wiki/JPEG#cite_note-68) [Sony PlayStation Vita](https://en.wikipedia.org/wiki/Sony_PlayStation_Vita),[[69]](https://en.wikipedia.org/wiki/JPEG#cite_note-69) [Panasonic Lumix DMC-TZ20](https://en.wikipedia.org/wiki/Panasonic_Lumix_DMC-TZ20), [DMC-TZ30](https://en.wikipedia.org/wiki/Panasonic_Lumix_DMC-TZ30), [DMC-TZ60](https://en.wikipedia.org/wiki/Panasonic_Lumix_DMC-TZ60), DMC-TS4 (FT4), and [Sony](https://en.wikipedia.org/wiki/Sony) DSC-HX7V. Other devices use it to store "preview images" that can be displayed on a TV.

In the last few years, due to the growing use of stereoscopic images, much effort has been spent by the scientific community to develop algorithms for stereoscopic image compression.[[70]](https://en.wikipedia.org/wiki/JPEG#cite_note-70)[[71]](https://en.wikipedia.org/wiki/JPEG#cite_note-71)

### JPEG XT[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=30)]

*Main article:*[*JPEG XT*](https://en.wikipedia.org/wiki/JPEG_XT)

JPEG XT (ISO/IEC 18477) was published in June 2015; it extends base JPEG format with support for higher integer bit depths (up to 16 bit), high dynamic range imaging and floating-point coding, lossless coding, and alpha channel coding. Extensions are backward compatible with the base JPEG/JFIF file format and 8-bit lossy compressed image. JPEG XT uses an extensible file format based on JFIF. Extension layers are used to modify the JPEG 8-bit base layer and restore the high-resolution image. Existing software is forward compatible and can read the JPEG XT binary stream, though it would only decode the base 8-bit layer.[[72]](https://en.wikipedia.org/wiki/JPEG#cite_note-72)

### JPEG XL[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=31)]

*Main article:*[*JPEG XL*](https://en.wikipedia.org/wiki/JPEG_XL)

Since August 2017, JTC1/SC29/WG1 issued a series of draft calls for proposals on JPEG XL – the next generation image compression standard with substantially better compression efficiency (60% improvement) comparing to JPEG.[[73]](https://en.wikipedia.org/wiki/JPEG#cite_note-73) The standard is expected to exceed the still image compression performance shown by [HEVC](https://en.wikipedia.org/wiki/HEVC) HM, [Daala](https://en.wikipedia.org/wiki/Daala) and [WebP](https://en.wikipedia.org/wiki/WebP), and unlike previous efforts attempting to replace JPEG, to provide lossless more efficient recompression transport and storage option for traditional JPEG images.[[74]](https://en.wikipedia.org/wiki/JPEG#cite_note-74)[[75]](https://en.wikipedia.org/wiki/JPEG#cite_note-75)[[76]](https://en.wikipedia.org/wiki/JPEG#cite_note-Rhatushnyak-76) The core requirements include support for very high-resolution images (at least 40 MP), 8–10 bits per component, RGB/YCbCr/ICtCp color encoding, animated images, alpha channel coding, [Rec.709](https://en.wikipedia.org/wiki/Rec.709) color space ([sRGB](https://en.wikipedia.org/wiki/SRGB)) and gamma function (2.4-power), [Rec.2100](https://en.wikipedia.org/wiki/Rec.2100) [wide color gamut](https://en.wikipedia.org/wiki/Wide_color_gamut) color space (Rec.2020) and [high dynamic range](https://en.wikipedia.org/wiki/High_dynamic_range) transfer functions (PQ and HLG), and high-quality compression of synthetic images, such as bitmap fonts and gradients. The standard should also offer higher bit depths (12–16 bit integer and floating point), additional color spaces and transfer functions (such as Log C from [Arri](https://en.wikipedia.org/wiki/Arri_Alexa)), embedded preview images, lossless alpha channel encoding, image region coding, and low-complexity encoding. Any patented technologies would be licensed on a [royalty-free](https://en.wikipedia.org/wiki/Royalty-free) basis. The proposals were submitted by September 2018, leading to a committee draft in July 2019, with current target publication date in October 2019.[[77]](https://en.wikipedia.org/wiki/JPEG#cite_note-77)[[76]](https://en.wikipedia.org/wiki/JPEG#cite_note-Rhatushnyak-76)

### Incompatible JPEG standards[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=32)]

The Joint Photography Experts Group is also responsible for some other formats bearing the JPEG name, including [JPEG 2000](https://en.wikipedia.org/wiki/JPEG_2000), [JPEG XR](https://en.wikipedia.org/wiki/JPEG_XR), and [JPEG XS](https://en.wikipedia.org/wiki/JPEG_XS).

## Implementations[[edit](https://en.wikipedia.org/w/index.php?title=JPEG&action=edit&section=33)]

A very important implementation of a JPEG codec is the free programming library [*libjpeg*](https://en.wikipedia.org/wiki/Libjpeg) of the Independent JPEG Group. It was first published in 1991 and was key for the success of the standard.[[78]](https://en.wikipedia.org/wiki/JPEG#cite_note-78) This library or a direct derivative of it is used in countless applications. Recent versions introduce proprietary extensions [which broke ABI compatibility with previous versions](https://en.wikipedia.org/wiki/Libjpeg#history).

In March 2017, Google released the open source project [Guetzli](https://en.wikipedia.org/wiki/Guetzli), which trades off a much longer encoding time for smaller file size (similar to what [Zopfli](https://en.wikipedia.org/wiki/Zopfli) does for PNG and other lossless data formats).[[79]](https://en.wikipedia.org/wiki/JPEG#cite_note-79)

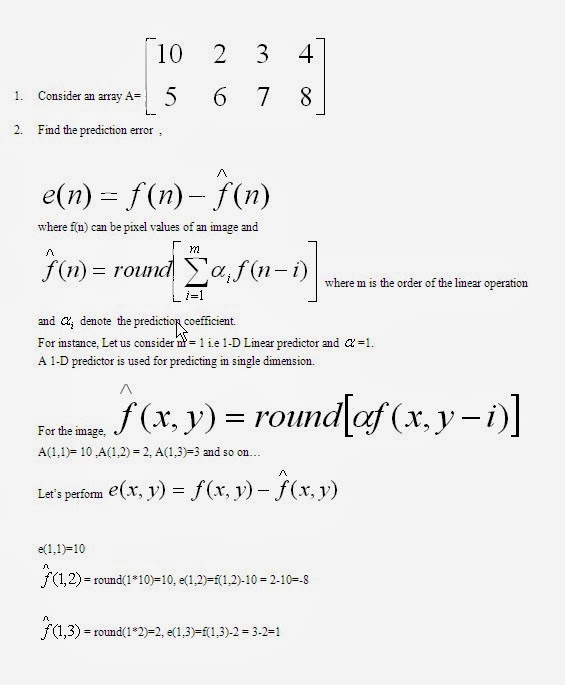
ISO/IEC [Joint Photography Experts Group](https://en.wikipedia.org/wiki/Joint_Photography_Experts_Group) maintains a reference software implementation which can encode both base JPEG (ISO/IEC 10918-1 and 18477-1) and [JPEG XT](https://en.wikipedia.org/wiki/JPEG_XT) extensions (ISO/IEC 18477 Parts 2 and 6-9), as well as [JPEG-LS](https://en.wikipedia.org/wiki/JPEG-LS) (ISO/IEC 14495).[[80]](https://en.wikipedia.org/wiki/JPEG#cite_note-80)

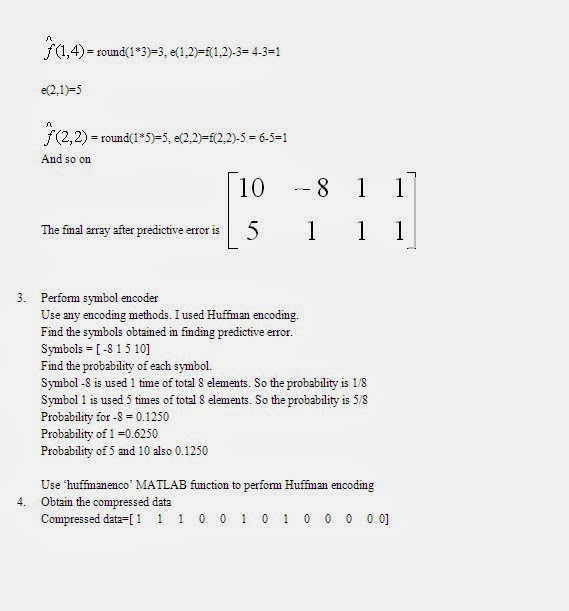
### Lossless Predictive coding - MATLAB CODE

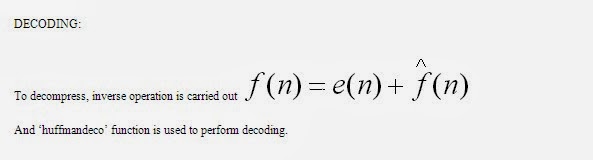
A new pixel value is obtained by finding the difference between the current pixel and the predicted pixel value.

**Encoding:**

**STEPS TO BE PERFORMED:**

[](https://3.bp.blogspot.com/-sRjEN9-tHjs/UwUPIT5jB4I/AAAAAAAABDg/upEE-yCPoFA/s1600/procedure_predictive.jpg)

[](https://2.bp.blogspot.com/-3UUEmkPa0To/UwUPK-Lj-pI/AAAAAAAABDo/LbexElrRAig/s1600/procedure_predictive2.jpg)

[](https://2.bp.blogspot.com/-B1JjfDjlGQk/UwUPMQh5K1I/AAAAAAAABDw/170M6AIc9XM/s1600/procedure_predictive3.jpg)

[MATLAB CODE:](https://www.google.com/)

clear all

clc

%Read the input image

% A=imread('rice.png');

% figure,imshow(A);

% A=double(A);

A=[10 2 3 4;5 6 7 8];

display(A);

e=A;

%Perform prediction error

for i = 1:size(A,1)

    for j = 2:size(A,2)

        e(i,j)=e(i,j)-A(i,j-1);

    end

end

display(e);

%Huffman coding

C=reshape(e,[],1);

[D1,x]=hist(C,min(min(e)):max(max(e)));

sym=x(D1>0);

prob=D1(D1>0)/numel(e);

[dict,avglen] = huffmandict(sym,prob);

comp = huffmanenco(C,dict);

%Huffman Decoding

dsig = huffmandeco(comp,dict);

e=reshape(dsig,size(A,1),size(A,2));

d=e;

for i = 1:size(A,1)

    for j = 2:size(A,2)

        d(i,j)=d(i,j-1)+e(i,j);

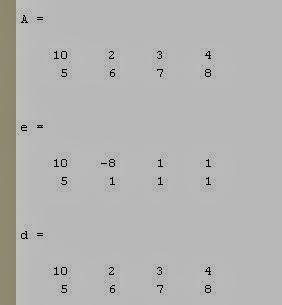
    end

end

display(d);

%Decompressed Image

%figure,imshow(uint8(d));

[](https://4.bp.blogspot.com/-HqOi3qnqkiA/UwULxD9OQ3I/AAAAAAAABCw/dxrclFGT-pg/s1600/predictive_1d.jpg)

**2-D Prediction:**

Consider an array [ a  b

                    c  d]

To perform prediction error using 2-D linear operator, e(2,2)=d-(c+b) i.e subtract the pixels left and top to the current pixel

To decompress, perform inverse operation f(2,2)=d+(c+b)

[MATLAB CODE:](https://www.google.com/)

clear all

clc

%Input sequence

A=[10 11 12 13; 2 14 26 39];

display(A);

e=A;

A1=padarray(A,[1,0],0);

%Prediction error

for i = 2:size(A1,1)-1

    for j = 2:size(A1,2)

        fx=round(A1(i,j-1)+A1(i-1,j));

         e(i-1,j)=e(i-1,j)-fx;

    end

end

display(e);

%Huffman encoding

C=reshape(e,[],1);

[D1,x]=hist(C,min(min(e)):max(max(e)));

sym=x(D1>0);

prob=D1(D1>0)/numel(e);

[dict,avglen] = huffmandict(sym,prob);

comp = huffmanenco(C,dict);

%Huffman decoding

dsig = huffmandeco(comp,dict);

e=reshape(dsig,size(A,1),size(A,2));

%Inverse operation

d=e;

e1=padarray(e,[1,0],0);

for i = 2:size(e1,1)-1

    for j = 2:size(e1,2)

        fx=round(e1(i,j-1)+e1(i-1,j));

        d(i-1,j)=d(i-1,j)+fx;

        e1=padarray(d,[1,0],0);

    end

end

display(d);

|  |
| --- |
| <https://4.bp.blogspot.com/-oyrxEsjMGxA/UwULxKc_B0I/AAAAAAAABC0/TDTAP3Ntz_Y/s1600/predictive_2d.jpg> |
| A is the input value, e is the encoded value, d is the decoded value |

The **Moving Picture Experts Group** (**MPEG**) is a [working group](https://en.wikipedia.org/wiki/Working_group) of [authorities](https://en.wikipedia.org/wiki/Experts) that was formed by [ISO](https://en.wikipedia.org/wiki/International_Organization_for_Standardization) and [IEC](https://en.wikipedia.org/wiki/International_Electrotechnical_Commission) to set standards for [audio](https://en.wikipedia.org/wiki/Audio_compression_(data)) and [video compression](https://en.wikipedia.org/wiki/Video_compression) and transmission.[[1]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-TMH-2-1) It was established in 1988 by the initiative of [Hiroshi Yasuda](https://en.wikipedia.org/wiki/Hiroshi_Yasuda) ([Nippon Telegraph and Telephone](https://en.wikipedia.org/wiki/Nippon_Telegraph_and_Telephone)) and [Leonardo Chiariglione](https://en.wikipedia.org/wiki/Leonardo_Chiariglione),[[2]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-2) group Chair since its inception. The first MPEG meeting was in May 1988 in Ottawa, Canada.[[3]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-about-mpeg-3)[[4]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-4)[[5]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-chiariglione-history-5) As of late 2005, MPEG has grown to include approximately 350 members per meeting from various industries, universities, and research institutions.[[*needs update*](https://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style/Dates_and_numbers#Chronological_items)] MPEG's official designation is [ISO/IEC JTC 1](https://en.wikipedia.org/wiki/ISO/IEC_JTC_1)/[SC 29](https://en.wikipedia.org/wiki/ISO/IEC_JTC_1/SC_29)/WG 11 – *Coding of moving pictures and audio* (ISO/IEC Joint Technical Committee 1, Subcommittee 29, Working Group 11).[[6](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-wg11-structure-6)

The MPEG standards consist of different *Parts*. Each *part* covers a certain aspect of the whole specification.[[14]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-14) The standards also specify *Profiles* and *Levels*. *Profiles* are intended to define a set of tools that are available, and *Levels* define the range of appropriate values for the properties associated with them.[[15]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-15) Some of the approved MPEG standards were revised by later amendments and/or new editions.

MPEG has standardized the following compression formats and ancillary standards. All of the MPEG formats listed below use [discrete cosine transform](https://en.wikipedia.org/wiki/Discrete_cosine_transform) (DCT) based [lossy](https://en.wikipedia.org/wiki/Lossy_compression) [video compression](https://en.wikipedia.org/wiki/Video_compression) algorithms.[[16]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-Ghanbari-16)

* [MPEG-1](https://en.wikipedia.org/wiki/MPEG-1) (1993): *Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s* (ISO/IEC 11172). This initial version is known as a lossy fileformat and is the first MPEG compression standard for [audio](https://en.wikipedia.org/wiki/Audio_compression_(data)) and [video](https://en.wikipedia.org/wiki/Video_compression). It is commonly limited to about 1.5 Mbit/s although the specification is capable of much higher bit rates. It was basically designed to allow moving pictures and sound to be encoded into the [bitrate](https://en.wikipedia.org/wiki/Bitrate) of a [Compact Disc](https://en.wikipedia.org/wiki/Compact_Disc). It is used on [Video CD](https://en.wikipedia.org/wiki/Video_CD) and can be used for low-quality video on DVD Video. It was used in digital satellite/cable TV services before MPEG-2 became widespread. To meet the low bit requirement, MPEG-1 [downsamples](https://en.wikipedia.org/wiki/Downsample) the images, as well as uses picture rates of only 24–30 Hz, resulting in a moderate quality.[[17]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-handbook_p4-17) It includes the popular MPEG-1 Audio Layer III ([MP3](https://en.wikipedia.org/wiki/MP3)) audio compression format.
* [MPEG-2](https://en.wikipedia.org/wiki/MPEG-2) (1995): *Generic coding of moving pictures and associated audio information* (ISO/IEC 13818). Transport, video and audio standards for broadcast-quality television. MPEG-2 standard was considerably broader in scope and of wider appeal – supporting [interlacing](https://en.wikipedia.org/wiki/Interlaced_video) and [high definition](https://en.wikipedia.org/wiki/High-definition_video). MPEG-2 is considered important because it has been chosen as the compression scheme for over-the-air [digital television](https://en.wikipedia.org/wiki/Digital_television) [ATSC](https://en.wikipedia.org/wiki/ATSC_Standards), [DVB](https://en.wikipedia.org/wiki/Digital_Video_Broadcasting) and [ISDB](https://en.wikipedia.org/wiki/ISDB), digital satellite TV services like [Dish Network](https://en.wikipedia.org/wiki/Dish_Network), digital [cable television](https://en.wikipedia.org/wiki/Cable_television) signals, [SVCD](https://en.wikipedia.org/wiki/SVCD) and [DVD Video](https://en.wikipedia.org/wiki/DVD_Video).[[17]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-handbook_p4-17) It is also used on [Blu-ray Discs](https://en.wikipedia.org/wiki/Blu-ray_Disc), but these normally use MPEG-4 Part 10 or SMPTE [VC-1](https://en.wikipedia.org/wiki/VC-1) for high-definition content.
* [MPEG-3](https://en.wikipedia.org/wiki/MPEG-3): MPEG-3 dealt with standardizing scalable and multi-resolution compression[[17]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-handbook_p4-17) and was intended for HDTV compression but was found to be redundant and was merged with MPEG-2; as a result there is no MPEG-3 standard.[[17]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-handbook_p4-17)[[18]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-18) MPEG-3 is not to be confused with [MP3](https://en.wikipedia.org/wiki/MP3), which is MPEG-1 or MPEG-2 Audio Layer III.
* [MPEG-4](https://en.wikipedia.org/wiki/MPEG-4) (1998): *Coding of audio-visual objects*. (ISO/IEC 14496) MPEG-4 provides a framework for more advanced compression algorithms potentially resulting in higher compression ratios compared to MPEG-2 at the cost of higher computational requirements. MPEG-4 supports Intellectual Property Management and Protection (IPMP), which provides the facility to use proprietary technologies to manage and protect content like [digital rights management](https://en.wikipedia.org/wiki/Digital_rights_management).[[19]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-19) It also supports MPEG-J, a fully programmatic solution for creation of custom interactive multimedia applications ([Java application](https://en.wikipedia.org/wiki/Java_application) environment with a [Java API](https://en.wikipedia.org/wiki/Java_API)) and many other features.[[20]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-20)[[21]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-21)[[22]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-mpeg4-part21-22) Several new higher-efficiency video standards (newer than MPEG-2 Video) are included, notably:
  + [MPEG-4 Part 2](https://en.wikipedia.org/wiki/MPEG-4_Part_2) (or Simple and Advanced Simple Profile) and
  + [MPEG-4 AVC](https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC) (or MPEG-4 Part 10 or H.264). MPEG-4 AVC may be used on [HD DVD](https://en.wikipedia.org/wiki/HD_DVD) and [Blu-ray Discs](https://en.wikipedia.org/wiki/Blu-ray_Disc), along with [VC-1](https://en.wikipedia.org/wiki/VC-1) and MPEG-2.

MPEG-4 has been chosen as the compression scheme for over-the-air in Brazil (ISDB-TB), based on original digital television from Japan (ISDB-T).[[23]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-23)

In addition, the following standards, while not sequential advances to the video encoding standard as with MPEG-1 through MPEG-4, are referred to by similar notation:

* [MPEG-7](https://en.wikipedia.org/wiki/MPEG-7) (2002): *Multimedia content description interface*. (ISO/IEC 15938)
* [MPEG-21](https://en.wikipedia.org/wiki/MPEG-21) (2001): *Multimedia framework (MPEG-21)*. (ISO/IEC 21000) MPEG describes this standard as a [*multimedia framework*](https://en.wikipedia.org/wiki/Multimedia_framework) and provides for intellectual property management and protection.

Moreover, more recently than other standards above, MPEG has started following international standards; each of the standards holds multiple MPEG technologies for a way of application.[[24]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-autogenerated1-24)[[25]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-mpeg-achievemnts-25)[[26]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-mpeg-terms-of-reference-26)[[27]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-mpeg-standards-27)[[28]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-28) (For example, MPEG-A includes a number of technologies on multimedia application format.)

* [MPEG-A](https://en.wikipedia.org/wiki/MPEG-A) (2007): *Multimedia application format (MPEG-A)*. (ISO/IEC 23000) (e.g., Purpose for multimedia application formats,[[29]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-29) MPEG music player application format, MPEG photo player application format and others)
* [MPEG-B](https://en.wikipedia.org/w/index.php?title=MPEG-B&action=edit&redlink=1) (2006): *MPEG systems technologies*. (ISO/IEC 23001) (e.g., [Binary MPEG format for XML](https://en.wikipedia.org/wiki/BiM),[[30]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-30) Fragment Request Units, Bitstream Syntax Description Language (BSDL) and others)
* [MPEG-C](https://en.wikipedia.org/w/index.php?title=MPEG-C&action=edit&redlink=1) (2006): *MPEG video technologies*. (ISO/IEC 23002) (e.g., Accuracy requirements for implementation of integer-output 8x8 inverse discrete cosine transform[[31]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-31) and others)
* [MPEG-D](https://en.wikipedia.org/wiki/MPEG-D) (2007): *MPEG audio technologies*. (ISO/IEC 23003) (e.g., [MPEG Surround](https://en.wikipedia.org/wiki/MPEG_Surround),[[32]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-32) SAOC-Spatial Audio Object Coding and USAC-[Unified Speech and Audio Coding](https://en.wikipedia.org/wiki/Unified_Speech_and_Audio_Coding))
* [MPEG-E](https://en.wikipedia.org/w/index.php?title=MPEG-E&action=edit&redlink=1) (2007): *Multimedia Middleware*. (ISO/IEC 23004) (a.k.a. M3W) (e.g., Architecture,[[33]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-33) Multimedia application programming interface (API), Component model and others)
* [MPEG-G](https://en.wikipedia.org/w/index.php?title=MPEG-G&action=edit&redlink=1) (2019): *Genomic Information Representation*. (ISO/IEC 23092) Part 1 – [Transport and Storage of Genomic Information](https://en.wikipedia.org/w/index.php?title=Transport_and_Storage_of_Genomic_Information&action=edit&redlink=1); Part 2 – [Coding of Genomic Information](https://en.wikipedia.org/w/index.php?title=Coding_of_Genomic_Information&action=edit&redlink=1); Part 3 – [APIs](https://en.wikipedia.org/wiki/Application_programming_interface); Part 4 – [Reference Software](https://en.wikipedia.org/w/index.php?title=Reference_Software&action=edit&redlink=1); Part 5 – [Conformance](https://en.wikipedia.org/wiki/Conformance_testing); Part 6 – Genomic Annotations
* *Supplemental media technologies* (2008). (ISO/IEC 29116) Part 1: Media streaming application format protocols will be revised in MPEG-M; Part 4 – MPEG extensible middleware (MXM) protocols.[[34]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-34)
* [MPEG-V](https://en.wikipedia.org/w/index.php?title=MPEG-V&action=edit&redlink=1) (2011): *Media context and control*. (ISO/IEC 23005) (a.k.a. Information exchange with Virtual Worlds)[[35]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-35)[[36]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-36) (e.g., Avatar characteristics, Sensor information, Architecture[[37]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-iso23005-1-37)[[38]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-38) and others)
* [MPEG-M](https://en.wikipedia.org/w/index.php?title=MPEG-M&action=edit&redlink=1) (2010): *MPEG eXtensible Middleware (MXM)*. (ISO/IEC 23006)[[39]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-39)[[40]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-40)[[41]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-41) (e.g., MXM architecture and technologies,[[42]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-iso23006-1-42) API, MPEG extensible middleware (MXM) protocols[[43]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-43))
* [MPEG-U](https://en.wikipedia.org/w/index.php?title=MPEG-U&action=edit&redlink=1) (2010): [*Rich media*](https://en.wikipedia.org/wiki/Rich_media)*user interfaces*. (ISO/IEC 23007)[[44]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-iso23007-1-44)[[45]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-45) (e.g., Widgets)
* [MPEG-H](https://en.wikipedia.org/wiki/MPEG-H) (2013): *High Efficiency Coding and Media Delivery in Heterogeneous Environments*. (ISO/IEC 23008) Part 1 – [MPEG media transport](https://en.wikipedia.org/wiki/MPEG_media_transport); Part 2 – [High Efficiency Video Coding](https://en.wikipedia.org/wiki/High_Efficiency_Video_Coding); Part 3 – [3D Audio](https://en.wikipedia.org/wiki/MPEG-H_3D_Audio).
* [MPEG-DASH](https://en.wikipedia.org/wiki/MPEG-DASH) (2012): *Information technology – Dynamic adaptive streaming over HTTP (DASH)*. (ISO/IEC 23009) Part 1 – Media presentation description and segment formats
* [MPEG-I](https://en.wikipedia.org/w/index.php?title=MPEG-I&action=edit&redlink=1) (2020): *Coded Representation of Immersive Media*.[[46]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-46) (ISO/IEC 23090) Part 3 - [Versatile Video Coding](https://en.wikipedia.org/wiki/Versatile_Video_Coding), Part-2 OMAF ([Omnidirectional Media Format](https://en.wikipedia.org/w/index.php?title=Omnidirectional_Media_Format&action=edit&redlink=1)).

| **MPEG groups of standards**[[25]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-mpeg-achievemnts-25)[[26]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-mpeg-terms-of-reference-26)[[27]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-mpeg-standards-27)[[47]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-47)[[48]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-48) | | | | |
| --- | --- | --- | --- | --- |
| **Acronym for a group of standards** | **Title** | **ISO/IEC standards** | **First public release date (First edition)** | **Description** |
| [MPEG-1](https://en.wikipedia.org/wiki/MPEG-1) | Coding of moving pictures and associated audio for digital storage media. Commonly limited to about 1.5 Mbit/s although specification is capable of much higher bit rates | ISO/IEC 11172 | 1993 |  |
| [MPEG-2](https://en.wikipedia.org/wiki/MPEG-2) | Generic coding of moving pictures and associated audio information | ISO/IEC 13818 | 1995 |  |
| [MPEG-3](https://en.wikipedia.org/wiki/MPEG-3) |  |  |  | abandoned, incorporated into MPEG-2 |
| [MPEG-4](https://en.wikipedia.org/wiki/MPEG-4) | Coding of audio-visual objects | ISO/IEC 14496 | 1999 |  |
| [MPEG-7](https://en.wikipedia.org/wiki/MPEG-7) | Multimedia content description interface | ISO/IEC 15938 | 2002 |  |
| [MPEG-21](https://en.wikipedia.org/wiki/MPEG-21) | Multimedia framework (MPEG-21) | ISO/IEC 21000 | 2001 |  |
| MPEG-A | Multimedia application format (MPEG-A) | ISO/IEC 23000 | 2007 |  |
| MPEG-B | MPEG systems technologies | ISO/IEC 23001 | 2006 |  |
| MPEG-C | MPEG video technologies | ISO/IEC 23002 | 2006 |  |
| [MPEG-D](https://en.wikipedia.org/wiki/MPEG-D) | MPEG audio technologies | ISO/IEC 23003 | 2007 |  |
| MPEG-E | Multimedia Middleware | ISO/IEC 23004 | 2007 |  |
| MPEG-G | Genomic Information Representation | ISO/IEC 23092 | 2019 |  |
| (none) | Supplemental media technologies | ISO/IEC 29116 | 2008 | will be revised in MPEG-M Part 4 – MPEG extensible middleware (MXM) protocols |
| MPEG-V | Media context and control | ISO/IEC 23005[[37]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-iso23005-1-37) | 2011 |  |
| MPEG-M | MPEG extensible middleware (MXM) | ISO/IEC 23006[[42]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-iso23006-1-42) | 2010 |  |
| MPEG-U | Rich media user interfaces | ISO/IEC 23007[[44]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-iso23007-1-44) | 2010 |  |
| [MPEG-H](https://en.wikipedia.org/wiki/MPEG-H) | High Efficiency Coding and Media Delivery in Heterogeneous Environments | ISO/IEC 23008[[49]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-HEVCNovember2013ISOIEC-49) | 2013 |  |
| [MPEG-DASH](https://en.wikipedia.org/wiki/MPEG-DASH) | Information technology – DASH | ISO/IEC 23009 | 2012 |  |
| [MPEG-I](https://en.wikipedia.org/w/index.php?title=MPEG-I&action=edit&redlink=1) | Coded Representation of Immersive Media | ISO/IEC 23090 | TBD (2020) |  |

## Standardization process[[edit](https://en.wikipedia.org/w/index.php?title=Moving_Picture_Experts_Group&action=edit&section=6)]

*Main article:*[*International Organization for Standardization § Standardization process*](https://en.wikipedia.org/wiki/International_Organization_for_Standardization#Standardization_process)

A standard published by ISO/IEC is the last stage of a long process that starts with the proposal of new work within a committee. Here are some abbreviations used for marking a standard with its status:[[3]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-about-mpeg-3)[[50]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-50)[[51]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-iso-stages-51)[[52]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-acronyms-committees-52)[[53]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-iso-directives-procedures-53)[[54]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-abbr-54)

* PWI – Preliminary Work Item
* NP or NWIP – New Proposal / New Work Item Proposal (e.g., ISO/IEC NP 23007)
* AWI – Approved new Work Item (e.g., ISO/IEC AWI 15444-14)
* WD – Working Draft
* CD – Committee Draft (e.g., ISO/IEC CD 23000-5)
* FCD – Final Committee Draft (e.g., ISO/IEC FCD 23000-12)
* DIS – Draft International Standard
* FDIS – Final Draft International Standard
* PRF – Proof of a new International Standard
* IS – International Standard (e.g., ISO/IEC 13818-1:2007)
* CD Amd / PDAmd (PDAM) – Committee Draft Amendment / Proposed Draft Amendment (e.g., ISO/IEC 13818-1:2007/CD Amd 6)
* FPDAmd / DAM (DAmd) – Final Proposed Draft Amendment / Draft Amendment (e.g., ISO/IEC 14496-14:2003/FPDAmd 1)
* FDAM (FDAmd) – Final Draft Amendment (e.g., ISO/IEC 13818-1:2007/FDAmd 4)
* Amd – Amendment (e.g., ISO/IEC 13818-1:2007/Amd 1:2007)

Other abbreviations:

* TR – Technical Report (e.g., ISO/IEC TR 13818-5:2005)
* TS – Technical Specification
* IWA – International Workshop Agreement
* Cor – Technical Corrigendum (e.g., ISO/IEC 13818-1:2007/Cor 1:2008)

A proposal of work (New Proposal) is approved at Subcommittee and then at the Technical Committee level (SC29 and JTC1 respectively – in the case of MPEG). When the scope of new work is sufficiently clarified, MPEG usually makes open requests for proposals – known as "Call for proposals". The first document that is produced for audio and video coding standards is called a Verification Model (VM). In the case of MPEG-1 and MPEG-2 this was called Simulation and Test Model, respectively. When a sufficient confidence in the stability of the standard under development is reached, a Working Draft (WD) is produced. This is in the form of a standard but is kept internal to MPEG for revision. When a WD is sufficiently solid, becomes Committee Draft (CD) (usually at the planned time). It is then sent to National Bodies (NB) for ballot. The CD becomes Final Committee Draft (FCD) if the number of positive votes is above the quorum. After a review and comments issued by NBs, FCD is again submitted to NBs for the second ballot. If the FCD is approved, it becomes Final Draft International Standard (FDIS). ISO then holds a ballot with National Bodies, where no technical changes are allowed (yes/no ballot). If approved, the document becomes International Standard (IS).[[3]](https://en.wikipedia.org/wiki/Moving_Picture_Experts_Group#cite_note-about-mpeg-3)

ISO/IEC Directives allow also the so-called "Fast-track procedure". In this procedure a document is submitted directly for approval as a draft International Standard (DIS) to the ISO member bodies or as a final draft International Standard (FDIS) if the document was developed by an international standardizing body recognized by the ISO Council.

**BIT STREAM SYNTAX PERFOMANCE**

A **bitstream format** is the format of the data found in a [stream](https://en.wikipedia.org/wiki/Stream_(computing)) of bits used in a [digital communication](https://en.wikipedia.org/wiki/Digital_communication) or [data storage](https://en.wikipedia.org/wiki/Data_storage_device) application. The term typically refers to the data format of the output of an encoder, or the data format of the input to a decoder when using [data compression](https://en.wikipedia.org/wiki/Data_compression).

## Processing

Standardized [interoperability](https://en.wikipedia.org/wiki/Interoperability) specifications such as the [video coding](https://en.wikipedia.org/wiki/Video_coding) standards produced by the [MPEG](https://en.wikipedia.org/wiki/MPEG) and the [ITU-T](https://en.wikipedia.org/wiki/ITU-T), and the audio coding standards produced by the MPEG, often specify only the [bitstream](https://en.wikipedia.org/wiki/Bitstream) format and the decoding process. This allows encoder implementations to use any methods whatsoever that produce bitstreams which conform to the specified bitstream format.[[*clarification needed*](https://en.wikipedia.org/wiki/Wikipedia:Please_clarify)]

Normally, decoding of a bitstream can be initiated without having to start from the beginning of a [file](https://en.wikipedia.org/wiki/Computer_file), or the beginning of the data transmission. Some bitstreams are designed for this to occur, for example by using indexes or [key frames](https://en.wikipedia.org/wiki/Key_frame).[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

Uses of *bit stream decoders* (BSD):

* [Graphics processing unit](https://en.wikipedia.org/wiki/Graphics_processing_unit#SP-GPGPU) (GPU)
* [H.264/MPEG-4 AVC](https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC#HW-BASED)
* [Unified Video Decoder](https://en.wikipedia.org/wiki/Unified_Video_Decoder) (UVD) – the video decoding bit-stream technology from ATI Technologies/AMD
* [PureVideo](https://en.wikipedia.org/wiki/Nvidia_PureVideo) – the video decoding bit-stream technology from Nvidia
* [Quick Sync Video](https://en.wikipedia.org/wiki/Intel_Quick_Sync_Video) – the video decoding and encoding bit-stream technology from Intel

### [Understanding the Difference Between MPEG-2 & MPEG-4 Encoders](https://www.rccfiber.com/understanding-difference-mpeg-2-mpeg-4-encoders/)

In a world where technology is rapidly expanding the way that we send, receive, compress, and decompress data, it can be difficult to maintain a full understanding of formats and processes. One of the most common sources of confusion when it comes to video encoding arises when trying to understand the difference between MPEG-2 and MPEG-4. Thankfully, our team at Radiant Communications Corporation, one of the best places to [buy MPEG-2 encoders online](https://www.rccfiber.com/), is well-versed on the subject. As a way of bringing clarity to this difference, we will explain how MPEG-2 and MPEG-4 encoders handle compression, file size, quality, and application.

**MPEG-2**

MPEG-2 encoders have been around since the late 1990s and are responsible for great strides in the capabilities of video encoding as compared to their predecessor, MPEG-1. MPEG-2 video coding uses an algorithm for compression that uses a block-based 8 x 8 discrete cosine transform (DCT) and allows for high-quality, but restricts the ability to shrink file size, which has become integral in the world of digital multimedia. Due to the larger file sizes associated with MPEG-2 encoders, they are typically used with local video such as DVDs and broadcast cable. While the quality of MPEG-2 is more than capable of handling high-definition local video, there is a significant degradation in quality when applying MPEG-2 to online and device-based streaming.

**MPEG-4**

After passing over MPEG-3, the MPEG-4 video coding format was released to coincide with the massive expansion of the internet that occurred in the first few years of the 21st century. One of the most notable differences between the MPEG-2 and MPEG-4 formats involves the use of an algorithm which has a 16 x 16 DCT, allowing for much a much higher rate of compression. This higher compression rate means that files sizes can be cut in half, opening the door for decreased bandwidth without losing picture quality. MPEG-4 is a step in the natural progression of encoding methods, as the internet has created an environment where online and mobile streaming are the primary forms of video consumption.

**The Basics**

Ultimately, both MPEG-2 and MPEG-4 are capable of maintaining high-definition video quality. The difference, however, lies within the application. MPEG-2 is more than capable of handling video streams from local sources like DVDs and cable broadcasts, but due to the larger file size, it struggles with portable devices and internet streaming. MPEG-4, on the other hand, utilizes its high rate of compression and smaller file sizes to provide high-quality video and audio across mobile networks.