Chapter 5

Data Compression

5.1 Data Compression and Coding Fundamentals

Data compression is the process of converting an input data stream or the source stream or the original raw data into another data stream that has a smaller size. For example, text compression, image compression, audio compression and video compression.

There are two types of data compression:

1. *Lossy Compression*
2. *Lossless Compression*

Lossy Compression

Lossy compression algorithms are normally not to reproduce an exact copy of the source information after decompression but rather a version of its which is perceived by the recipient as a true copy.

In lossy compression some information is lost during the processing, where the image data is stored into important and unimportant data. The system then discards the unimportant data.

It provides much higher compression rates but there will be some loss of information compared to the original source file. The main advantage is that the loss cannot be visible to eye or it is visually lossless. Visually lossless compression is based on knowledge about color images and human perception.

Lossless Compression

In this type of compression, no information is lost during the compression and the decompression process. Here the reconstructed image is mathematically and visually identical to the original one. It achieves only about a 2:1 compression ratio.

This type of compression technique looks for patterns in strings of bits and then expresses them more concisely.

Lossless compression algorithm the aim is to be transmitted in such a way that, when the compressed information is decompressed, therefore is no loss information.

Techniques of Data Compression

There are three important techniques of data compression:

1. *Basic Technique*
2. *Statistical Technique*
3. *Dictionary Method*

Basic Technique

These are the techniques, which have been used only in the past. The important basic techniques are *run length encoding* and *move to front encoding*.

Run Length Encoding:

The basic idea behind this approach to data compression is this: if a data item occurs n consecutive times in the input stream replace the n occurrences with a single pair. The n consecutive occurrences of a data item are called run length of n and this approach is called run length encoding or RLE.

Move to Front Coding:

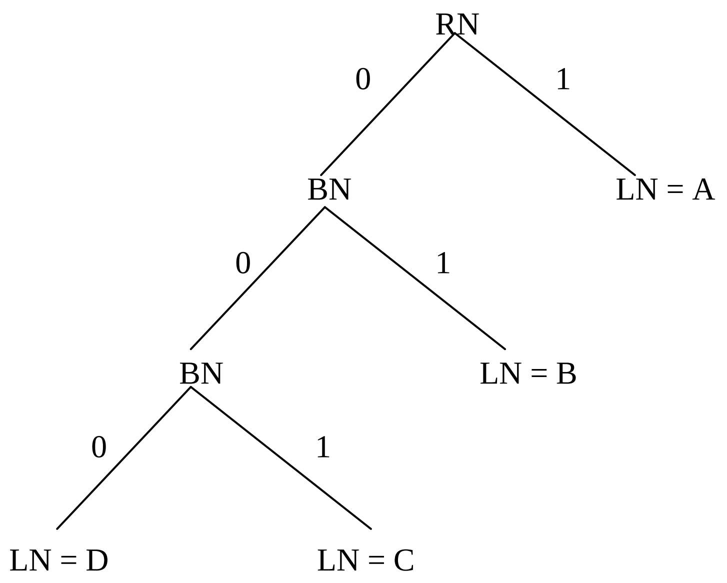
The basic idea of this method is to maintain the alphabet A of symbols as a list where frequently occurring symbols are located near the front. A symbol ‘a’ is encoded as the number of symbols that precede it in this list. Thus, if A = (‘t’, ‘h’, ‘e’, ‘s’) and the next symbol in the input stream to be encoded is ‘e’, it will be encoded as ‘2’ since it is preceded by two symbols. The next step is that after encoding ‘e’ the alphabet is modified to A = (‘e’, ‘t’, ‘h’, ‘s’). This move to front step reflects the hope that once ‘e’ has been read from the input stream it will read many more times and will at least for a while be a common symbol.

Huffman Coding

A commonly used method for data compression is Huffman coding. The method starts by building a list of all the alphabet symbols in descending order of their probabilities. It then constructs a tree with a symbol at every leaf from the bottom up.

The Character string to be transmitted is first analyzed and the character types and their relative frequency determined.

A Huffman (code) tree is a binary tree with branches assigned the value 0 or 1. As each branch divides, a binary value of 0 or 1 is assigned to each new branch: a binary 0 for the left branch and a binary 1 for the right branch.



Where RN = Root Node, BN = Branch Node, LN = Leaf Node

A = 1, B = 01, C = 001, D = 000

Figure 5.1: Huffman Tree

Arithmetic Coding

In this method the input stream is read symbol and appends more to the code each same time a symbol is input and processed. To understand this method, it is useful to imagine resulting code as a number in the range [0,1] that is the range of real numbers from 0 to 1 not including one. The first step is to calculate or at least to estimate the frequency of occurrence of each symbol.

LZ77 (Sliding Window)

LZ77 and LZ78 are the names for the two lossless data compression algorithms published in papers by Abraham Lempel and Jacob Ziv in 1977 and 1978. They are also known as LZ1 and LZ2 respectively. They are both dictionary coders. LZ77 is the Sliding Window compression algorithm.

The main idea of this method is to use part previously seen input stream as the dictionary. The encoder maintains a window to the input stream and shifts the input in that window from right to left as strings of symbols are being encoded. The method is thus based on "Sliding Window". The window is divided into two parts that is search buffer, which is the current dictionary and lookahead buffer, containing text yet to be encoded.

Source, Entropy and Hybrid Coding

Entropy Coding

- lossless encoding
- used regardless of media's specific characteristics
- data taken as a simple digital sequence
- decompression process regenerates data completely
- e.g. run-length coding, Huffman coding, Arithmetic coding *Source Coding*
- lossy encoding
- takes into account the semantics of the data
- degree of compression depends on data content.
- E.g. content prediction technique - DPCM, delta modulation *Hybrid Coding*

(used by most multimedia systems)

- combine entropy with source encoding
- E.g. JPEG, H.263, DVI (RTV & PLV), MPEG-1, MPEG-2, MPEG-4

Major Steps of Data Compression

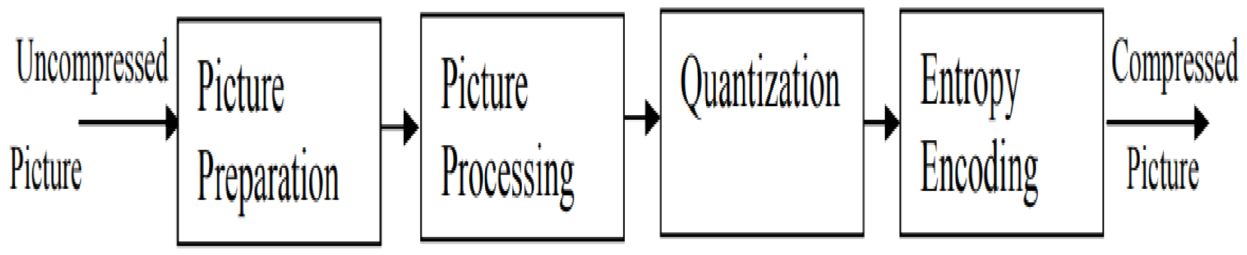


Figure 5.2: Major Steps of Data Compression

Picture Preparation:

Preparation includes analog to digital conversion and generating an appropriate digital representation of the information. An image is divided into blocks of 8x8 pixels, and represented by a fixed number of bits per pixel.

Picture Processing:

Processing is actually the first step of the compression process which makes use of sophisticated algorithms. A transformation from the time to the frequency domain can be performed using DCT. In the case of motion video compression, interframe coding uses a motion vector for each 8x8 blocks. Motion video computation for digital video.

Quantization:

Quantization processes the results of the previous step. It specifies the granularity of the mapping of real numbers into integers. This process results in a reduction of precision. For example, they could be quantized using a different number of bits per coefficient. For example, 12 bits for real values, 8 bits for integer value.

Entropy Encoding:

Entropy encoding is usually the last step. It compresses a sequential digit data stream without loss. For example, a sequence of zeroes in a data stream can be compressed by specifying the number of occurrences followed by the zero itself.

JPEG (Joint Photographic Expert Group)

JPEG is a commonly used method of lossy compression for digital photography (image).

The JPEG lossy compression scheme is one of the most popular and versatile compression schemes in widespread use. It's ability to attain considerable size reductions with minimal visual impact with relative light computational requirements and the ability to fine tune the compression level to suit the image at hand has made it the standard for continuous tone still images.

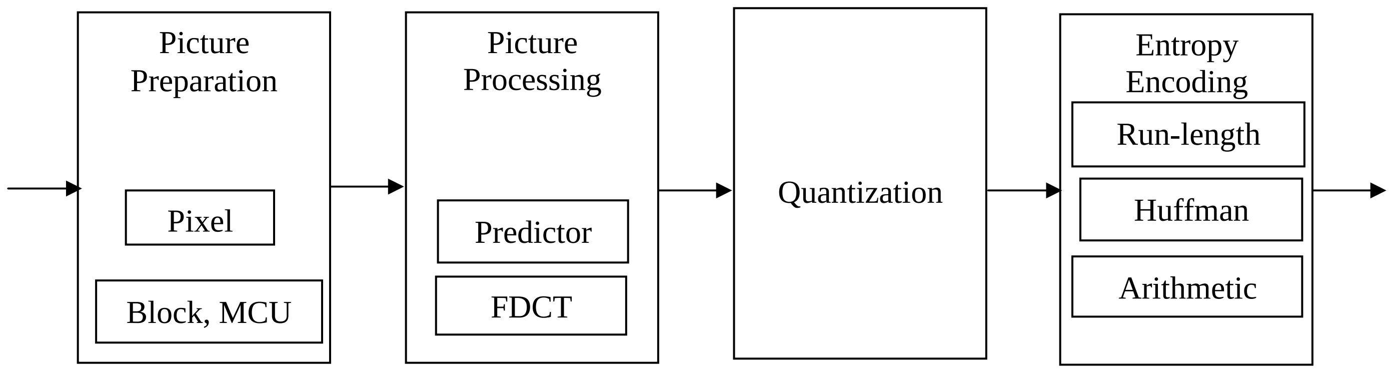


Figure 5.3: Steps of the JPEG compression process.

Requirements on JPEG implementations

JPEG Image Preparation

- Blocks, Minimum Coded Units (MCU) *JPEG*

Image Processing

- Discrete Cosine Transformation (DCT) *JPEG*

Quantization

- Quantization Tables *JPEG Entropy Encoding*
- Run-length Coding/Huffman Encoding

Additional Requirements - JPEG

- JPEG implementation is independent of image size and applicable to any image and pixel aspect ratio.
- Image content may be of any complexity (with any statistical characteristics).
- JPEG should achieve very good compression ratio and good quality image.
- From the processing complexity of a software solution point of view: JPEG should run on as many available platforms as possible.

- Sequential decoding (line-by-line) and progressive decoding (refinement of the whole image) should be possible.

Variants of Image Compression

Four different modes

Lossy Sequential DCT based mode

- Baseline process that must be supported by every JPEG implementation. *Expanded Lossy DCT based mode*
- enhancements to baseline process *Lossless mode*
- low compression ratio
- allows perfect reconstruction of original image

Hierarchical mode

- accommodates images of different resolutions

Image Preparation

The image preparation model in JPEG is very general. It
is not based on

- 9-bit YUV encoding
- fixed number of lines, columns
- mapping of encoded chrominance

It is independent from image Parameters such as *image size*, *image* and *pixel ratio*.

Source image consists of 1 to 255 components (planes).

- For example, each component Ci (1≤ i ≤ 255) may be assigned to YUV, RGB or YIQ signals.

Each pixel is presented by ‘p’ bits, value is in the range of (2^p -1).

All pixels of all components within the same image are coded with the same number of bits.

- Lossy modes of JPEG use precision of 8-12 bits per pixel.
- Lossless mode uses precision of 2 up to 12 bits per pixel.

- If JPEG application makes use of any other number of bits, then application must perform a suitable image transformation to the well-defined number of bits/pixel (JPEG standard).

Images are divided into data units, called blocks

- Lossy modes operate on blocks of 8X8 pixels.
- Lossless modes operate on data units equal to 1 pixel.
- DCT transformation operates on blocks.

Data units are processed component by component and passed to image processing. Processing of data units per component can be

- Non-interleaved data ordering
- Left to right, top to bottom
- Interleaved data ordering

Interleaved Data units of different components are combined to minimum coded units (MCUs).

If image has the same resolution, then MCU consists of exactly one data unit for each component.

Decoder displays the image MCU by MCU.

If image has different resolution for single components, then reconstruction of MCUs is more complex.

For each component, determine regions of the data units. Each component has same number of regions, MCU corresponds to one region. Data units in a region are ordered left-right, top-bottom

Build MCU

JPEG standard - only 4 components can be encoded in interleaved mode

Bound on length of MCU

MCU consists of at most 10 data units.

-

After image preparation, uncompressed image samples are grouped into data units of 8x8 pixels and passed to the JPEG encoder

order of the data units is defined by the MCUs

- precision 8 bits/pixel represents the baseline mode
- values are in the range of [0,255];

Image Processing

First step:

- Pixel values are shifted (ZERO-SHIFT) into the range [-128,127] with 0 in the center.

  Values in the 8x8 pixel are defined by S_yx with y,x in the range [0,7] and there are 64 sampled values S_yx in each block.

- DCT maps values from time to frequency domain.

  1D Forward Discrete Cosine Transformation

$$S(u) = \frac{C(u)}{2}\sum_{x=0}^{7} S(x)\cos\left(\frac{(2x+1)u\pi}{16}\right)$$

$S(x) - 1D\ sampled\ value,$

$C(u) - scaling\ coefficient,$

$S(u) - 1D\ DCT\ coefficient\ (transforrms\ S(x) into\ frequency\ domain)$

2D Forward Discrete Cosine Transformation

•

S(v,u)

$$S(u,v)=\frac{1}{4}C(u)C(v)\sum_{x=0}^{7}\sum_{y=0}^{7}S(y,x)\cos\left(\frac{(2x+1)u\pi}{16}\right)\cos\left(\frac{(2x+1)v\pi}{16}\right)$$

$C(u),C(v)={}^{1}/_{\sqrt{2}}$ $for\ u,v=0, C(u),C(v)=1\ otherwise$

$S(y,x)-2D\ sampled\ values$

$C(u),C(v)-scaling\ coefficients$

$S(v,u)-2D\ DCT\ coefficients$

coefficients:

- S(0,0) includes the lowest frequency in both directions and it is called the DC coefficient. S(0,0) determines the fundamental color of the BLOCK(64 pixels). For this coefficient the frequency is equal 0 in both directions.
- S(0,1)…S(7,7) are called AC coefficients. Their frequency is non-zero in one or both directions. There exist many AC coefficients with a value around 0.

Factoring

- By computing the DCT coefficients, we can use factoring; the problem will be reduced to a series of 1D FDCTs.

$$S(v,u) = \frac{1}{4}\sum_{x=0}^{7} C(u)\cos\left(\frac{(2x+1)u\pi}{16}\right)\left(\sum_{y=0}^{7} C(v)\cos\left(\frac{(2y+1)v\pi}{16}\right)S(y,x)\right)$$

Quantization

GOAL: To throw out bits.

Example:

- 101101 = 45 (6 bits).
- We can truncate this to 4 bits: 1011 - 11
- or 3 bits 101 = 5 (original value - 40) or 110 = 6 (value = 48)

Uniform quantization is achieved by dividing the DCT coefficient value S(v,u) by N and rounding the result.

In S(v,u) how many bits do we throw away?

ANSWER: *Use quantization tables*

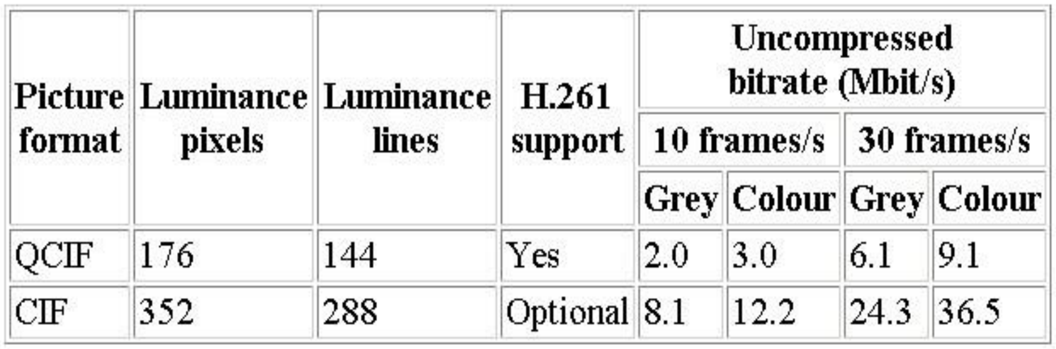
Entropy Encoding

- After image processing we have quantized DC and AC coefficients.
- Initial step of entropy encoding is to map 8x8 plane into 64 element vectors

H.261

- ❖ H.261 is a video coding standard published by the ITU-T in 1990.
- ❖ It is the most widely used international video compression standard for video coding.
- ❖ H.261 is usually used in conjunction with other control and framing standards.
- ❖ The H.261 standard describes the video coding/decoding methods for the video portion of an audiovisual service
- ❖ Designed for data rates of p*64 kbps, where p is in the range 1-30
- ❖ Targeted for circuit-switched networks (ISDN was the communication channel considered within the framework of the standard)

- Defines two picture formats:
    - CIF (352*288) and QCIF(176*144)



Picture Formats Supported

Picture format	Luminance pixels	Luminance lines	H.261 support	Uncompressed bitrate (Mbit/s)			
				10 frames/s		30 frames/s	
				Grey	Colour	Grey	Colour
QCIF	176	144	Yes	2.0	3.0	6.1	9.1
CIF	352	288	Optional	8.1	12.2	24.3	36.5

- The H.621 encoding algorithm is a combination of:
    - inter-picture prediction (to remove temporal redundancy)
    - transform coding (to remove spatial redundancy)
    - motion vectors (for motion compensation)
- The three main elements are:

❖ Prediction: blocks are intra- or inter-coded
  - Intra-coded blocks stand alone
  - Inter-coded blocks are based on differences between the previous frame and the current one
- Block Transformation:
    - each block (inter and intra) is composed into 8x8 blocks and processed by a 2-D DCT function
- Quantization & Entropy Coding:
    - achieves further compression by representing DCT coefficients with only the necessary precision

      entropy encoding (non-lossy) using Huffman encoding
- Coding algorithm is a hybrid of:
    - Inter-picture prediction - removes temporal redundancy
    - Transform coding - removes the spatial redundancy
    - Motion compensation – uses motion vectors to help the codec compensate for motion
- Data rate can be set between 40 Kbit/s and 2 Mbit/s

- Input signal format
    - CIF (Common Intermediate Format) and QCIF (Quarter CIF) available
- Bit rate
    - The target bit rate is ~ 64Kbps to 1920Kbps

MPEG (Motion/Moving Pictures Expert Group)

Motion Pictures Expert Group (MPEG) standards are digital video encoding processes that coordinate the transmission of multiple forms of media (multimedia). MPEG is a working committee that defines and develops industry standards for digital video systems. These standards specify the data compression and decompression process and how they are delivered on digital broadcast systems. MPEG is a part of International Standards Organization (ISO).

Started in 1988, the MPEG project was developed by a group of hundreds of experts under the auspices of the ISO (International Standardization Organization) and the IEC (International Electrotechnical Committee). The name MPEG is an acronym for Moving Pictures Experts Group. MPEG is a method for video compression, which involves the compression of digital images and sound, as well as synchronization of the two. There currently are several MPEG standards. MPEG-1 is intended for intermediate data rates, on the order of 1.5 Mbit/s. MPEG-2 is intended for high data rates of at least 10 Mbit/s. MPEG-3 was intended for HDTV compression but was found to be redundant and was merged with MPEG-2. MPEG-4 is intended for very low data rates of less than 64 Kbit/s. A third international body, the ITU-T, has been involved in the design of both MPEG-2 and MPEG-4. This section concentrates on MPEG-1 and discusses only its image compression features.

MPEG standard consists of both video and audio compression. MPEG standard includes also many technical specifications such as image resolution, video and audio synchronization, multiplexing of the data packets, network protocol, and so on. Here we consider only the video compression in the algorithmic level. The MPEG algorithm relies on two basic techniques

- Block based *motion compensation*
- DCT based compression

MPEG itself does not specify the encoder at all, but only the structure of the decoder, and what kind of bit stream the encoder should produce. Temporal prediction techniques with motion compensation are used to exploit the strong temporal correlation of video signals. The motion is estimated by predicting the current frame on the basis of certain previous and/or forward frame. The information sent to the decoder consists of the compressed DCT coefficients of the residual block together with the *motion vector*. There are three types of pictures in MPEG:

- Intra-pictures (*I*)
- Predicted pictures (*P*)
- Bidirectionally predicted pictures (*B*)

Figure 1 demonstrates the position of the different types of pictures. Every *N*th frame in the video sequence is an *I*-picture, and every *M*th frame a *P*-picture. Here *N*=12 and *M*=4.
The rest of the frames are *B*-pictures.

Compression of the picture types:

Intra pictures are coded as still images by DCT algorithm similarly than in JPEG. They provide access points for random access, but only with moderate compression. *Predicted pictures* are coded with reference to a past picture. The current frame is predicted on the basis of the previous *I*- or *P*-picture. The residual (difference between the prediction and the original picture) is then compressed by DCT. Bidirectional pictures are similarly coded than the *P*-pictures, but the prediction can be made both to a past and a future frame which can be *I*- or *P*-pictures. Bidirectional pictures are never used as reference.

The pictures are divided into 16□16 macroblocks, each consisting of four 8□8 elementary blocks. The *B*-pictures are not always coded by bidirectional prediction, but four different prediction techniques can be used:

- Bidirectional prediction
- Forward prediction • Backward prediction
- Intra coding.

The choice of the prediction method is chosen for each macroblock separately. The bidirectional prediction is used whenever possible. However, in the case of sudden camera movements, or a breaking point of the video sequence, the best predictor can sometimes be given by the forward predictor (if the current frame is before the breaking point), or backward prediction (if the current frame is after the breaking point). The one that gives the best match is chosen. If none of the predictors is good enough, the macroblock is coded by intra-coding. Thus, the *B*-pictures can consist of macroblock coded like the *I*-, and *P*-pictures.

The intra-coded blocks are quantized differently from the predicted blocks. This is because intra-coded blocks contain information in all frequencies and are very likely to produce 'blocking effect' if quantized too coarsely. The predicted blocks, on the other hand, contain mostly high frequencies and can be quantized with more coarse quantization tables.

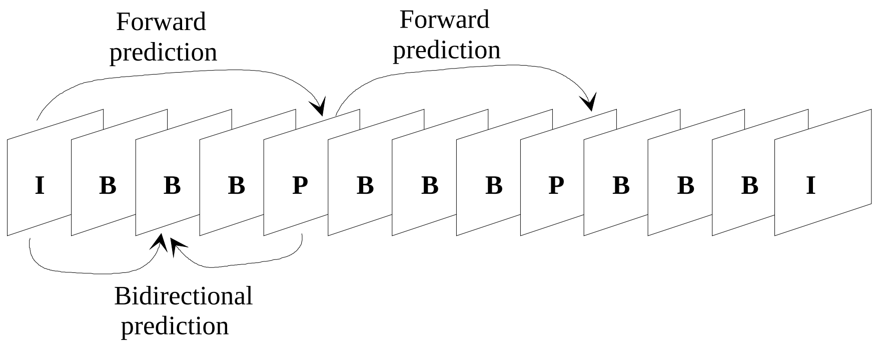


Figure 5.4: Interframe coding in MPEG.

Motion estimation:

The prediction block in the reference frame is not necessarily in the same coordinates than the block in the current frame. Because of motion in the image sequence, the most suitable predictor for the current block may exist anywhere in the reference frame. The *motion estimation* specifies where the best prediction (best match) is found, whereas *motion compensation* merely consists of calculating the difference between the reference and the current block.

The motion information consists of one vector for forward predicted and backward predicted macroblocks, and of two vectors for bidirectionally predicted macroblocks. The MPEG standard does not specify how the motion vectors are to be computed, however, *block matching techniques* are widely used. The idea is to find in the reference frame a similar macroblock to the macroblock in the current frame (within a predefined search range). The candidate blocks in the reference frame are compared to the current one. The one minimizing a cost function measuring the mismatch between the blocks, is the one which is chosen as reference block.

Exhaustive search where all the possible motion vectors are considered are known to give good results. Because the full searches with a large search range have such a high computational cost, alternatives such as *telescopic* and *hierarchical searches* have been investigated. In the former one, the result of motion estimation at a previous time is used as a starting point for refinement at the current time, thus allowing relatively narrow searches even for large motion vectors. In hierarchical searches, a lower resolution representation of the image sequence is formed by filtering and subsampling. At a reduced resolution the computational complexity is greatly reduced, and the result of the lower resolution search can be used as a starting point for reduced search range conclusion at full resolution.

MPEG in the Real-world

MPEG has found a number of applications in the real world, including:

- *Direct Broadcast Satellite.* MPEG video streams are received by a dish/decoder, which unpacks the data and synthesizes a standard NTSC television signal.
- *Cable Television.* Trial systems are sending MPEG-II programming over cable television lines.

- *Media Vaults.* Silicon Graphics, Storage Tech, and other vendors are producing on-demand video systems, with twenty file thousand MPEG-encoded films on a single installation.
- *Real-Time Encoding.* This is still the exclusive province of professionals. Incorporating special-purpose parallel hardware, real-time encoders can cost twenty to fifty thousand dollars.

Comparison between H.261 Vs MPEG

H.261	**MPEG**
■ based on JPEG	■ based on H.261 and JPEG
■ encodes video only	■ encodes audio & video
■ lossy algorithm with compression in space and time	■ lossy algorithm with compression in space and time
■ uses I and P-frames	■ uses I, P, and B-frames
■ uses DCT on 8x8 blocks	■ uses DCT on 8x8 blocks
■ best for video with little motion (eg. video conferencing)	■ designed to handle moving picture components
■ optimized for bandwidth efficiency and low delay	■ less bandwidth efficient

Differences between H.261 and MPEG

H.261	**MPEG**
■ Only 1, 2 or 3 skipped pictures allowed.	■ No restrictions on skipped pictures.
■ Pixel accurate motion vectors.	■ Sub-pixel accurate motion vectors.
■ Typical motion vector range is +/-7 pixels.	■ Typical motion vector range is +/-15 pixels
■ Used mostly in interactive applications. End-to-end delay is very critical.	■ The end-to-end coding delay is not critical.

DVI (Digital Video Interactive)

Digital Video Interactive (DVI) is a technology that includes coding algorithms. The fundamental components are a VLSI (Very Large-Scale Integration) chip set for the video subsystem, a well-specified data format for audio and video files, an application user interface to the audio-visual kernel (Audio Video Kernel, the kernel software interface to the DVI hardware) and compression, as well as decompression, algorithms.

DVI can process data, text, graphics, still images, video and audio. The original essential characteristic was the asymmetric technique of video compression and decompression known as Presentation-Level Video (PLV).

Concerning audio, the demand for a hardware solution that can be implemented at a reasonable price is met by using a standard signal processor. Processing of still images and video is performed by a video processor. The video hardware of a DVI board is shown in Figure. It consists of two VLSI chips containing more than 265,000 transistors each. This Video Display Processor (VDP) consists of the pixel processor VDP1 and the display processor VDP2. VDP1 processes bitmaps and in programmed in microcode; VDP2 generates analog RGB signals out of the different bitmap formats and its configuration is also programmable. The coupling of the processors is carried out by the Video-RAM (VRAM). An important characteristic is the capability for microprogramming. It allows one to change and adapt the compression and decompression algorithms to new developments without investing in new hardware.

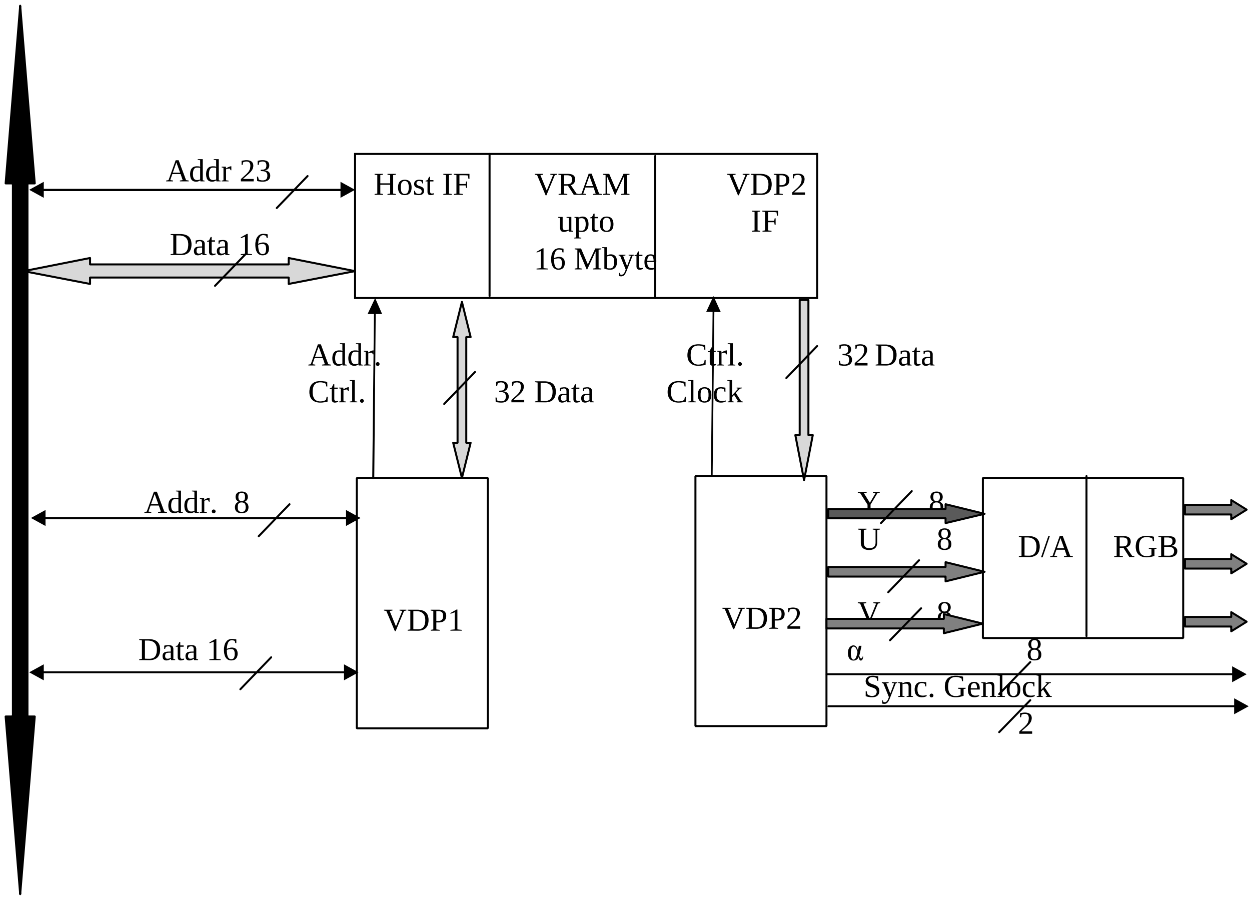


Figure 5.5: DVI Video Processing