11.3. MPEG

MPEG is a compression standard for digital video sequences, such as used in computers and video, and digital television network. In addition, MPEG also provide for the compression of the sound track associated with the video. The name come from its originating organization, the moving pictures expert Group. In additional to reducing the data rate, MPEG has several important features. The movie can be played forward or in reverse, and at either normal or fast speed. The encoded information is random access, that is any individual frame in the sequence can be easily displayed as a still picture. This goes along with movie editable, meaning that short segments from the movie can be encoded only with reference to themselves, not the entire sequence. MPEG is designed to be robust to errors.

The approach used by MPEG can be divided into two types of compression: Within the frame and between frame. Within the frame compression means that individual frame making up the video sequence are encoded as if they are still images. In MPEG, the frame that has been encoded in this way is called inter coded or I-picture. Most of the pixels in a vide sequence changes very litter from one frame to the next. Unless the camera is moving, most of the image is composed of a background that remain constant over degree of frame MPEG takes advantage of this with a sophisticated form of delta encoding is compress the redundant information between frames. After compression one of the frames as an I-picture, MPEG encoders succession frames as prediction. Coded or P-picture. That is only the pixels that have changed since the I-picture are undivided in P-picture. The main distortion associated with MPEG occurs when large section of the image change quickly. If the data rate is fixed, the viewer notice "blocky" patterns when changing from one scene to the next. This can be minimized in the networks that transmit multiple video channels simultaneously, such as cable television.

11.3.1. MPEG-1 Audio layer III (MP3)

It is difficult to carry music waveforms across the internet because of lengthy transmission times. The MPEG layer 3 data reduction algorithm is widely used to compress files prior to electronic distribution. The layer 3 algae is widely known as MP3. The MP3 CODEC like other MPEG compression algo. Is lossy in nature. This means that it discard data from original uncompressd file to reduce its file size. The crucial point, is which data to keep and which data to discard, under certain circumstances specific sounds may not be avoidable to human ear, a phenomenon is called, masking, MP3 algo attempts to analyze the input audio in the frequency domain and identify the sounds which are masked within each critical band. The masked sounds are subsequently removed and the remaining data from the final compressed file.

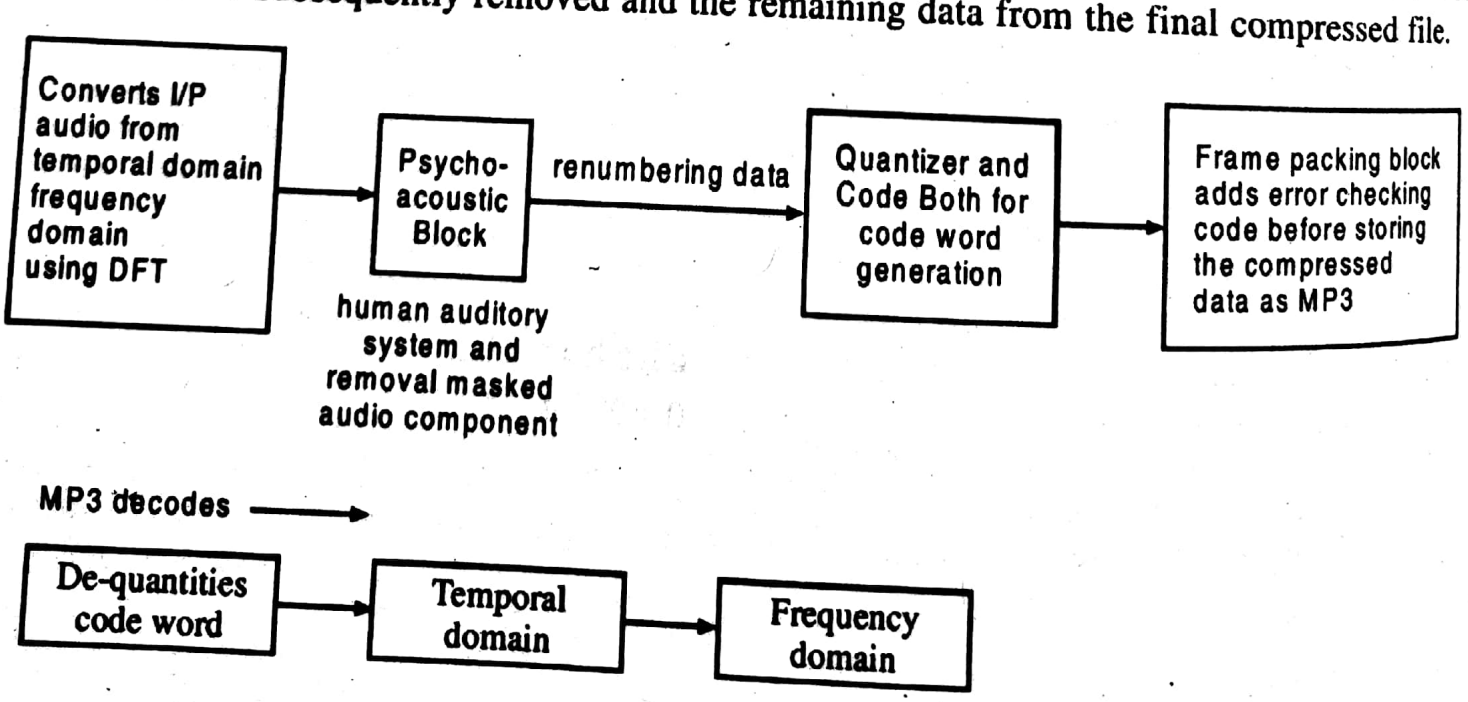

Fig. 11.4.

11.3.2. MPEG-1 Video

The MPEG-1 video standard was completed in 1991 with the development of the ISO/IEC specification 11172 which the standard for coding of moving pictures and associated audio for digital storage at about 1.5 Mbps. Important features provided by MPEG-1 include frame-based random access of video, fast/ fast-reverse searches, reverse playback of video, and editability of the compressed bitstream. To achieve a high compression ratio, both intra-frame and inter-frame redundancy are exploited.

MPEG-I Video uses three different types of frames:

1. **I-frame (Intracoded frame):** These are the codes without any reference to other images. MPEG make use of IPEG coding for i-frame. They can be used as reference for other frames. In video sequence I-frame are present at regular intervals in order to allow for the possibility of the contents of an I-frame being corrupted during transmission. The collection of frames between succession I-frame is known as group of picture. Typical range of frames in GoP is 3 to 12. Compression ratio 10:1 to 20:1.
2. **P-frame prediction frame:** They require information from previous I and/or P-frame for encoding and decoding. P frame can be accused only after referenced I or P frame has been decoded. The number of P frame in Group of picture is generally limited as error in one frame may propagated in others. The number of frames between a P-frame and preceding I or P frame is called prediction span may range from 1 to 3 ratio compression 20:1 to 30:1.
3. **B-frames (Bidirectional Predictive):** Requires information from the previous and following I and / or P frame for encoding and decoding. The highest compression ratio is attainable by using these frames. B frames are never used as reference for other frames. Reference frames must be transmitted first. Thus transmission order and display order differ. The I frame must be transmitted first followed by the next P frame and B frames must be transmitted. Typical compression ratio range 30:1 to 50:1.

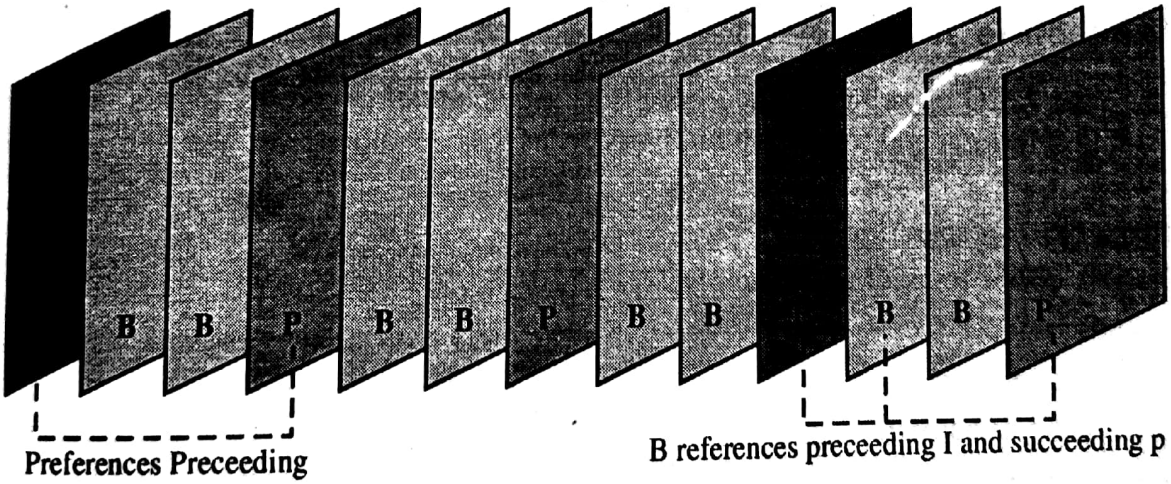

Fig. 11.5.

4. **PB-frames:** A fourth of frame known as PB frame has been defined. This refers to the encoding to two neighbouring. P and B frames as a single frames. This enables increasing the frame rate without significantly increasing the resulting bit rate.
5. **D-frames:** A fifth type of frame known as D-frame is used in VoD applications. These are used to displayed the content when the user rewinds or fast-forwards through the movie and are ignored during the normal decoding of frames. The D-frames are highly compressed and inserted at regular intervals throughout the stream. They consists of only the encoded DC coefficients during the DCT phase and hence, are necessarily of low resolution which can be decoded at higher speeds.

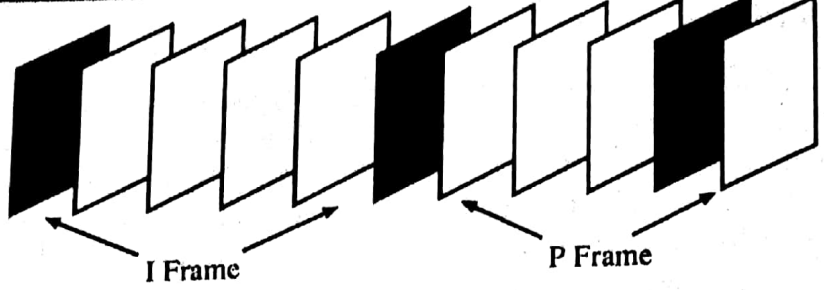

Fig. 11.6.

A complete video stream is known as a sequence, which consists of a string of GOP, GOP comprises of a string of I, P and B frames. Each frame is made of a number of macroblocks, and each macroblock is made of a number of 8 by 8 pixel block.

11.4. JPEG STANDARDS

The JPEG standards was developed by the International Standards Organization (JPEG). Broad goal of JPEG was to develop a general purpose still image compression standard that would be applicable to virtually all continuous-tone still image, for its transmission and storage problems. Its purpose was to

1. Achieve rate and reconstructed image quality "at or near the state of art" with image fidelity classifiable ranging from "very good" to "excellent".
2. Be useful for compression almost on any continuous-tone image, including both gray-scale and color, any color space and most image sizes.
3. Have complexity that would allow software implementation on many common computing platform and affordable hardware implementations.

Base mode of JPEG compression standard is explained in the next section.

11.4.1. Baseline JPEG

The baseline JPEG encoding method is called the *sequential* encoding. In this mode, the image is scanned from left to right and top to bottom. It produces lossy compression. Steps involved in this mode are:

1. Image block preparation.
2. Discrete cosine transform (forward DCT).
3. Quantization.
4. Zig-zag scanning.
5. Run-length encoding.
6. Statistical encoding.

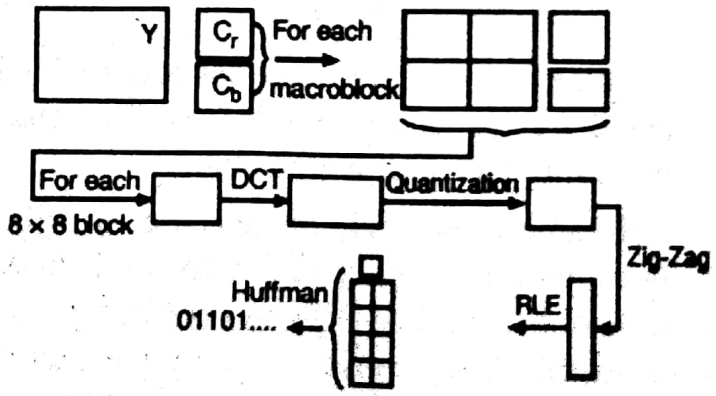

Fig. 11.7.

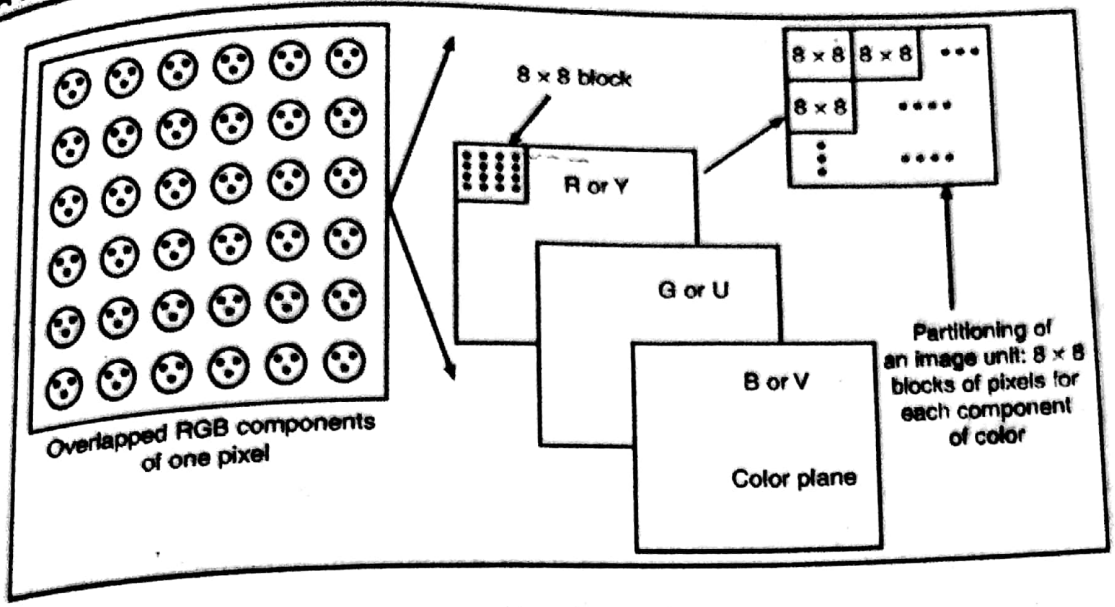

Fig. 11.8.

11.4.2. Image Block Preparation

Pixel matrix of the source image is represented by the three colour components. JPEG compression echnique works on each colour component independently, as shown in figure, resulting in three matrices f pixels corresponding to each colour component.

Each colour plane is divided into blocks of 8 × 8 pixels.

Example: An image with pixel matrix of 640 × 480 will have 14,400 blocks of 8 × 8 each.

$$\left[\frac{640}{8} \times \frac{480}{8}\right] = 80 \times 60 = 4800 \text{ blocks/colour component}$$

Therefore, for three components,

Total blocks ($\Rightarrow$) $4800 \times 3 = 14{,}400$ blocks

1.4.3. Discrete Cosine Transform

The forward DCT converts pixel amplitudes in the spatial domain into DCT coefficients in the equency domain. The forward DCT is computed on each block, resulting in 64 DCT coefficients. The plitude variations in the spatial domain are smooth variations, which is the case for continuous-tone ages. JPEG is aimed at continuous-tone images, the high frequency components can be given less portance.

Forward DCT formula is given as

$$F(u,v) = \frac{2c(u)\,c(v)}{\sqrt{MN}} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \cos\frac{(2i+1)}{2M} \cos\frac{(2j+1)}{2N} f(i,j)$$

ere

$i, u = 0, 1, 2, \ldots, M-1$

$j, v = 0, 1, 2, \ldots, N-1$

$$c(u) = \begin{cases} \frac{1}{\sqrt{2}}; & \text{if } u = 0 \\ 1; & \text{otherwise} \end{cases}$$

$$c(v) = \begin{cases} \frac{1}{\sqrt{2}}; & \text{if } v = 0 \\ 1; & \text{otherwise} \end{cases}$$

Since we have partitioned the image in terms of 8 × 8 blocks, in the above equation, we have $N = M = 8$.

11.4.4. Quantization

The quantizatiopn step is include to give different levels of important to the different frequency components. The level of importance given to a frequency components is decide by the quantiation table. The quantization tables are prepared based upon psychvisual experiments. The DCT coefficient value in a specific position in the coefficient matrix is divided by the corresponding entry in the quantization table. Thus, DCT coefficient that a '1' entry in the quantization table show the step size.

The following example illustrates the procedure. Luminance and chrominance quantization tables show the step size.

1. Large step size indicates a coarse quantization or low bit allocation.
2. Small step size represents a higher bit allocation.

Perceptually design quantization table take advantage of masking properties of the eye and zero out many small coefficient. Table 11.3 shows a table for quantizing chrominanace component.

Table 11.3. Quantization table for chrominance components.

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

Table 11.4. 8 × 8 block of an Image.

124	125	122	120	122	119	117	118
121	121	120	119	119	120	120	118
126	124	123	122	121	121	120	120
124	124	125	125	126	125	124	124
127	127	128	129	130	128	127	125
143	142	143	142	140	139	139	139
150	148	152	152	152	152	150	161
156	159	158	155	158	158	157	156

Table 11.5. DCT Coefficient of Table 11.5 after level shift.

39.88	6.56	–2.24	1.22	–0.37	–1.08	0.79	1.13
–102.43	4.56	2.26	1.12	0.35	–0.63	–1.05	–0.48
37.77	1.31	1.77	0.25	–1.50	–2.21	–0.10	0.23
–5.67	2.24	–1.32	–0.81	1.41	0.22	–0.13	0.17
–3.37	–0.74	–1.75	–0.77	–0.62	–2.65	–1.30	0.76
5.98	–0.13	–0.45	–0.77	–1.99	–0.26	1.48	0.00
3.97	5.52	2.39	–0.55	–0.05	–0.84	–0.52	–0.13
–3.43	0.51	–1.07	–0.87	0.96	0.09	0.33	0.01

Example: Table 11.4 gives an 8 × 8 block from an image. Table 11.5 gives the DCT coefficients for the image block of Table 11.4 after level shift.

Example: Another example of 8 × 8 block of an image is shown in Table 11.6.

11.4.5. Zig-Zag Scan

The coefficients within the block are ordered according to zig-zag scan to increase the run length, because most of zeros occupy the diagonal positions. Figures 11.9 shows the order of zig-zag scan.

11.4.6. Run Length Coding

Zig-zag scanning increase he chance of the same amplitude coefficients appearing next to each other. The high frequency components are the most likely ones to give large sequences of zeros or other low values. Run length coding generates an intermediate symbol sequences which are statically encoded.

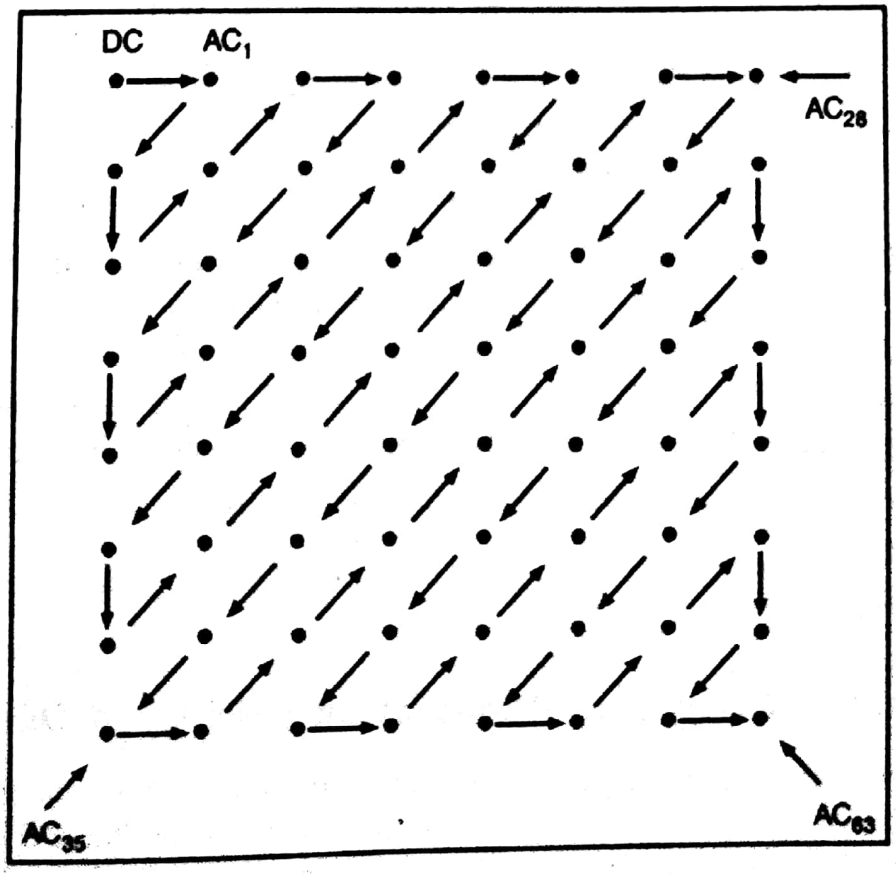

Fig. 11.9.

11.4.7. Statistical Encoding

Run length encoding is following by a statistical step: Huffman encoding and arithmetic encoding. Separate Huffman tables are used for DC and AC coefficients as their statistics are different. (DC coefficient are differentially coded from block to block). Lets us consider the matrix of coefficient got after DCT operation and quantization, as shown in Fig. 11.10(*a*). After zig-zag scanning, we get the sequence that is shown in Fig. 11.10(*b*). The first value for which DC coefficient in Fig. 11.10(*b*) is nor considered, and the other values *i.e.* AC coefficient are coded in the following format. For example, '6' is coded as (0,6) ⇒ (Run length, value).

The Run length *is the number of zeros in the run. The value is the next nonzero coefficient.*

Therefore he sequence

$$\{6, -1, -1, 0, -1,0,0,0 -1,0,0,1,0,0,\ldots,0\}$$

is coded as

$$(0,6), (0,-1), (0,-1), (1,-1), (3,-1), (2, 1), (0, 0)$$

(*a*)

32	6	–1	0	0	0	0
–1	0	0	0	0	0	0
–1	0	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	0	0	0

(*b*) {32, 6, –1, –1, 0, –1, 0, 0, 0, –1, 0, 0, 1, 0, 0, ..., 0}

Fig. 11.10.

The last value (0,0) is EOB, end of block indicator. The run length coding step replaces values by a pair (RUNLENGTH, VALUE) for each run of zeros in the run and value is the next non zero coefficient. Pair (0,0) indicates the end-of-block after the last non zero AC coefficient.

Value component in (RUNLENGTH,VALUE) pair, is further represented by size and amplitude. To save bits, run length and size are allocation only 4 bits and squeezed into a single byte-symbol 1.

Symbol 2 is the amplitude, its number of bits is indicated by size.

Symbol 1 : (Runlength, Size).

Symbol 2 : (Amplitude).

11.4.8. Differential Pulse Code Modulation (DPCM) of DC Coefficients

The DC coefficient are coded separately from the AC ones. Each 8 × 8 image block has only one DC coefficients for various blocks that could be large and different, because the DC value reflects the average intensity of each block. The DC coefficient is unlikely to change drastically within a short distance. This makes DPCM an ideal scheme for coding the DC coefficient. As an example, if the DC coefficient for the first five image blocks are 150, 155, 149, 152, &144.

Then, DPCM would produce 150, +5, –6, +3, –8, assuming the first sample is same, and the subsequent sample values are the difference of the present and previous samples.

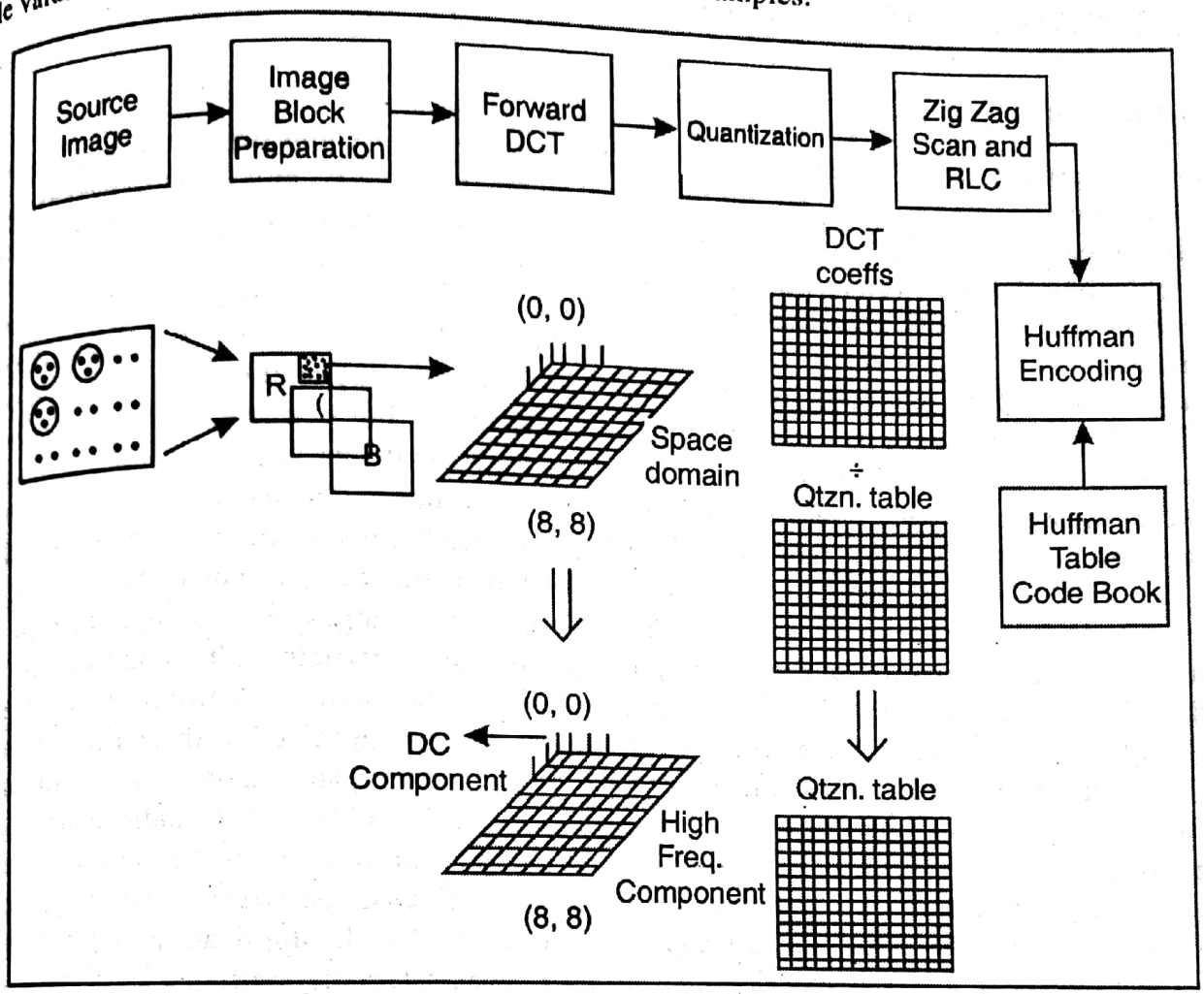

Fig. 11.11.

That is,

150 → 150 I sample,

155 – 150 → +5 II sample

149 – 155 → +6 III sample

DPCM for the DC coefficients in JPEG is carried out on the entire image at once, whereas the run length coding of the AC coefficient is performed on each individual block.

11.4.9. Statistical Encoding

Runlength coding is followed by a statistical encoding step. The JPEG standard allows a choice of two statistical encoding methods :

(i) Huffman encoding

(ii) Arithmetic encoding

Huffman encoding produces variable length codes that require a code-book. Huffman coding does not allow a non integer less than ideal entropy encoding. Arithmetical encoding makes it possible to use a non integer number of bits to represent input values, though it requires more processing.

11.5. SYNCHRONIZATION IN MULTIMEDIA SYSTEMS

Introduction

Synchronization in multimedia systems refers to temporal relationships between media objects in the multimedia systems. In future multimedia systems (based, e.g., on MPEG-4) synchronization may also refer to spatial and content relationships, as well as temporal. Synchronization between media objects comprises relationships between time-dependent media objects as well as time-independent media objects. Synchronization may need to occur at different levels in a multimedia system, consequently synchronization support is typically found in the operating system, communication system, databases, multimedia documents, and the application. A general scheme might involve a layered approach to achieving synchronization. For example, a Computer-Supported Collaborative Workgroup (CSCW) session might involved a multi-party video conferencing session with audio, and a shared whiteboard. Parties may make reference to objects on the shared whiteboard, using a pointer, to support what they are saying (e.g., saying "This area here..." while indicating the area with a pointer). Here, video and audio are continuous media objects which are highly periodic, whereas the shared whiteboard is a discrete media stream, as changes to it are highly irregular (the content, including the position of the pointer, depends on which participant has control of the object and when they make changes to it). The media streams must be highly synchronized, so that speech remains lip synchronized, and the whiteboard updates are synchronized with audio references to them. The operating system and lower levels of the communication system are responsible for ensuring that jitter on individual streams does not occur during presentation of the video, audio, and whiteboard streams (intramedia synchronization). At a higher level, the runtime support for the synchronization of multiple multimedia media streams must ensure that the various media streams remain synchronized with respect to each other (intermedia synchronization). Finally, the application(s) are responsible for ensuring synchronicity between application-level events (usually initiated by the users). For example, if the application at the source does not capture timing dependencies between a user waving the pointer over part of the object in the whiteboard and the supporting audio stream, then it will be impossible for the application at the sink to know that the whiteboard and audio events need to be synchronized.

The temporal relations between media objects must be specified during capture of the media objects, if the goal of the presentation is to present media in the same way that they were originially captured. Synchronization information of events in an animation sequence or a slide show is usually specified by the designer, using, for example, a time-axis.

11.6. A REFERENCE MODEL FOR MULTIMEDIA SYNCHRONIZATION

A reference model is needed to understand the requirements of multimedia synchronization, identify and structure runtime mechanisms that can support these requirements, identify interfaces between runtime mechanisms, and compare solutions for multimedia synchronization systems. Figure 11.12 shows a reference model for multimedia synchronization systems. Each layer implements synchronization mechanisms which are provided by an appropriate interface. These interfaces can be used to specify or enforce the temporal relationships. Each interface can be used by the application directly, or by the next higher layer to implement an interface. Higher layers offer higher programming and QoS abstractions.

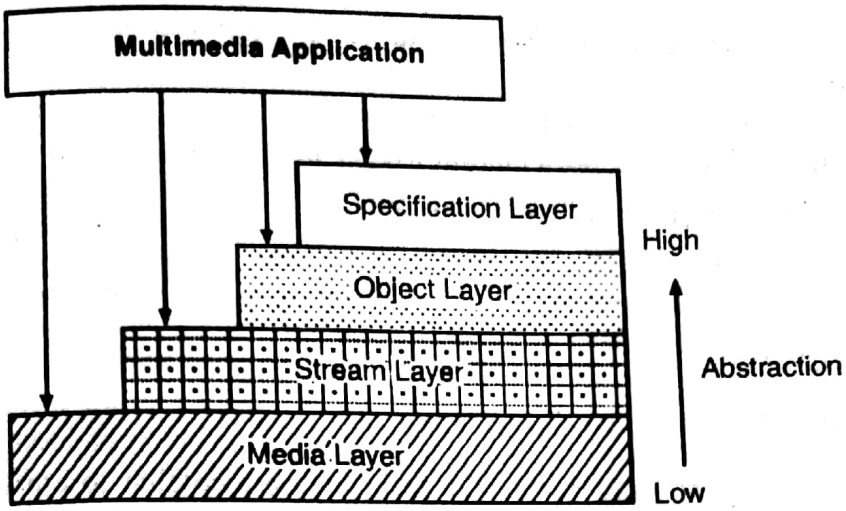

Fig. 11.12.

Media Layer

An application operates on a single continuous media stream, which is treated as a sequence of LDUs. Networking components must be taken into account. Provides access to files and devices.

Stream Layer

The stream layer operates on continuous media streams as well as groups of media streams. In a group, all streams are presented in parallel by using mechanisms for interstream synchronization. QoS parameters will specify intrastream and interstream synchronization requirements.

Continuous media is seen as a data flow with implicit time constraints; individual LDUs are not visible. An application using the stream layer is responsible for starting, stopping and grouping the streams, and for the definition of the required QoS in terms of timing parameters supported by the stream layer. It is also responsible for the synchronization with time-independent media objects. Tasks include resource reservation and LDU process scheduling.

Object Layer

The object layer operates on all media streams and hides the differences between continuous and discrete media. An application that interacts with this layer will be presented with a view of a complete, synchronized presentation. This layer takes a complete synchronization specification as its input and is responsible for the correct schedule of the overall presentation.

Specification Layer

This layer contains applications and tools that are allowed to create synchronization specifications (*e.g.*, authoring tools, multimedia document editors).

The specification layer is also responsible for mapping user-required QoS parameters to the qualities offered at the object layer interface.

Synchronization specifications can be:

- **Interval-based:** specifications of the temporal relations between the time intervals of the presentation of media objects

- **Axes-based:** Allows presentation events to be synchronized according to shared axes, e.g., a global timer
- **Control flow-based:** At specified points in presentations, they are synchronized
- **Event-based:** Events in the presentation trigger presentation actions

Synchronization in a Distributed Environment

Synchronization in a distributed environment is complex, because there may be more than one source of multimedia data, and more than one sink consuming it. The synchronization information for the various media stream may also reside at different sources.

Transport of the synchronization specification

The sink needs to have the synchronization information available to correctly display an object. There are three main approaches to delivering the synchronization information to the sink:

- Delivery of the synchronization information before the start of the presentation
- Use of an additional synchronization channel
- Multiplexed data streams

If the multimedia presentation is live and multiple parties are involved, then none of the approaches above is suitable for delivering synchronization information to the sink(s) in a timely fashion. Figure 11.13 shows typical communication patterns.

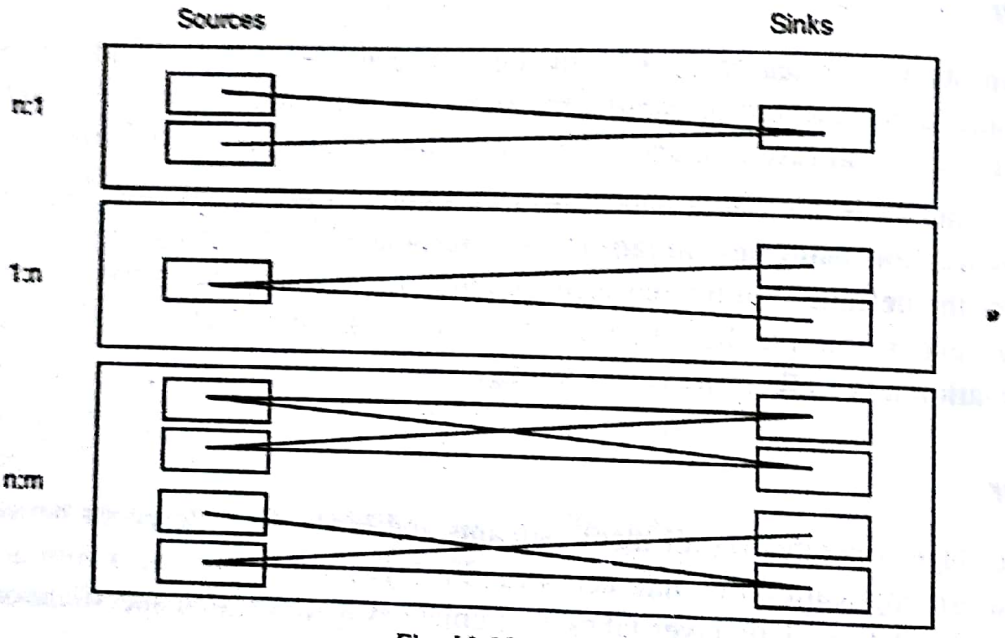

Fig. 11.13.

Of particular interest here, is that if multiple sinks are involved, then they will receive identical data. It would be inefficient if the data were replicated at the source for separate transmission to each of the sinks. It would also be inefficient if the same operation was carried out at different sinks. Multicasting *or broadcasting of streams is the responsibility of the stream layer, whereas efficient planning of operation execution in the different communication patterns is a responsibility of the object layer.*

Multi-Step Synchronization

In a distributed environment, synchronization is typically a multi-step process, during which the synchronization must be maintained so as to enable the sink to perform the final synchronization. The synchronization steps are:

- during object acquisition, *e.g.*, during frame digitization
- during retrieval, *e.g.*, synchronized access to frames of a stored video
- during delivery of the LDUs to the network
- during the transport of the LDUs, *e.g.*, using isochronous protocols
- at the sink, *e.g.*, synchronized delivery to the output devices
- within the output device

With many different points at which synchronization must occur decisions must be made about how to implement it. A first decision is the selection of the type of transport for the synchronization specification. In runtime, decisions must be taken concerning the location of synchronization operations, keeping clocks in synchrony (if used to provide common timing information), and the handling of multicast and broadcast messages. Coherent planning of the steps in the synchronization process, together with the necessary operations of the objects, *e.g.*, decompression, must also be done. In addition, presentation manipulation operations demand additional replanning at runtime.

Synchronization Specification

A synchronization specification should comprise:

- Intra-object synchronization specifications for the media objects of the presentation
- QoS descriptions for intra-object specifications
- Inter-object synchronization specifications for media objects of the presentation
- QoS descriptions for inter-object synchronization

In addition, the form, or alternate forms, of a multimedia object may be described. For example, a text could be presented as text on the screen or as a generated audio sequence. In the case of live synchronizations, the temporal relations are implicitly defined during capture. QoS requirements are specified before the start of the capture. In the case of synthetic synchronization, the specification must be created explicitly.

11.7. INTEGRATED MULTIMEDIA MESSAGE

To transmit the multimedia environment, the multimedia data must be captured and prepared for transmission. We focus our attention on the step where disparate sources of traffic are combined into a message. Here, we assume that the "message" is the unit of data acceptable to lower layers of whatever protocol hierarchy the real-time multimedia application uses. The integrated multimedia message format is imposed on a message which contains discrete media such as text and graphics, with interelement sequencing but lacking interelement timing, as well as continuous media such as video and voice, which add interelement timing requirements. The time dependency of all information types in the IMM as stated before is obvious. We can combine discrete and continuous media, since while the continuous media appear to listeners or viewers as continuously changing over time, their internal representation in a digital system is discrete. It consists of single audio samples or video frames. Each IMM thus is associated with a time interval in the time scale of the application. The time intervals are not assumed to be of fixed length, although this may be the most common case. The duration could be variable, depending on the properties of data comming from the devices such as camera and microphone. For example, long periods of silence or inactivity can be taken advantage of by not sending messages as frequently. However, in any case, any structural relationship such as sequencing must be preserved.

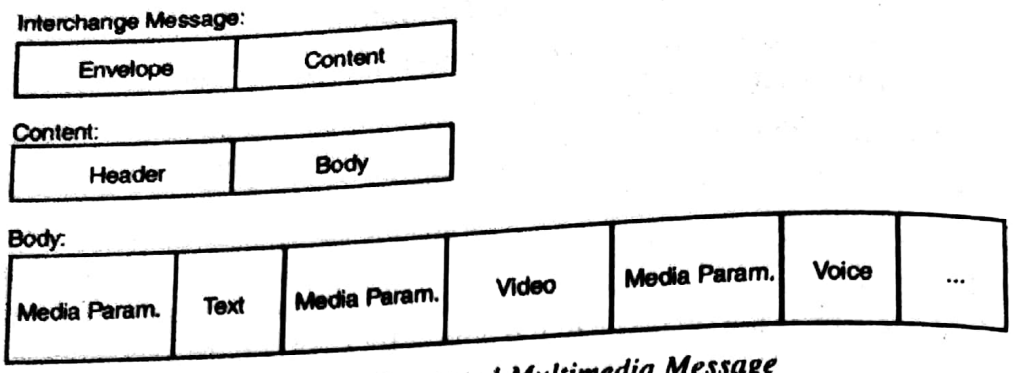

Fig. 11.14. Integrated Multimedia Message

Our IMM is illustrated in Fig. 11.14. The IMM contains a general descriptive Envelope, which includes the message addressing parameters and parameters for the intermessage synchronization. The Content includes the general Header and the actual multimedia data in the Body. The general header can be viewed as temporal relation descriptor which has the information for the intrarnessage synchronization. Media Parameters in the body are important for the presentation of each media, or for the articular application, the media has to be delivered to. A summary table of parameters and brief description for each is given in Table 11.6. In essence, the sender's data are multiplexed into a single multimedia stream composed of IMM's and demultiplexed on the receiver side for presentation or computation.

Table 11.6.

Structure-Elements	Functionality
Envelope	descriptive and addressing information
Receiver	Receiver address in the application layer
Sender	sender address
Message Identifier	unique identifier of the IMM
Message Type	in dependence of QOS (loss-rate sensitive data) divide the message if it is original or copy IMM
Time Begin	Beginning of the time intervalm, the IMM was created.
Time Duration	Length of the time interval, the IMM belongs to
Content	Contains all information the application layer need to provide the processing at the receiver side.
Header	Information related to the local handling of IMM
Body Description	Description of the body (single, composed)
Single Body	Specifies the information type, because the body has only one kind of information
Composed Body	Specifies the body parts in the body
Body Parts Relations	Relations among the body parts in the particular time interval (independent, synchronized)
QOS-maxima	QOS parameters for every media of the sender
Body	contains the actual content
Media Parameter	Specifies the type, length, time parameters of the body part
Identifier	Unique ordering number of the body part inside of the IMM
Time Begin	Time when the body part was created
Time Duration	Duration time of the body part
Optimization of Data	Media specific processing (e.g., compression)
Data	Data with specific description of layout, etc. according to application - specific standards