PART II

VIDEO PROCESSING

CHAPTER 20

VIDEO FUNDAMENTALS

WHAT WILL WE LEARN?

- What is an analog video raster and what are its main components and parameters?
- What are the most popular analog TV and video standards?
- What is digital video and how it differs from analog video?
- What are the most popular digital video standards?
- How is color information encoded in analog and digital video systems?
- How can we read, manipulate, and play digital video files in MATLAB?

20.1 BASIC CONCEPTS AND TERMINOLOGY

In this section, we present a list of technical concepts and terms used in analog and digital TV and video systems.[1] Similar to what we have done earlier (Section 1.2), this section is structured in a question-and-answer format in a sequence that starts with relatively simple concepts and builds up to more elaborated ones.

[1] Since many of these concepts are interconnected, there is no perfect sequence by which they should be presented. The reader is encouraged to read the whole chapter even if certain parts of the text refer to concepts that have not been officially introduced yet. I believe and hope that at the end of the chapter, the entire picture will be clear in the reader's mind.

Practical Image and Video Processing Using MATLAB®. By Oge Marques.

What is a Video Signal?

A *video signal* is a one-dimensional (1D) analog or digital signal varying over time whose spatiotemporal contents represent a sequence of images (or *frames*) according to a predefined scanning convention. Mathematically, a continuous (analog) video signal will be denoted by $f(x, y, t)$, where t is the temporal variable.

An *analog* video signal refers to a 1D electrical signal $f(t)$ obtained by sampling $f(x, y, t)$ in the vertical and temporal dimensions. A *digital* video signal is also sampled along the horizontal axis of each frame.

What is Scanning?

Scanning is a method used by all video systems as part of the process of converting optical images into electrical signals. Figure 20.1 shows the basic scanning operation in a television camera. During the scanning process, an electronic sensing spot moves across the image in a pattern known as a *raster*. The sensing spot converts differences in brightness into differences in instantaneous voltages. Starting at the upper left corner of the image, the spot moves in a horizontal direction across the frame to produce a scanning line. It then quickly returns to the left edge of the frame (a process called *horizontal retrace*) and begins scanning another line. These lines are slightly tilted downward, so that after the retrace the spot is just below the previously scanned line and ready to scan a new line. After the last line is scanned (i.e., when the sensing spot reaches the bottom of the image), both horizontal and vertical retraces occur, bringing the spot back to the upper left corner of the image. A complete scan of the image is called a *frame* [LI99].

Scanning also occurs at the time of reproducing the frames on a display device. The main difference, of course, is the replacement of the sensing spot by a spot of

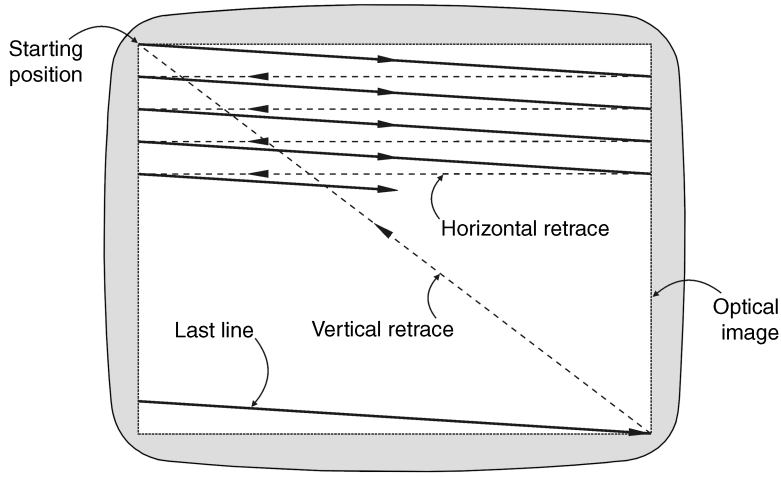

FIGURE 20.1 Scanning raster. Redrawn from [LI99].

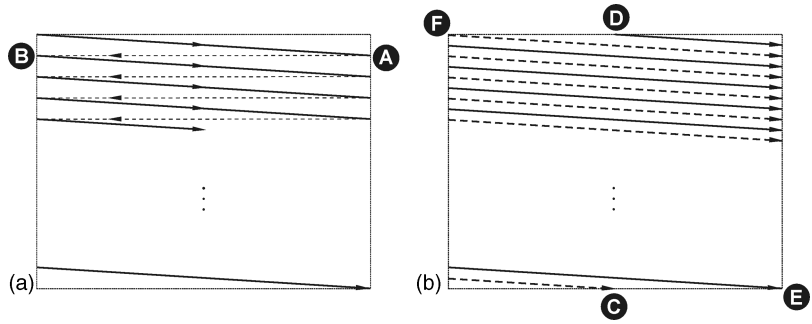

FIGURE 20.2 Scan and retrace: (a) progressive scan (dashed lines indicate horizontal retrace); (b) interlaced scan (solid and dashed lines represent even and odd fields, respectively). Adapted and redrawn from [WOZ02].

light whose intensity is controlled by the electrical signal originally produced at the output of the camera.

What are the Differences Between Interlaced and Progressive Scanning?

Progressive scanning is the process by which each image is scanned in one single pass, called *frame*, at every Δt (Figure 20.2a). It is used, for example, in computer displays, where a representative value for Δt is 1/72 s. Progressive scanning is a natural way to create an analog video raster. However, due to technological limitations at the time the early analog TV systems were being developed, it was not possible to implement it. As a result, the concept of *interlaced scanning* was proposed, which trades off spatial resolution for temporal resolution.[2]

With interlaced scanning (Figure 20.2b), each frame is scanned in two successive vertical passes, first the odd numbered lines and then the even numbered ones. Each pass is called a *field*. Since each field contains half of the lines in a frame, it lasts only one-half of the duration of an entire frame. Consequently, fields flash at a rate twice as fast as entire frames would, creating a better illusion of motion. If the field flash rate is greater than the *critical flicker frequency* (CFF) for the human eye, the result is successful and the motion is perceived as smooth. Perhaps more importantly, even though each field contains only half of the lines (in other words, half of the horizontal resolution that one would expect), viewers tolerate this drop in spatial resolution and are usually content with the overall quality of the resulting video.

[2]Appendix A has a more detailed explanation of spatial and temporal resolution, the relationship between them, and the implications of human visual perception experiments in the design of analog and digital video systems.

What is Blanking Interval?

It is the time interval at the end of each line (*horizontal retrace*, A to B in Figure 20.2a) or field (*vertical retrace*, C to D and E to F in Figure 20.2b) during which the video signal must be blanked before a new line or field is scanned.

What is Refresh Rate?

Most displays for moving images involve a period when the reproduced image is absent from the display, that is, a fraction of the frame time during which the display is black. To avoid objectionable flicker, it is necessary to flash the image at a rate higher than the rate necessary to portray motion. The rate at which images are flashed is called *flash* (or *refresh*) rate.

Typical refresh rates are 48 (cinema), 60 (conventional TV), and 75 Hz (computer monitors). The refresh rate highly depends on the ambient illumination: the brighter the environment, the higher the flash rate must be in order to avoid flicker. Refresh rates also depend on the display technology. In conventional movie theater projection systems, a refresh rate of 48 Hz is obtained by displaying each negative of a 24 frames/s movie twice. For progressive scan systems, the refresh rate is the same as the *frame rate*, whereas in the case of interlaced scan systems, the refresh rate is equivalent to the *field rate* (twice the frame rate).

What is the Meaning of Notation Such As 525/60/2:1, 480i29.97, or 720p?

There is no universally adopted scanning notation for video systems, which may be a potential source of confusion, due to the lack of consistency among different sources.[3]

Analog monochrome video scanning systems can be denoted by

- Total number of lines (including sync and blanking overhead)
- Refresh rate (in Hz) (which will be equal to the field rate for interlaced scan or the frame rate for progressive scan)
- Indication of interlaced (2:1) or progressive (1:1) scan

According to this notation, the analog TV system used in North America and Japan would be represented by 525/60/2:1 (or, more accurately, 525/59.94/2:1) and the European SDTV (standard definition TV) would be 625/50/2:1.

A more compact way of indicating the same information is by concatenating the number of visible lines per frame with the type of scanning (*p*rogressive or *i*nterlaced)

[3]For example, the use of NTSC to refer to 525 lines/60 Hz/interlaced monochrome video systems is common, but not accurate (after all, NTSC is a color TV standard). Worse yet is the use of PAL to refer to the 625 lines/50 Hz/interlaced monochrome video system used in great part of Europe, since PAL not only is a color encoding standard, but also contains a great number of variants. For a concrete example of how confusing this may get, in Brazil, the analog color TV standard adopted for TV broadcast (PAL-M) uses the PAL color encoding on a 525 lines/60 Hz/interlaced monochrome system.

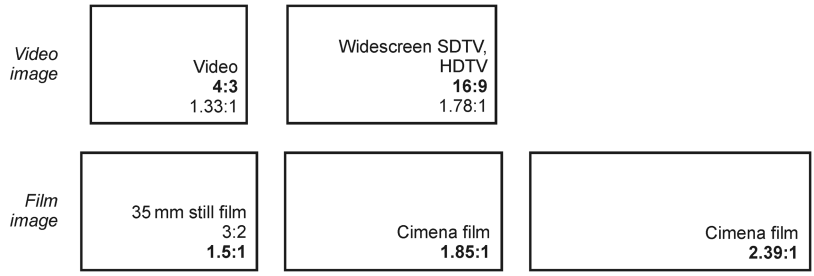

FIGURE 20.3 Aspect ratios of SDTV, HDTV, and film. Redrawn from [Poy03].

and the vertical frequency (roughly equal to the frame rate), for example, 480i29.97 and 576i25.

HDTV (high-definition TV) standards are usually represented in an even more compact notation that includes the number of lines and the type of scanning. Contemporary examples include the 720p and the 1080i standards.

What is Aspect Ratio?

Aspect ratio (AR) is the ratio of frame width to height. In the case of digital video, both dimensions can be specified in terms of numbers of pixels (e.g., 640 × 480 or 1920 × 1200). For analog video, however, only the number of lines can be counted and expressed as an integer value.

The most common values for TV are 4:3 (1.33:1) for SDTV and 16:9 (1.78:1) for HDTV. For movies, it usually varies between 1.66:1 and 2.39:1. Figure 20.3 shows representative examples of aspect ratios.

What is Gamma Correction?

Video acquisition and display devices are inherently nonlinear: the intensity of light sensed at the camera input or reproduced at the display output is a nonlinear function of the voltage levels.

For the cameras, this relationship is usually expressed as

$$v_c = {B_c}^{-\gamma_c} \tag{20.1}$$

where B_c represents the actual luminance (light intensity), v_c is the resulting voltage at the output of the camera, and γ_c is a value usually between 1.0 and 1.9.

Similarly, for display devices, the nonlinear relationship between the input voltage v_d and the displayed color intensity B_d is expressed as

$$B_d = {v_d}^{\gamma_d} \tag{20.2}$$

where γ_d is a value usually between 2.2 and 3.0.

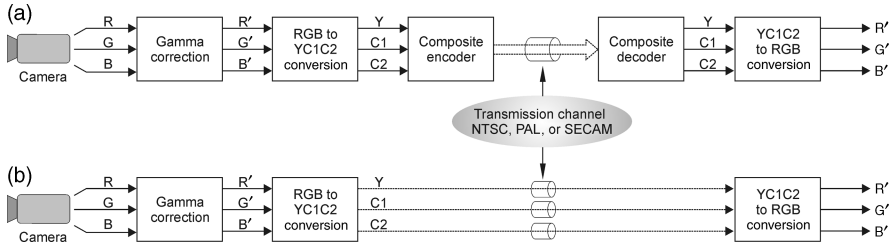

FIGURE 20.4 Gamma correction in video and TV systems: (a) composite video; (b) component video.

The process of compensating for this nonlinearity is known as *gamma correction* and is typically performed at the transmitter side. By gamma correcting the signal before displaying it, the intensity output of the display is roughly linear.

In addition to precompensating for the nonlinearity of the CRT display, gamma correction also codes the luminance information into a perceptually uniform space, thus compensating for the nonlinear characteristics of the human visual system in a way that Poynton calls an "amazing coincidence" [Poy03]. Moreover, the gamma-corrected signal also becomes less sensitive to noise.

Figure 20.4 shows how gamma correction is performed in a typical analog video system. First, a nonlinear transfer function is applied to each of the linear color components: R, G, and B right at the output of the color camera, resulting in the *gamma-corrected primaries* R', G', and B'. Then, a weighted sum of the nonlinear components is computed to form two generic chrominance signals (C1 and C2) and a luma signal, Y', representative of brightness, given by:[4]

$$Y' = 0.30R' + 0.59G' + 0.11B' \tag{20.3}$$

The resulting signals (Y', C1, and C2) are then transmitted—with or without being combined into a single composite signal—to the receiver, where eventually they are converted back into the gamma-corrected primaries (R', G', and B'), which will be used to drive the display device.

What are Component, Composite, and S-Video?

From our discussion on color (Chapter 16), we know that color images can be described by assigning three values (usually the R, G, and B color components) to each pixel. An extension of this representation scheme to color video requires using three independent one-dimensional color component signals (e.g., $R'G'B'$ or $Y'C_BC_R$), free from mutual interference, in what is known as *component analog video* (CAV).

Component video representation is convenient and feasible for certain applications (e.g., connecting a DVD player to a TV set) but could not be adopted for color TV

[4]The values of the coefficients in equation (20.3) are approximate. The exact values will vary among different analog video standards.

broadcast systems. In that case, primarily for backward compatibility reasons, a video encoding system was designed that combines the intensity and color information into a composite signal. This is known as *composite video*. A composite video signal relies on the fact that the chrominance components can be encoded using a significantly smaller bandwidth than the luminance component. Composite video systems combine brightness and color into one signal, at the expense of introducing a certain degree of mutual interference. For more details, refer to Section 20.3.

The *S-video* (also known as *Y/C video*) standard is an intermediate solution consisting of two components, luminance and a multiplexed chrominance component. S-video systems require less bandwidth (or data rate) than component video and produce better image quality than composite video.

20.2 MONOCHROME ANALOG VIDEO

In this section, we expand the discussion of the fundamental concepts and the most important aspects of monochrome (black-and-white) analog video.

20.2.1 Analog Video Raster

An analog video raster can be defined by two parameters: the *frame rate* (*FR*) (in frames/s or fps or Hz) that defines the temporal sampling rate and the *line number* (N_L) (in lines/frame or lines/picture height) that corresponds to the vertical sampling rate. These two parameters can be used to define other useful terms, such as

- Line rate (in lines/s): the product of FR and N_L.
- Temporal sampling interval (or frame interval): $\Delta_t = 1/\text{FR}$.
- Vertical sampling interval (or line spacing): $\Delta_y = \text{PH}/N_L$, where PH is the picture height.
- Line interval: $T_l = \Delta_t/N_L$, which is the total time required to scan a line (including the horizontal retrace, T_h).
- Actual scanning time for a line: $T'_l = T_l - T_h$.
- Number of active lines: $N'_L = (\Delta_t - T_v)/T_L = N_L - T_v/T_l$, where T_v is the *vertical retrace* and is usually chosen to be a multiple of T_l .

A typical waveform for an interlaced analog video raster is shown in Figure 20.5.

The maximum rate at which an analog video signal can change from one amplitude level to another is determined by the bandwidth of the video system. The bandwidth requirements of an analog TV system can be calculated by

$$\text{BW} = \frac{K\ \text{AR}\ N'_L}{2T'_l} \tag{20.4}$$

where BW is the system bandwidth (in hertz), AR is the aspect ratio, N'_L is the number of active TV scanning lines per frame, T'_l is the duration of a line, and K is

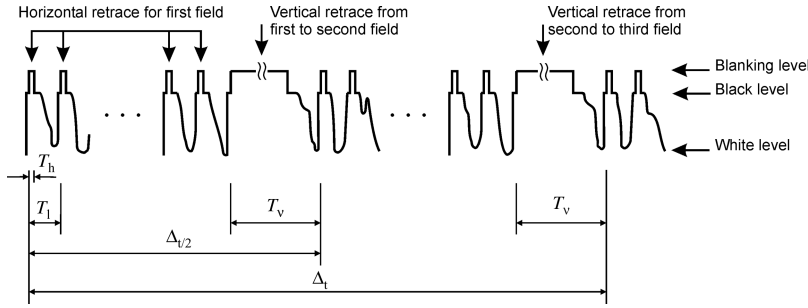

FIGURE 20.5 Typical interlaced video raster. Redrawn from [WOZ02].

the *Kell factor*: an attenuation factor (whose value is usually 0.7) that accounts for the differences between the *actual* vertical resolution of a frame and the *perceived* resolution when the human visual acuity is taken into account.

■ EXAMPLE 20.1

Calculate the required bandwidth for the luminance signal of an analog TV system with the following characteristics:

- 2:1 interlaced scanning
- 525 horizontal lines/frame, 42 of which are blanked for vertical retrace
- 30 frames per second
- 4:3 aspect ratio
- 10 μs of horizontal blanking
- Kell factor = 0.7

Solution

$T_l = 1/(525 \times 30) = 63.5\,\mu s$
$T_l' = 63.5 - 10 = 53.5\,\mu s$
$N_L' = 525 - 42 = 483$ lines
$BW = (0.7 \times 4/3 \times 483)/(2 \times 53.5 \times 10^{-6}) = 4.21$ MHz

20.2.2 Blanking Intervals

Most analog TV and video systems use a composite signal that includes all the information needed to convey a picture (brightness, color, synchronization) encoded into a single one-dimensional time-varying signal. As discussed previously, part of the time occupied by the signal is used to bring the scanning starting point back to the beginning of a new line or frame, a process known as horizontal and vertical retraces, respectively. These retrace periods are also known as *blanking intervals*, during which the amplitude of the signal is such that it cannot be seen on the screen (indicated in Figure 20.5 as "blanking level").

The duration of the *vertical blanking interval* (VBI) (also known as *vertical retrace*, T_v) is approximately 7.5% of the frame interval (Δ_t). The *horizontal blanking interval* (also known as *horizontal retrace*, T_h) usually lasts 14–18% of the total line interval (T_l).

20.2.3 Synchronization Signals

In analog TV and video systems, it is necessary to establish a way by which the scanning process at the display device is synchronized with the scanning process that took place in the imager (camera). This is done by adding synchronizing (or simply *sync*) pulses to the horizontal and vertical blanking intervals. To ensure that these pulses do not interfere with the purpose of the blanking intervals, their amplitude is such as to correspond to "blacker than black" luminance levels. This also allows them to be easily separated from the rest of the video by simply clipping the composite video signal, in a process known as *sync separation*.

20.2.4 Spectral Content of Composite Monochrome Analog Video

A monochrome analog video signal usually covers the entire frequency spectrum from DC to the maximum frequency determined by the desired picture resolution (see equation (20.4)). However, the frequency spectrum is not continuous—it contains a fine-grained structure determined by the scanning frequencies (Figure 20.6).

The spectrum shown in Figure 20.6 reflects the fact that video signals contain three periodic components: one repeating at the line rate, one at the field rate, and one at the

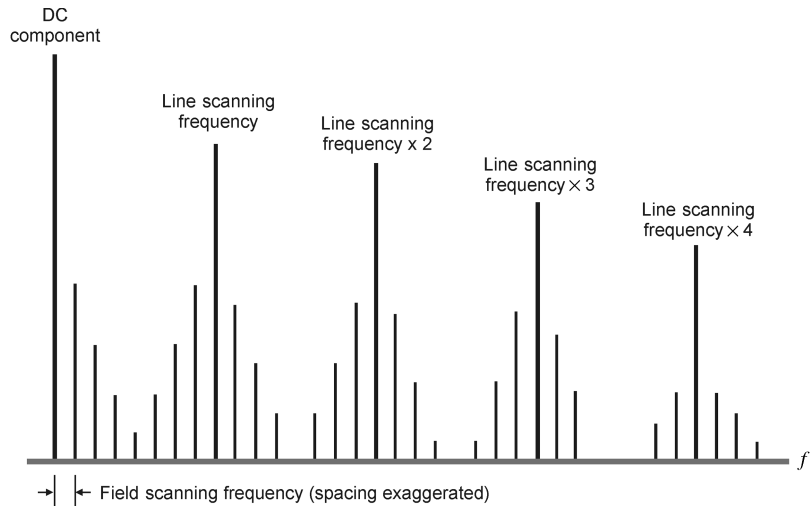

FIGURE 20.6 Fine-grained frequency spectrum of a monochrome analog video signal. Redrawn from [LI99].

frame rate (at 15,750, 60, and 30 Hz, respectively, for the NTSC standard adopted in the United States). Each component can be expressed as a Fourier series—a sequence of sinusoidal terms with frequencies that are multiple (harmonics) of the repetition rate and amplitudes determined by the waveform of the component. The summation of the amplitudes of the terms for the three periodic components over the entire video spectrum is the frequency content of the signal.

Figure 20.6 also shows that the spectrum of a monochrome TV signal consists of a series of harmonics of the line frequency, each surrounded by a cluster of frequency components separated by the field frequency. The amplitude of the line frequency components is determined by the horizontal variations in brightness, whereas the amplitude of the field frequency components is determined by the vertical brightness variations.

The gaps in the frequency spectrum of a monochrome TV signal have been utilized for two main purposes:

1. Interleave frequency components of the *color subcarrier* and its sidebands, making color TV systems backward compatible with their monochrome predecessors (see Section 20.3).
2. Improve the high-frequency signal to noise ratio using *comb filters*.

20.3 COLOR IN VIDEO

In this section, we discuss the most relevant issues associated with representing, encoding, transmitting, and displaying color information in analog TV and video systems. It builds upon the general knowledge of color in image and video (Chapter 16) and focuses on specific aspects of color modulation and encoding in analog TV and video systems.

Historically, the development of color TV came about at a time where monochrome TV already enjoyed widespread popularity. As a result of the existing equipment and technologies, and frequency spectrum regulations already in place, the development of color TV needed to be an extension of the existing system and was required to maintain *backward compatibility* with its predecessor. This requirement will be the underlying theme of the discussion that follows.

Analog color TV and video systems utilize red, green, and blue as primary colors that are then gamma corrected (resulting in R', G', B') and used to calculate a luma value:

$$Y' = 0.299R' + 0.587G' + 0.114B' \tag{20.5}$$

The R', G', B', and Y' signals are then combined into *color-difference* signals $(B' - Y')$ and $(R' - Y')$, in a process known as *matrixing*, according to the following equations:

$$B' - Y' = -0.299R' - 0.587G' + 0.889B' \tag{20.6}$$

$$R' - Y' = 0.701R' - 0.587G' - 0.114B' \tag{20.7}$$

The remaining color-difference signal ($G' - Y'$) does not need to be transmitted. It can be re-created at the receiver's side by a simple arithmetic combination of the other two color-difference signals.

The color-difference signals are then multiplied by a scaling factor (e.g., 0.493 for the ($B' - Y'$) and 0.877 for the ($R' - Y'$)) and in the case of NTSC rotated by a certain angle (33°). Those two signals are then usually modulated in quadrature (90° phase difference) and the result is used to modulate a color subcarrier whose bandwidth is usually between 0.5 and 1.5 MHz (significantly narrower than the 4.5 MHz typically used for the luma component), resulting in a single-wire composite video signal with a total bandwidth suited to the specific transmission standard.

At the receiver's side, first a decoder separates the composite signal into luma and chroma. The chroma portion is then bandpass filtered and demodulated to recover the color-difference signals that are then added to a delayed (to account for the processing time through filters and demodulators) version of the luma signal to a matrix circuit in charge of regenerating the gamma-corrected primaries needed as input signals for the monitor.

The details of the resulting composite video varies from one standard to the next (see Section 20.4) but have the following characteristics in common [RP00]:

- *Monochrome Compatibility*: A monochrome receiver must reproduce the brightness content of a color TV signal correctly and without noticeable interference from the color information.
- *Reverse Compatibility*: A color receiver must reproduce a monochrome signal correctly in shades of gray without spurious color components.
- *Scanning Compatibility*: The scanning system used for color systems must be identical to the one used by the existing monochrome standard.
- *Channel Compatibility*: The color signal must fit into the existing monochrome TV channel and use the same spacing (in Hz) between the luminance and audio carriers.
- *Frequency-Division Multiplexing*: All systems use two narrowband color-difference signals that modulate a color subcarrier and the chrominance and luminance signals are frequency-division multiplexed to obtain a single-wire composite video signal with a total bandwidth suited to the specific transmission standard.

Composite color encoding has three major disadvantages [Poy03]:

1. Some degree of mutual interference between luma and chroma is inevitably introduced during the encoding process.
2. It is impossible to perform certain video processing operations directly in the composite domain. Even something as simple as resizing a frame requires decoding.

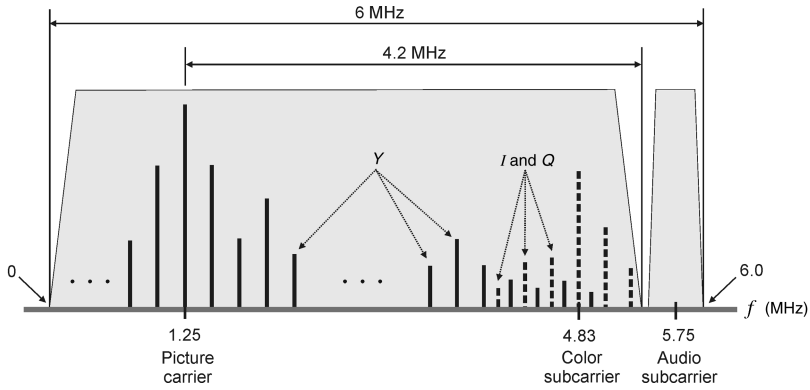

FIGURE 20.7 NTSC spectrum, showing how luminance (Y) and chrominance (I and Q) signals are interleaved. Redrawn from [LD04].

3. Digital compression techniques such as MPEG (Motion Pictures Expert Group) cannot be directly applied to composite video signals; moreover, the artifacts of NTSC or PAL encoding are destructive to MPEG encoding.

The spectral contents of composite color analog video differ from their monochrome equivalent because color information encoded on a high-frequency subcarrier is superimposed on the luminance signal (Figure 20.7). The frequency values have been carefully calculated so that the new spectral components due to color occupy gaps in the original monochrome spectrum shown earlier (Figure 20.6).

20.4 ANALOG VIDEO STANDARDS

In this section, we present a summary of some of the most relevant analog TV and video standards. There are two scanning standards used for conventional analog TV broadcast: the 480i29.97 used primarily in North America and Japan, and the 576i25 used in Europe, Asia, Australia, and Central America. The two systems share many features, such as 4:3 aspect ratio and interlaced scanning. Their main differences reside on the number of lines per frame and the frame rate.

Analog broadcast of 480i usually employs NTSC color coding with a color subcarrier of about 3.58 MHz; analog broadcast of 576i usually adopts PAL color encoding with a color subcarrier of approximately 4.43 MHz. Exceptions to these rules include the PAL-M system (480i scanning combined with PAL color coding) used in Brazil and the PAL-N system (576i scanning combined with a 3.58 MHz color subcarrier nearly identical to the NTSC's subcarrier) used in Argentina, among others.

20.4.1 NTSC

The NTSC (National Television System Committee) TV standard is an analog composite video standard that complies with the 480i component video standard. It uses two color-difference signals (U and V) that are scaled versions of the color-difference signals ($B' - Y'$) and ($R' - Y'$), respectively. The U and V color difference components are subject to low-pass filtering and combined into a single chroma signal, C:

$$C = U \sin \omega t + V \cos \omega t \tag{20.8}$$

where $\omega = 2\pi f_{sc}$ and f_{sc} is the frequency of color subcarrier (approximately 3.58 MHz).

In the past, to be compliant with FCC regulations for broadcast, an NTSC modulator was supposed to operate on I and Q components (scaled and rotated versions of U and V), where the Q component (600 kHz) was bandwidth limited more severely than the I component (1.3 MHz):

$$C = Q \sin(\omega t + 33^\circ) + I \cos(\omega t + 33^\circ) \tag{20.9}$$

Figure 20.7 shows details of the frequency interleaving between luma and chroma harmonics, as well as the overall spectral composition of the NTSC signal (including the audio component). In analog TV systems, audio is transmitted by a separate transmitter operating at a fixed frequency offset (in this case, 4.5 MHz) from the video transmitter. Contemporary NTSC equipment modulate equiband U and V color-difference signals.

20.4.2 PAL

The PAL (phase alternating line) TV standard is an analog composite video standard that is often used in connection with the 576i component video standard.

It uses two color-difference signals (U and V) that are scaled versions of the color-difference signals ($B' - Y'$) and ($R' - Y'$), respectively. The U and V color difference components are subject to low-pass filtering and combined into a single chroma signal, C:

$$C = U \sin \omega t \pm V \cos \omega t \tag{20.10}$$

where $\omega = 2\pi f_{sc}$ and f_{sc} is the frequency of color subcarrier (approximately 4.43 MHz).

The main difference between equations (20.10) and (20.8) reflects the fact that the V component switches phases on alternating lines, which is the origin of the acronym PAL and its most distinctive feature.

TABLE 20.1 Parameters of Analog Color TV Systems

Parameter	NTSC	PAL	SECAM
Field rate	59.94	50	50
Line number/frame	525	625	625
Line rate (lines/s)	15,750	15,625	15,625
Image aspect ratio (AR)	4:3	4:3	4:3
Color space	YIQ	YUV	YDbDr
Luminance bandwidth (MHz)	4.2	5.0, 5.5	6.0
Chrominance bandwidth (MHz)	1.5 (I), 0.5 (Q)	1.3 (U, V)	1.0 (U, V)
Color subcarrier (MHz)	3.58	4.43	4.25 (Db), 4.41 (Dr)
Color modulation	QAM	QAM	FM
Audio subcarrier (MHz)	4.5	5.5, 6.0	6.5
Composite signal bandwidth (MHz)	6.0	8.0, 8.5	8.0

Reproduced from [WOZ02].

20.4.3 SECAM

The SECAM (*Séquentiel couleur à mémoire*) standard is a color TV system with 576i25 scanning used in France, Russia, and a few other countries. Table 20.1 summarizes the main parameters of SECAM and their equivalent in the PAL (576i25) and NTSC (480i29.97) standards.

20.4.4 HDTV

HDTV is a name usually given to TV and video systems where each frame has 720 or more active lines. The most common variants of HDTV in use today are the 1280×720 and 1920×1080 image formats, also referred to as 720p60 and 1080i30 (or simply 720p and 1080i), respectively.

In addition to significantly higher spatial resolution, two salient differences between HDTV and SDTV are the aspect ratio of 16:9 instead of 4:3 and the use of progressive scanning.

20.5 DIGITAL VIDEO BASICS

A digital video may be obtained by sampling a raster scan (which requires analog-to-digital conversion (ADC), see Section 20.6) or directly using a digital video camera. Most of contemporary digital video cameras use CCD sensors. Similar to their analog counterpart, digital cameras sample the imaged scene over time, resulting in discrete frames. Each frame consists of output values from a CCD array, which is by nature discrete in both horizontal and vertical dimensions.

The result of the video acquisition stage—whether the digitization takes place within the camera or is performed by an external ADC embedded, for example, in a video capture card—is a collection of *samples*. These samples are numerical representation of pixel values along a line, for all the lines within a frame. Similar to

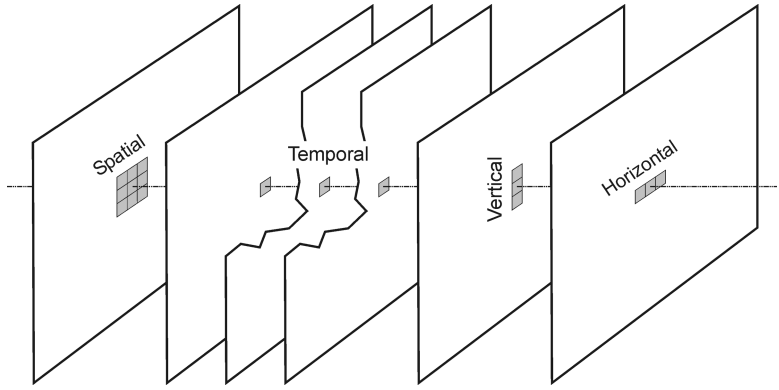

FIGURE 20.8 Sampling in the horizontal, vertical, and temporal dimensions. Redrawn from [Poy03].

their analog counterpart, to portray a smooth motion, digital video frames are captured and displayed at a specified frame rate.

Digital video can be understood as an alternative means of carrying a video waveform, in which the limitations of analog signals (any analog signal received at the destination is a valid one, regardless of any distortion, noise, or attenuation introduced on the original analog signal) are overcome by modulation techniques (such as PCM (pulse code modulation)) that enforce that each sample is encoded with a certain amplitude (from a finite set of values, known as *quantization levels*).

In analog video systems, the time axis is sampled into frames and the vertical axis is sampled into lines. Digital video simply adds a third sampling process along the lines (horizontal axis) (Figure 20.8).

20.5.1 Advantages of Digital Video

Digital representation of signals in general, and video in particular, have a number of well-known advantages over their analog counterpart [LI99]:

- Robustness to signal degradation. Digital signals are inherently more robust to signal degradation by attenuation, distortion, or noise than their analog counterpart. Moreover, error correction techniques can be applied so that distortion does not accumulate over consecutive stages in a digital video system.
- Smaller, more reliable, less expensive, and easier to design hardware implementation.
- Certain processes, such as signal delay or video special effects, are easier to accomplish in the digital domain.
- The possibility of encapsulating video at multiple spatial and temporal resolutions in a single scalable bitstream.
- Relatively easy software conversion from one format to another.

20.5.2 Parameters of a Digital Video Sequence

A digital video signal can be characterized by

- The frame rate ($f_{s,t}$)
- The line number ($f_{s,y}$)
- The number of samples per line ($f_{s,x}$)

From the above three quantities, we can find

- The temporal sampling interval or *frame interval* $\Delta_t = 1/f_{s,t}$
- The vertical sampling interval $\Delta_y = \text{PH}/f_{s,y}$, where PH is the picture height
- The horizontal sampling interval $\Delta_x = \text{PW}/f_{s,x}$, where PW is the picture width

In this book, we will use the notation $f(m, n, k)$ to represent a digital video, where m and n are the row and column indices and k is the frame number. The relationships between these integer indices and the actual spatial and temporal locations are $x = m\Delta_x$, $y = n\Delta_y$, and $t = k\Delta_t$.

Another important parameter of digital video is the number of bits used to represent a pixel value, N_b. For monochrome video, $N_b = 8$, whereas color videos require 8 bits per color component, that is, $N_b = 24$.

The data rate of the digital video, R, can be determined as

$$R = f_{s,t} \times f_{s,x} \times f_{s,y} \times N_b \quad \text{(in bps)} \tag{20.11}$$

Since the sampling rates for luma and chroma signals are usually different (see Section 20.7), N_b should reflect the equivalent number of bits used for each pixel in the sampling grid for the luma component. For example, if the horizontal and vertical sampling rates for each of the two chroma components are both half of that for the luma, then there are two chroma samples for every four luma samples. If each sample is represented by 8 bits, the equivalent number of bits per sample in the Y' (luma) resolution is $(4 \times 8 + 2 \times 8)/4 = 12$ bits.

When digital video is displayed on a monitor, each pixel is rendered as a rectangular region with constant color. The ratio of the width to the height of this rectangular area is the *pixel aspect ratio* (*PAR*). It is related to the aspect ratio of the entire frame (AR) by

$$\text{PAR} = \text{AR}\frac{f_{s,x}}{f_{s,y}} \tag{20.12}$$

Computer displays usually adopt a PAR of 1. In the TV industry, nonsquare pixels are used for historical reasons and PAR may vary from 8/9 (for 525/60 systems) to 16/15 (for 625/50 systems).

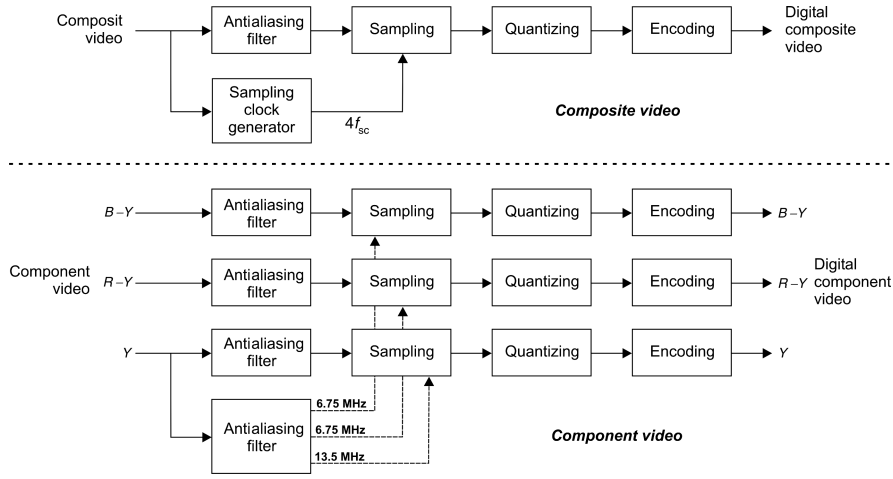

FIGURE 20.9 Analog-to-digital converters for composite (top) and component (bottom) video. Redrawn from [LI99].

20.5.3 The Audio Component

In analog TV systems, audio is transmitted by a separate transmitter operating at a fixed frequency offset from the video transmitter. In computer originated digital video, audio is typically encoded in a data stream that can be handled separately or interleaved with video data. In digital TV, audio is interleaved with video. In all cases, the audio part of the system involves processes equivalent to the ones used in video: creation, storage, transmission, and reproduction.

Digital audio formats are usually based on the PCM technique and its variants. The number of bits per sample and the sampling frequency are the determinant factors of the resulting data rate (and associated audio quality): from 12 kbps (or less) for speech to 176 kbps (or more) for CD quality stereo music.

20.6 ANALOG-TO-DIGITAL CONVERSION

Since digital video sequences are often the result of capturing and encoding a real-world analog scene, at some point an analog-to-digital conversion step is needed.[5] All digital cameras and camcorders perform ADC at the imager. Analog cameras output analog video that eventually needs to undergo ADC.

Figure 20.9 shows schematic block diagrams for analog-to-digital converters for composite and component video systems. The key components are described below [LI99].

[5]This statement implies that if the video sequence is generated from scratch by a computer application, no such conversion is necessary.

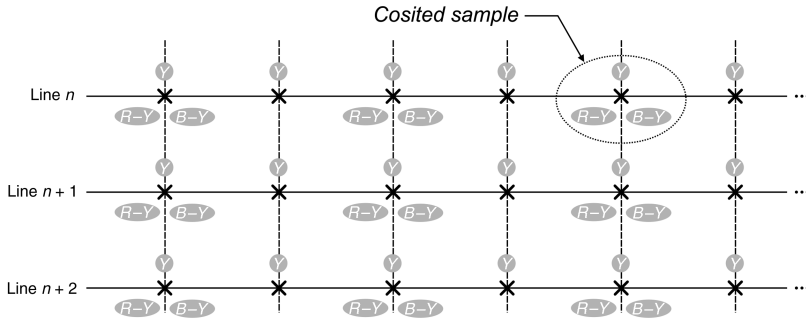

FIGURE 20.10 Location of sampling points in component video signals. Redrawn from [LI99].

Antialiasing Filter An optional low-pass filter with a cutoff frequency below the Nyquist limit (i.e., below one-half the sampling rate) whose primary function is to eliminate signal frequency components that could cause *aliasing*.[6]

Sampling The *sampling* block samples pixel values along a horizontal line at a sampling rate that is standardized for the main video formats as follows:

- NTSC $4f_{sc}$ (composite): 114.5 Mbps
- PAL $4f_{sc}$ (composite): 141.9 Mbps
- Rec. 601 (component) (luma): 108 Mbps
- Rec. 601 (component) (chroma): 54 Mbps

For component systems, the rates standardized by Rec. 601 were chosen to represent a sampling rate of 13.5 MHz, which is an integral multiple of both NTSC and PAL line rates. For composite signals, a sampling rate of $4f_{sc}$ (where f_{sc} is the frequency of the color subcarrier) exceeds the Nyquist limit by a comfortable margin and has become part of the SMPTE standard 244M.

Figure 20.10 shows the location of sampling points on individual lines for the Rec. 601 format. The luma and two color-difference sampling pulses are synchronized so that the color difference points are *cosited* with alternate luminance points. For composite signals, the sampling points follow a specified relationship with the phase of the color sync burst. For both NTSC and PAL at $4f_{sc}$, there are four sampling points for each cycle of the burst: for PAL, the sampling points are at 0°, 90°, 180°, and 270° points of the burst waveform. For NTSC, the first sample is located at 57° (the I axis), the second at 147° (the Q axis), and so on.

The sampling rate and phase must be synchronized with the line and subcarrier frequencies and phases to maintain these precise locations. This task is performed by the *sampling clock generator*.

[6]The concept of *aliasing* was introduced in Section 5.4.1.

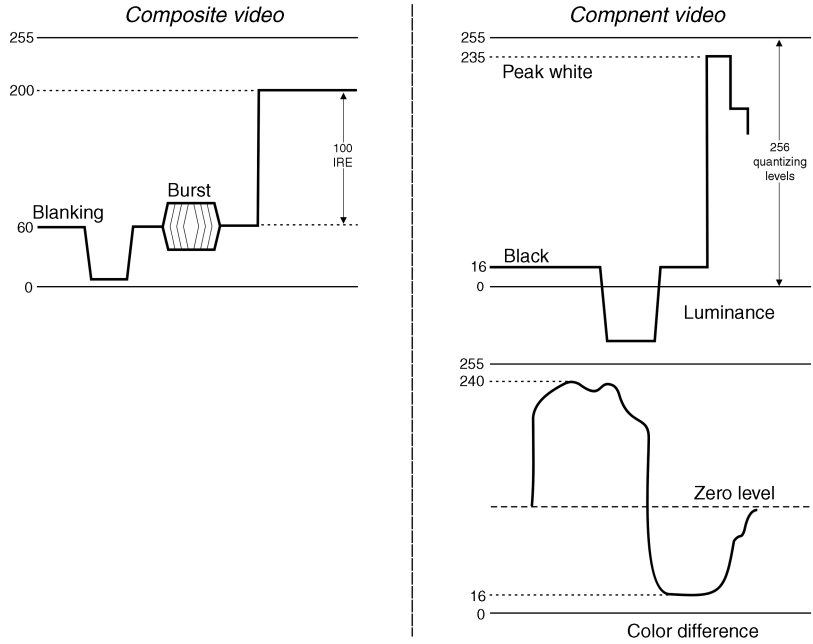

FIGURE 20.11 Assignment of quantization levels for component and composite video. Redrawn from [LI99].

Quantizing The next step in the ADC process is to quantize the samples using a finite number of bits per sample. This is achieved by dividing the amplitude range into discrete intervals and establishing a *quantum level* for each interval. The difference between the quantum level and the analog signal at the sampling point is called the *quantizing error*.

Most video quantization systems are based on 8-bit words, that is, 256 possible discrete values per sample (1024 levels are sometimes used for pregamma signals in cameras). Not all levels may be used for the video range because of the need to use some of the values for data or control signals.

Figure 20.11 shows how the resulting 256 quantized levels are typically used for composite and component video signals. In composite systems, the entire composite signal, including sync pulses, is transmitted within the quantized range of levels 4–200. Luma occupies levels 16–235. Since color-difference signals can be positive or negative, the zero level is placed at the center of the quantized range. In both cases, the values 0 and 255 are reserved for synchronization codes.

Encoding The final step in the ADC process is the encoding of the quantized levels of the signal samples. This is a topic that has been explored extensively, especially in connection with video compression (Section 20.9).

20.7 COLOR REPRESENTATION AND CHROMA SUBSAMPLING

A monochrome video frame can be represented using just one (typically 8-bit) value per spatiotemporal sample. This number usually indicates the gamma-corrected luminance information at that point, which we call *luma*: the larger the number, the brighter the pixel.

Color video requires multiple (usually three) values per sample. The meaning of each value depends on the adopted color model. The most common color representation for digital video adopts the $Y'CrCb$ color model. In this format, one component represents luma (Y'), while the other two are color-difference signals: Cb (the difference between the blue component and a reference value) and Cr (the difference between the red component and a reference value). $Y'CbCr$ is a scaled and offset version of the $Y'UV$ color space.

The exact equations used to calculate Y', Cr, and Cb vary among standards (see Chapter 3 of [Jac01] for details). For the sake of illustration, the following are the equations for the Rec. 601 SDTV standard:

$$Y'_{601} = 0.299R' + 0.587G' + 0.114B' \tag{20.13}$$

$$Cb = -0.172R' - 0.339G' + 0.511B' + 128 \tag{20.14}$$

$$Cr = 0.511R' - 0.428G' - 0.083B' + 128 \tag{20.15}$$

The key advantage of $Y'CrCb$ over RGB is that the Cr and Cb components may be represented with a lower spatial resolution than Y' because the HVS is less sensitive to variations in color than in luminance. This reduces the amount of data required to represent the chroma components, without significantly impacting the resulting visual quality. This process is known as *chroma subsampling*.

Figure 20.12 shows three of the most common patterns for chroma subsampling. The numbers indicate the relative sampling rate of each component in the *horizontal* direction. 4:4:4 means that the three components (Y', Cr, and Cb) have the same resolution; that is, there is a sample of each component at every pixel position. Another way of explaining it is to say that for every four luma samples, there are four Cr

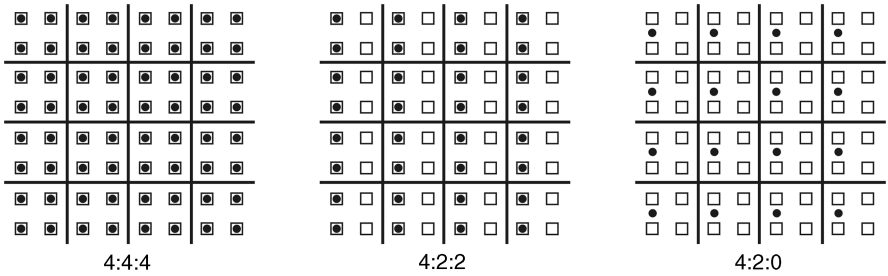

FIGURE 20.12 The most common chroma subsampling patterns: 4:4:4, 4:2:2, and 4:2:0.

and four *Cb* samples. The 4:4:4 sampling pattern preserves the full fidelity of the chroma components. In 4:2:2 sampling, the chroma components have the same vertical resolution but only half the horizontal resolution; that is, for every four luma samples in the horizontal direction, there are two *Cr* and two *Cb* samples. The 4:2:0 pattern corresponds to the case where both *Cr* and *Cb* have half the horizontal and the vertical resolution of Y'. The 4:2:0 pattern requires half as many samples—and, consequently, half as many bits—as the 4:4:4 video.

When chroma subsampling is used, the encoder discards selected color-difference samples after filtering. At the decoder side, these missing samples are approximated by interpolation. The most common interpolation scheme consists of using averaging filters whose size and shape vary according to the chroma subsampling pattern used at the encoder side.

■ EXAMPLE 20.2

Calculate the number of bits per frame required to encode a color Rec. 601 625/50 (720×480 pixels per frame) video, using the following chroma subsampling patterns: 4:4:4, 4:2:2, and 4:2:0.

Solution

The key to this problem is to calculate the equivalent number of bits per pixel (N_b) for each case. Once the value of N_b is obtained, it is a simple matter of multiplying that value by the frame height and width.

(a) $N_\mathrm{b} = 8 \times 3 = 24$. The total number of bits is $720 \times 480 \times 24 = 8{,}294{,}400$.

(b) Since for every group of four pixels, eight samples are required, $N_b = 8 \times 8/4 = 16$. The total number of bits is $720 \times 480 \times 16 = 5{,}529{,}600$.

(c) Since for every group of four pixels, six samples are required, $N_\mathrm{b} = 8 \times 6/4 = 12$. The total number of bits is $720 \times 480 \times 12 = 4{,}147{,}200$.

20.8 DIGITAL VIDEO FORMATS AND STANDARDS

The topic of digital video formats and standards is a broad and ever-changing one about which many entire books have been written. The enormous amount of formats and standards—and the myriad of technical details within each and every one of them—can easily overwhelm the reader. In this section, we present a brief summary of relevant digital video formats and standards. This list will be expanded in Section 20.9 when we incorporate (an even larger number of) standards for compressed video.

Digital video formats vary according to the application. For digital video with quality comparable to analog SDTV, the most important standard is the ITU-R (formerly CCIR) Recommendation BT.601-5, 4:2:2[7] (Section 20.8.1). For many video coding applications—particularly in video conferencing and video telephony—a family of

[7]For most of this chapter, we refer to this format simply as *Rec. 601*.

TABLE 20.2 Representative Digital Video Formats

Format (Application)	Y' size ($H \times V$)	Color Sampling	Frame Rate	Raw Data (Mbps)
QCIF (video telephony)	176 × 144	4:2:0	30p	9.1
CIF (videoconference)	352 × 288	4:2:0	30p	37
SIF (VCD, MPEG-1)	352 × 240 (288)	4:2:0	30p/25p	30
Rec. 601 (SDTV distribution)	720 × 480 (576)	4:2:0	60i/50i	124
Rec. 601 (video production)	720 × 480 (576)	4:2:2	60i/50i	166
SMPTE 296M (HDTV distribution)	1280 × 720	4:2:0	24p/30p/60p	265/332/664
SMPTE 274M (HDTV distribution)	1920 × 1080	4:2:0	24p/30p/60i	597/746/746

Reproduced from [WOZ02].

intermediate formats is used (Section 20.8.2). For digital television, different standards are adopted in different parts of the world. Table 20.2 provides a sample of relevant digital video formats, ranging from low bit rate QCIF for video telephony to high-definition SMPTE 296M and SMPTE 274M for HDTV broadcast.

20.8.1 The Rec. 601 Digital Video Format

The ITU-R Recommendation BT.601-5 is a digital video format widely used for television production. The luma component of the video signal is sampled at 13.5 MHz (which is an integer multiple of the line rate for both 50i and 60i standards) and the chroma at 6.75 MHz to produce a 4:2:2 $Y'CrCb$ component signal. The parameters of the sampled digital signal depend on the video frame rate (25 or 30 fps) and are shown in Table 20.3. It can be seen that the higher (30 fps) frame rate is compensated by a lower resolution, so the total bit rate is the same in both cases (216 Mbps). Figure 20.13 shows the resulting *active area*—720 × 480 and 720 × 576—for each case as well as the number of pixels left out to make up for horizontal and vertical blanking intervals (shaded portion of the figure).

The Rec. 601 formats are used in high-quality (standard definition) digital video applications. The 4:4:4 and 4:2:2 are typically used for video production and editing, whereas the 4:2:0 variant is used for video distribution, whether on DVDs, video on demand, or other format. The MPEG-2 compression standard was primarily developed for compression of Rec. 601 4:2:0 signals, although it has been made flexible enough to also handle video signals in lower and higher resolutions. The typical compression ratio achieved by MPEG-2-encoded Rec. 601 4:2:0 videos allows a reduction in data rate from 124 Mbps to about 4–8 Mbps.

TABLE 20.3 ITU-R Recommendation BT.601-5 Parameters

Parameters	525 lines, 30 (29.97) fps	625 lines, 25 fps
Fields Per Second	60 (59.94)	50
Luma channel		
Bandwidth (MHz)	5.5	5.5
Sampling frequency (MHz)	13.5	13.5
Number of samples per line	858	864
Number of samples per active line	720	720
Bits per sample	8	8
Bit rate (Mbps)	108	108
Chroma (color-difference) channels		
Bandwidth (MHz)	2.2	2.2
Sampling frequency (MHz)	6.75	6.75
Number of samples per line	429	432
Number of samples per active line	355	358
Bits per sample	8	8
Bit rate (Mbps)	54	54
Total bit rate (Mbps)	216	216

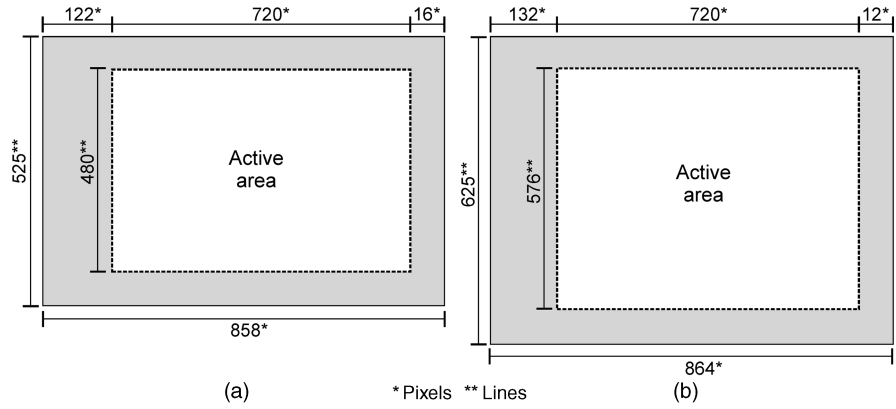

FIGURE 20.13 Rec. 601 format: screen dimensions and active area for the 525/59.94/2:1 (a) and 625/50/2:1 (b) variants. Redrawn from [WOZ02].

20.8.2 The Common Intermediate Format

For video coding applications, video is often converted into an "intermediate format" before compression and transmission. The names and frame sizes for the common intermediate format (CIF) family are presented in Table 20.4. The CIF format has about half of the horizontal and the vertical resolution of the Rec. 601 4:2:0. It was primarily developed for videoconference applications. The QCIF format, whose resolution is half of that of CIF in either dimension—hence the prefix *quarter*—is used for videophone and similar applications. Both are noninterlaced.

TABLE 20.4 Intermediate Formats

Format	Luma Resolution ($H \times V$)
Sub-QCIF	128×96
Quarter CIF (QCIF)	176×144
CIF	352×288
4 CIF	704×576

20.8.3 The Source Intermediate Format

The SIF source intermediate format is an ISO-MPEG format whose characteristics are about the same as CIF. It was targeted at video applications with medium quality such as Video CD (VCD) and video games. There are two variants of SIF: one with a frame rate of 30 fps and a line number of 240 and the other with a frame rate of 25 fps and a line number of 288. Both have 352 pixels per line. There is also a corresponding set of SIF-I (2:1 interlaced) formats. SIF files are often used in connection with the MPEG-1 compression algorithm that can reduce the necessary data rate to transmit them from 30 to approximately 1.1 Mbps with a visual quality comparable to a standard VHS VCR tape (which is lower than broadcast SDTV). MPEG-1-based VCDs have become all but obsolete with the popularization of MPEG-2-based DVDs. This story might take yet another turn with the popularization of *Blu-Ray* high-definition DVDs.

20.9 VIDEO COMPRESSION TECHNIQUES AND STANDARDS

In this section, we present a brief and broad overview of video compression principles, techniques, and standards. As anticipated in Chapter 17, video compression techniques exploit a type of redundancy not available in the case of image compression, but is quite intuitive for the case of video sequences, based on the fact that consecutive frames in time are usually very similar. This is often called interframe (or *temporal*) redundancy.

The simplest way to leverage the similarity between consecutive frames and save bits while encoding a video sequence is to use *predictive coding* techniques. The basic idea behind predictive coding is to predict the contents of frame $k + 1$ based on frame k, compute the difference between the predicted and the actual frame, and encode that difference. The simplest predictor would be one that claims that frame $k + 1$ is identical to frame k, and the simplest measure of difference would be the subtraction of one frame from the other. For certain video sequences (e.g., sequences with fixed background and little foreground activity), this naive approach may work surprisingly well. After all, regardless of the entropy-based technique used to convert the differences into bits, the frame differences are likely to be small (i.e., a significant entropy reduction will be obtained) and the bit savings will be significant.

In practice, predictive video coding techniques do better than that: they use block-based motion estimation (ME) techniques (described in Chapter 22) to estimate

motion vectors containing the amount and direction of motion of blocks of pixels in the frame, compute a predicted frame based on the ME results, and encode the errors (between the actual pixel values and the predicted ones). Once again, since the differences are significantly smaller than the actual pixel motion vectors' values, the bit savings will be substantial. Moreover, the method will work for videos with virtually any amount or type of motion.

There is no universally accepted taxonomy for video compression and coding techniques. For the purposes of this book, we will classify them into four major groups:

- *Transform Based*: This group includes techniques that leverage the properties of mathematical transforms, particular the discrete cosine transform (DCT).
- *Predictive*: Techniques that encode the error terms obtained by comparing an actual frame with a predicted version of that frame based on one or more of its preceding (and succeeding) frames.
- *Block-Based Hybrid*: Combine transform-based techniques (applied to nonoverlapping blocks in each frame) with predictive methods (using motion compensated predictions of frame contents based on adjacent frames). Used in several influential video coding standards, such as H.261, H.263, MPEG-1, and MPEG-2.
- *Advanced*: In this group, we include techniques based on 2D shape and texture, as well as scalable, region-based, and object-based video coding techniques used in standards such as MPEG-4 and H.264.

20.9.1 Video Compression Standards, Codecs, and Containers

Numerous video compression standards have been proposed during the past 20 years. Some have been widely adopted and incorporated into consumer electronics products and services, while some others were little more than an academic exercise. Navigating the landscape of video coders and decoders (or *codecs* for short) can be a daunting task: there are too many standards, their scope and applications often overlap, their availability ranges from proprietary to open source, the terminology is not always consistent, and their relevance can be fleeting. To compound the problem a bit further, end users normally manipulate video files using *containers* (e.g., AVI or MOV) mapping to different subsets of codecs, which cannot simply be determined by the file extension and may not be widely supported by different video players.

To help you make sense of this picture and adopt a consistent terminology, we will refer to a *standard* as a collection of official documents usually issued by an international organization, a *codec* as a software implementation of one or more standards, and a *container* as a file format that acts as a wrapper around the encoded audiovisual data, additional information (e.g., subtitles), and associated metadata. Here is an overview of relevant video coding and compression standards, codecs, and containers at the time of this writing.

MPEG Standards The MPEG has overseen the development of video coding standards for consumer applications. The best-known representatives of the MPEG family of video compression standards are MPEG-1 (for VCD quality videos), MPEG-2 (which became a standard for digital broadcast TV), and MPEG-4 (which is significantly more complex than its predecessors and covers a broad range of applications).

ITU-T Standards The International Telecommunications Union (ITU-T) regulated the standardization of video formats for telecommunication applications, for example, video telephony and videoconferencing over conventional telephone lines. The most prominent results of those efforts were the H.261 standard for video telephony, the H.263 standard (and its variants) for videoconferencing, and the H.264 standard, which would become—in a joint effort with the MPEG-4 Part 10, or MPEG-4 AVC (for *Advanced Video Coding*) standard—the most successful video coding standard of the early twenty-first century.

Open Source Codecs There are many free software libraries containing open source implementations of popular video coding standards available online, such as x264 (for H.264/MPEG-4 AVC), Xvid (for MPEG-4), and FFmpeg (for many formats and standards).

Proprietary Codecs Proprietary codecs include DivX (for MPEG-4), Sorenson (used in Apple's QuickTime and Adobe Flash), Microsoft's Windows Media Video (WMV) (which is also the name of a container), and RealNetworks's RealVideo.

Popular Video Containers The most popular video containers in use today are the 3GP (for 3G mobile phones), Microsoft's Advanced Systems Format (.asf, .wma, .wmv), Microsoft's AVI, the DivX Media Format, Adobe Systems' Flash Video (whose notable users include YouTube, Google Video, and Yahoo! Video), MP4 (MPEG-4 Part 14), MPEG video file (.mpg, .mpeg), and Apple's Quicktime (.mov, .qt).

20.10 VIDEO PROCESSING IN MATLAB

MATLAB provides the necessary functionality for basic video processing using short video clips and a limited number of video formats. Not long ago, the only video container supported by built-in MATLAB functions was the AVI container, through functions such as `aviread`, `avifile`, `movie2avi`, and `aviinfo` (explained in more detail later). Moreover, the support was operating system dependent (on UNIX platforms, only uncompressed AVI files are supported) and limited only to a few codecs. Starting with MATLAB Version 7.5 (R2007b), a new library function (`mmreader`) was added to extend video support to formats such as AVI, MPEG, and WMV in a platform-dependent way: Windows machines can be used to read AVI (.avi), MPEG-1 (.mpg), Windows Media Video (.wmv, .asf, .asx), and any format supported by Microsoft DirectShow, whereas Mac users can employ it to read

AVI (.avi), MPEG-1 (.mpg), MPEG-4 (.mp4, .m4v), Apple QuickTime Movie (.mov), and any format supported by QuickTime.

MATLAB's ability to handle matrices makes it easy to create and manipulate 3D or 4D data structures to represent monochrome and color video, respectively, provided that the video sequences are short (no more than a few minutes worth of video). Moreover, once a frame needs to be processed individually, it can be converted to an image using the `frame2im` function, which can then be processed using any of the functions available in the Image Processing Toolbox (IPT).

Finally, it is worth mentioning that another MathWorks product, Simulink, contains a Video and Image Processing Blockset[8] that can be integrated with MATLAB and its closest toolboxes, particularly the IPT and the Image Acquisition Toolbox (IAT).

20.10.1 Reading Video Files

The MATLAB functions associated with reading video files are as follows:

- `aviread`: reads an AVI movie and store the frames into a MATLAB `movie` structure.
- `aviinfo`: returns a structure whose fields contain information (e.g., frame width and height, total number of frames, frame rate, file size, etc.) about the AVI file passed as a parameter.
- `mmreader`: constructs a multimedia reader object that can read video data from a variety of multimedia file formats.

You will learn how to use these functions in Tutorial 20.1.

20.10.2 Processing Video Files

Processing video files usually consist of the following steps (which can embedded in a `for` loop if the same type of processing is to be applied to all frames in a video):

1. Convert frame to an image using `frame2im`.
2. Process the image using any technique such as the ones described in Part I.
3. Convert the result back into a frame using `im2frame`.

20.10.3 Playing Video Files

The MATLAB functions associated with playing back video files are as follows:

- `movie`: primitive built-in video player
- `implay`: fully functional image and video player with VCR-like capabilities

You will learn how to use these functions in Tutorial 20.1. (page 528).

[8] A *blockset* in Simulink is equivalent to a *toolbox* in MATLAB.

20.10.4 Writing Video Files

The MATLAB functions associated with writing video files are as follows:

- `avifile`: creates a new AVI file that can then be populated with video frames in a variety of ways.
- `movie2avi`: creates an AVI file from a MATLAB `movie`.

You will learn how to use some of these functions in Tutorial 20.1.

20.11 TUTORIAL 20.1: BASIC DIGITAL VIDEO MANIPULATION IN MATLAB

Goal

The goal of this tutorial is to learn how to read and view video data in MATLAB, as well as extract and process individual frames.

Objectives

- Learn how to gather video file information using the `aviinfo` function.
- Learn how to read video data into a variable using the `aviread` function.
- Explore the `montage` function for viewing multiple frames simultaneously.
- Learn how to play a video using the `movie` function and the `implay` movie player.
- Learn how to convert from frame to image and vice versa using the `frame2im` and `im2frame` functions.
- Explore techniques for assembling images into video, including the `immovie` function.
- Learn how to write video data to a file using the `movie2avi` function.
- Learn how to read and play video files in different formats using the `mmreader` function.

What You Will Need

- Test files `original.avi` and `shopping_center.mpg`.

Procedure

We will start by learning how to use built-in functions to read information about video files and load them into the workspace. The `aviinfo` function takes a video file name as its parameter and returns information about the file, such as compression and number of frames.

Reading Information About Video Files

1. Read information about the `original.avi` file and save it in a local variable.

```
file_name = 'original.avi';
file_info = aviinfo(file_name);
```

2. View the video compression and the number of frames for this file.

```
file_info.VideoCompression
file_info.NumFrames
```

Question 1 What other information does the `aviinfo` function provide?

Question 2 Try viewing information for another AVI video file. What parameters are different for the new file?

Reading a Video File

The function `aviread` allows us to load an AVI file into the MATLAB workspace. The data are stored as a structure, where each field holds information for each frame.

3. Load the example.avi file using the `aviread` function.

```
my_movie = aviread(file_name);
```

Question 3 What is the size of the `my_movie` structure?

We can also load individual frames from a video file by specifying the frame numbers as a second parameter.

4. Load frames 5, 10, 15, and 20.

```
frame_nums = [5 10 15 20];
my_movie2 = aviread(file_name, frame_nums);
```

Question 4 What is the size of `my_movie2` structure?

Viewing Individual Frames

When we use `aviread` to load a movie file, each element of the structure holds information for that particular frame. This information is stored in two fields: `cdata`, which is the actual image data for that frame, and `colormap`, which stores the color map for the `cdata` field when the image type is indexed. If the image is truecolor, then the `colormap` field is left blank.

5. Inspect the first frame of the `my_movie` structure.

```
my_movie(1)
```

6. View the first frame as an image using the `imshow` function.

```
imshow(my_movie(1).cdata)
```

We can view all the frames simultaneously using the `montage` function. This function will display images in an array all at once in a grid-like fashion.

7. Preallocate a 4D array that will hold the image data for all frames.

```
image_data = uint8(zeros(file_info.Height, file_info.Width, 3, ...
    file_info.NumFrames));
```

8. Populate the `image_data` array with all the image data in `my_movie`.

```
for k = 1:file_info.NumFrames
    image_data(:,:,:,k) = my_movie(k).cdata;
end
```

9. Use the `montage` function to display all images in a grid.

```
montage(image_data)
```

Question 5 Explain how the data for each frame are stored in the `image_data` variable.

Playing a Video File

The function `movie` will play video data.

10. Play the video with default settings.

```
movie(my_movie)
```

Question 6 What is the default frame rate when playing a video?

11. Play the video five times with a frame rate of 30 fps.

```
movie(my_movie, 5, 30)
```

12. Play only frames 1–10.

```
frames = [5 1:10];
movie(my_movie, frames, 30)
```

Question 7 How many times will this movie play?

Question 8 At what frame rate will this movie play?

As you have probably noticed, the `movie` function has very limited functionality, with no support for simple operations such as pausing and stepping through frames. To make video analysis easier, we will use the `implay` function.

13. Play the movie with the `implay` function.

```
implay(my_movie)
```

The `implay` function opens a movie player with VCR-like controls and several familiar options available on other video players, for example, Apple QuickTime or Microsoft Windows Media Player.

Question 9 Explore the user interface of the movie player. How can we specify the frame rate of playback?

Question 10 How do we play back the movie in a continuous loop?

Processing Individual Frames

To perform image processing operations on individual frames, we can use the `frame2im` function, which will convert a specified frame to an image. Once we have this image, we can use familiar image processing tools such as the ones described in Part I.

14. Convert frame 10 to an image for further processing.

```
old_img = frame2im(my_movie(10));
```

15. Blur the image using an averaging filter and display the result.

```
fn = fspecial('average',7);
new_img = imfilter(old_img, fn);
figure
subplot(1,2,1), imshow(old_img), title('Original Frame');
subplot(1,2,2), imshow(new_img), title('Filtered Frame');
```

16. Using another frame, create a negative and display the result.

```
old_img2 = frame2im(my_movie(15));
image_neg = imadjust(old_img2, [0 1], [1 0]);
figure
subplot(1,2,1), imshow(old_img2), title('Original Frame');
subplot(1,2,2), imshow(image_neg), title('Filtered Frame');
```

Now that we have processed our image, we can convert it back to frame using the `im2frame` function.

17. Convert the images back to frames and save the new frames in the video structure.

```
my_movie2 = my_movie;
new_frame10 = im2frame(new_img);
new_frame15 = im2frame(image_neg);
my_movie2(10) = new_frame10;
my_movie2(15) = new_frame15;
```

Question 11 Use `implay` to view the new video with the two processed frames. Which of the two is more noticeable during playback?

A straightforward way to perform processing on all frames is to use a `for` loop.

18. Create a negative of all the frames and reconstruct the frames into a video.

```
movie_neg = my_movie;
for k = 1:file_info.NumFrames
    cur_img = frame2im(movie_neg(k));
    new_img = imadjust(cur_img, [0 1], [1 0]);
    movie_neg(k) = im2frame(new_img);
end
implay(movie_neg)
```

If you have an array of image data, another way to construct a video structure is through the `immovie` function. This function will take an array of image data and return a video structure.

19. Create an array of negative images from the original movie.

```
my_imgs = uint8(zeros(file_info.Height, file_info.Width,3, ...
    file_info.NumFrames));
for i = 1:file_info.NumFrames
    img_temp = frame2im(my_movie(i));
    my_imgs(:,:,:,i) = imadjust(img_temp, [0 1], [1 0]);
end
```

20. Now construct a video structuring using the `immovie` function.

```
new_movie = immovie(my_imgs);
implay(new_movie);
```

Question 12 How does this procedure differ from using the `im2frame` function in a loop? Is it beneficial?

Writing to a Video File

To write movies, that is, image sequences, to a video file, we use the `movie2avi` function. Note that in the following steps an AVI file will be created in your current directory, so make sure you have the permissions to do so.

21. Set the file name of the new movie to be created on disk.

```
file_name = 'new_video.avi';
```

22. Create the AVI file.

```
movie2avi(new_movie, file_name, 'compression', 'None');
```

23. Read and play the resulting AVI file.

```
my_movie3 = aviread(file_name);
implay(my_movie3);
```

Reading and Playing Video Files in Different Formats

24. Use the sequence below to read and play the first 100 frames of an MPEG movie.

```
obj = mmreader('shopping_center.mpg');
video = read(obj, [1 100]);
frameRate = get(obj,'FrameRate')
implay(video, frameRate)
```

Question 13 What are the frame rate, frame width, frame height, and total number of frames in file `shopping_center.mpg`?

FIGURE 20.14 Visual representation of a YUV file.

20.12 TUTORIAL 20.2: WORKING WITH YUV VIDEO DATA

Goal

The goal of this tutorial is to learn how to read, process, and display YUV video data[9] in MATLAB.

Objectives

- Explore the contents of a YUV video file.
- Explore the `readYUV` function used to read in YUV video data.

What You Will Need

- Test files `miss_am.yuv`, `foreman.yuv`, and `whale_show.yuv`
- The `readYUV.m` script

Procedure

A YUV video file contains raw video data stored as separate components: the luminance component Y is first, followed by the U and V chrominance components (Figure 20.14). Owing to chroma subsampling (Section 20.7), each U and V component contains only $1/4$ of the data that Y does.

For the following steps, note that the `miss_am.yuv` video contains 30 frames, each formatted as `QCIF_PAL`, that is, 176×144 pixels.

1. Ensure that your current directory contains the file `miss_am.yuv`.

Question 1 Based on the format and number of frames in the video, calculate the size of the file (in kB).

Question 2 Does this value coincide with what MATLAB displays in the *Current Directory* pane?

YUV files do not use headers to store any information about the video itself, for example, frame size, frame rate, or color standard. Consequently, you must know in advance what those parameters are and use them whenever needed. Once we obtain this information, we can parse the video file, reading the proper data for each of

[9] YUV video files are very common in video processing research and will also be used in the following chapters.

the components and then converting it to a MATLAB video structure for further manipulation.

To read in a YUV file, we will use the `readYUV` function[10] and the test file `miss_am.yuv`.

2. Define all relevant data before calling the `readYUV` function.

```
file_name = 'miss_am.yuv';
file_format = 'QCIF_PAL';
num_of_frames = 30;
```

3. Read the video data and display the movie using `implay`.

```
[yuv_movie, yuv_array] = readYUV(file_name, num_of_frames, ...
    file_format);
implay(yuv_movie)
```

Question 3 Describe the contents and record the dimensions of the `yuv_movie` variable.

Question 4 Describe the contents and record the dimensions of the `yuv_array` variable. How is this variable different from `yuv_movie`?

4. Try to read and play the `foreman.yuv` and `whale_show.yuv` video files.

```
file_name = 'foreman.yuv';
file_format = 'QCIF_PAL';
num_of_frames = 25;
[yuv_movie, yuv_array] = readYUV(file_name, num_of_frames, ...
    file_format);
implay(yuv_movie)

file_name = 'whale_show.yuv';
file_format = 'NTSC';
num_of_frames = 25;
[yuv_movie, yuv_array] = readYUV(file_name, num_of_frames, ...
    file_format);
implay(yuv_movie)
```

Question 5 After reading both these files, show how do their sizes compare and why?

[10]This function was developed by Jeremy Jacob and can be downloaded from the book web site.

Question 6 When using the `readYUV` function, what happens if you specify more frames than there actually are in the file?

Question 7 What happens if you specify fewer frames than there are in the file?

WHAT HAVE WE LEARNED?

- Video is the electronic representation of visual information whose spatial distribution of intensity values varies over time.
- The video signal is a 1D analog or digital signal varying over time in such a way that the spatiotemporal information is ordered according to a predefined scanning convention that samples the signal in the vertical and temporal dimensions.
- An analog video raster is a fixed pattern of parallel scanning lines disposed across the image. A raster's main parameters include the line rate, the frame interval, the line spacing, the line interval, and the number of active lines.
- Some of the most important concepts and terminologies associated with (analog) video processing include
  - *Aspect Ratio*: the ratio of frame width to height.
  - *Vertical Resolution*: the number of horizontal black and white lines in the image (vertical detail) that can be distinguished or resolved in the picture height; it is a function of the number of scanning lines per frame.
  - *Horizontal Resolution*: the number of vertical lines in the image (horizontal detail) that can be distinguished in a dimension equal to the picture height; it is determined by the signal bandwidth in analog systems.
  - *Progressive Scanning*: the process by which each image is scanned in one single pass called *frame*.
  - *Interlaced Scanning*: the process by which each frame is scanned in two successive vertical passes, first the odd numbered lines and then the even numbered ones. Each pass is called a *field*.
  - *Blanking Interval*: the time interval at the end of each line (horizontal retrace) or field (vertical retrace) during which the video signal must be blanked before a new line or field is scanned.
  - *Component Video*: an analog video representation scheme that uses three 1D color component signals.
  - *Composite Video*: an analog video representation scheme that combines luminance and chrominance information into a single composite signal.
- Gamma correction is a process by which the nonlinearity of an image acquisition or display device is precompensated (on the transmitter's side) in such a way as to ensure the display of the correct dynamic range of color values within the signal.
- Color information is encoded in analog video systems in a backward-compatible way. The (gamma-corrected) luma signal and the three (gamma-corrected)

primary color channels are combined into color-difference signals that are then used to modulate a color subcarrier. The frequency of the color subcarrier is carefully chosen so that its spectral components fit within the gaps of the existing (monochrome) spectrum.

- The most popular analog SDTV and video standards used worldwide are NTSC and PAL (with SECAM in a distant third place). Although they share many common ideas (interlaced scanning, QAM color modulation, etc.), these standards use different values for most of the main raster parameters (such as lines per frame and frames per second).
- A digital video is a sampled two-dimensional (2D) version of a continuous three-dimensional (3D) scene. Digital video not only employs vertical and temporal sampling—similar to analog video—but also includes horizontal sampling, that is, sampling of pixel values along a line.
- The main parameters that characterize a digital video sequence are the frame rate, the line number, and the number of samples per line. The product of these three parameters times the average number of bits per pixel provides an estimate of the data rate needed to transmit that video in its uncompressed form.
- Video digitization, or analog-to-digital conversion, involves four main steps: (1) antialiasing filter—an optional low-pass filter used to eliminate signal frequency components that could cause aliasing; (2) sampling—pixel values are sampled along a horizontal line at a standardized rate; (3) quantizing—the samples are represented using a finite number of bits per sample; and (4) encoding—the quantized samples are converted to binary codewords.
- Chroma subsampling is the process of using a lower spatial resolution, and consequently fewer bits, to represent the color information of a digital video frame.
- Some of the most popular contemporary digital video formats are QCIF, CIF, SIF, Rec. 601, SMPTE 296M, and SMPTE 274M.
- MATLAB can be used to read, process, and play back digital video files in several different formats.

LEARN MORE ABOUT IT

- The following is a list of selected books on video processing and related fields:
  - Bovik, A. (ed.), *Handbook of Image and Video Processing*, San Diego, CA: Academic Press, 2000.
  - Grob, B. and Herndon, C. E., *Basic Television and Video Systems*, 6th ed., New York: McGraw-Hill, 1999.
  - Haskell, B. G., Puri, A., and Netravali, A. N., *Digital Video: an Introduction to MPEG-2*, Norwell, MA: Kluwer Academic Publishers, 1997.
  - Jack, K., *Video Demystified: A Handbook for the Digital Engineer*, 3rd ed., Eagle Rock, VA: LLH Technology Publishing, 1993.

- Luther, A. C. & Inglis, A. F., *Video Engineering*, 3rd ed., New York: McGraw-Hill, 2000.
- Poynton, C., *Digital Video and HDTV Algorithms and Interfaces*, San Francisco, CA: Morgan Kaufmann, 2003.
- Poynton, C., *A Technical Introduction to Digital Video*, New York: Wiley, 1996.
- Robin, M. and Poulin, M., *Digital Television Fundamentals: Design and Installation of Video and Audio Systems*, 2nd ed., New York: McGraw-Hill, 2000.
- Tekalp, A. M., *Digital Video Processing*, Upper Saddle River, NJ: Prentice Hall, 1995.
- Wang, Y., Ostermann, J., and Zhang, Y.-Q., *Video Processing and Communications*, Upper Saddle River, NJ: Prentice-Hall, 2002.
- Watkinson, J., *The Art of Digital Video*, 3rd ed., Oxford: Focal Press, 2000.
- Whitaker, J. C. and Benson, K. B. (Ed.), *Standard Handbook of Video and Television Engineering*, 3rd ed., New York: McGraw-Hill, 2000.
- Woods, J. W., *Multidimensional Signal, Image, and Video Processing and Coding*, San Diego, CA: Academic Press, 2006.

- The book by Grob and Herndon [GH99] provides a broad and detailed coverage of analog TV and video systems.
- Chapter 2 of [RP00] and Chapters 8 and 9 of [Jac01] describe analog video standards in great detail.
- For more on gamma correction, please refer to Chapter 23 of [Poy03] and Section 5.7 of [BB08].
- Chapter 3 of [RP00] has a very detailed explanation of sampling, quantization, and (component and composite) digital video standards from a TV engineering perspective.
- Chapter 10 of [GH99] explains in detail the contents of VBI in analog TV systems.
- Chapter 13 of [WOZ02] provides a good overview of video compression standards.
- Chapter 2 of [Wat00] contains a broad overview of video principles.

ON THE WEB

- World TV Standards
  http://www.videouniversity.com/standard.htm
- Advanced Television Systems Committee (ATSC): American standards
  http://www.atsc.org/
- Digital Video Broadcasting Project (DVB): European standards
  http://www.dvb.org/

- Society of Motion Picture and Television Engineers (SMPTE)
  http://www.smpte.org/home
- The MPEG home page
  http://www.chiariglione.org/mpeg/
- MPEG Industry Forum
  http://www.mpegif.org/
- Test images and YUV videos—Stanford University
  http://scien.stanford.edu/pages/labsite/scien_test_images_videos.php
- YUV 4:2:0 video sequences—Arizona State University
  http://trace.eas.asu.edu/yuv/index.html
- Test video clips (with ground truth) from the CAVIAR project
  http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/
- Charles Poynton's "Gamma FAQ"
  http://www.poynton.com/GammaFAQ.html
- Adobe digital audio and video primers
  http://www.adobe.com/motion/primers.html
- Keith Jack's blog
  http://keithjack.net/

20.13 PROBLEMS

20.1 Explain in your own words the differences between composite and component video systems in terms of bandwidth, historical context, and image quality.

20.2 The human visual system is much more sensitive to changes in the brightness of an image than to changes in color. In other words, our color spatial resolution is poor compared to the achromatic spatial resolution.

Knowledge of this property has been somehow embedded in the design of both analog and digital video systems. Answer the following questions:

(a) How did this property get exploited in analog TV systems?

(b) How did it get factored into the design of digital video formats?

20.3 Provide an objective explanation as to *why* the luminance signal in PAL/NTSC systems is calculated as $Y = 0.299R + 0.587G + 0.114B$, and *not* $Y = (R + G + B)/3$.

20.4 Calculate the raw data rate of a digital video signal with the following characteristics:

- 2:1 interlaced scanning
- 352×240 luminance samples per frame
- 30 frames per second
- 4:2:2 chroma subsampling

20.5 Regarding progressive and interlaced scanning methods,

(a) What are the pros and cons of each method?

(b) For the same line number per frame, what is the relation between the maximum temporal frequency that a progressive raster can have and that of an interlaced raster that divides each frame into two fields?

(c) What about the relation between the maximum vertical frequencies?

20.6 Which considerations would you use to determine the frame rate and line number when designing a video capture or display system?

20.7 Why does a computer monitor use a higher temporal refresh rate and line number than the one adopted by a typical TV monitor?

20.8 Regarding NTSC and PAL color TV systems,

(a) What do Y, I, and Q stand for in NTSC?

(b) What do Y, U, and V stand for in PAL?

(c) How are I and Q related to U and V?

(d) What is the relationship between NTSC's Y, PAL's Y, and CMY(K) color model's Y?

20.9 Write a MATLAB script to

(a) read an RGB color image and convert it to $Y'CrCb$;

(b) subsample this image into the 4:2:0 format;

(c) upsample the Cr and Cb components to full size (4:4:4 format) and convert the result back to RGB;

(d) Compute the difference between the original and processed RGB images.

20.10 Prove (using simple numerical calculations) that the raw data rate required to transmit an SMPTE 295M digital video signal (1920×1080 luminance samples per frame, 30 frames per second, progressive scanning, 4:2:0 chroma subsampling) is approximately 746 Mbps.