Chapter 1

Introduction

Multimedia means, from the user's perspective, that computer information can be represented through audio and/or video, in addition to text, image, graphics and animation. For example, using audio and video, a variety of dynamic situations in different areas, such as sport or ornithology lexicon, can often be presented better than just using text and image alone.

The integration of these media into the computer provides additional possibilities for the use of computational power currently available (e.g., for interactive presentation of huge amounts of information). Furthermore, these data can be transmitted through computer and telecommunication networks, which implies applications in the areas of information distribution and cooperative work. Multimedia provides the possibility for a spectrum of new applications, many of which are in place today. On the other hand, one also has to keep in mind the problems of global communication, with its social and legal implications. However, these issues will not be discussed in this book.

One of the first and best-known institutes working on different aspects of multimedia is the MIT Media Lab in Boston [HS93]. Research is going on there on a variety of future applications, such as personal newspaper and holography [Bra87]. In the meantime, many research institutes, universities and computer and telecommunication companies work in the multimedia area.

1.1 Branch-overlapping Asᵨects of Multimedia

In addition to the strong interest in multimedia systems from the applications and technology viewpoints, one should not underestimate the ongoing migration process, and the evolution of the different industrial branches, as follows:

- *Telecommunication* began primarily with the telephone. Telephone networks changed gradually into digital networks, which are similar to computer networks. In the early days, intermediate switching systems had mechanical switching dials. Today, computers do the job. Now the telephone is becoming more and more of a computer, or is part of a computer (e.g., using an ISDN card).
- *Consumer electronics* contributed massively to the reduction of the price of video support in computers. Similarly, optical disc technology in computing is dependent on the success of the CD player. Consequently, the same companies very often produce CD drives for computer and stereo-systems, or both television and computer monitors.
- *Recording studios* and *TV producers* are pioneers in dealing with professional audio and video equipment. Today, professional systems are available for the digital cutting of TV movies. Some of these systems are standard computers extended through additional special cards. These information providers convey data through cable, satellite and plain old antennas, which will further allow them to serve as information providers via computer networks in the future.
- Many of the large *publishing houses* already offer their publications in electronic form. Further, there are many close relations between publishing houses and movie companies. These branches offer gradually more and more multimedia information.

This short overview shows that the different branches grow together because of coming multimedia technology and applications. However, to allow multimedia applications, many software and hardware components have to be adapted, extended or replaced.

From the technical perspective, besides handling the huge amount of data, the *timing requirements* among all components of the data computation is the ma·‧r challenge. Traditional data computation tries to finish its task as soon as possible. Real-time systems must work internally within given time bounds, mostly as error-tolerant systems. The fault tolerance in multimedia is generally not the most important aspect.

Another challenge is the *integration requirement* of different types of media in a multimedia application. In such applications, the traditional media (e.g., text, image) as well as the continuous media (e.g., video, audio), must be processed. Moreover, if a timing requirement is set by a multimedia application, it should hold for both classes (traditional and continuous media) to achieve the timing specification of the application. These media are not independent of each other and therefore the integration requires concepts, which are more complex than just the integration of current concepts. In an integrated system, different components have to process both kinds of data, and moreover, different relations can occur in the form of *synchronization* among the media.

The notion of *multimedia* is often defined very differently in the literature in comparison to our (above) description. There is some need for clarification. The technology connections and binding different components, were considered only partially and in isolation from each other. Based on [Ste93b], we wrote this book to provide an integrated, consistent and total view.

1.2 Content

This book has the character of a *reference book*, covering numerous areas and allowing the reader to learn about a topic of interest without having previously studied extensively or having read areas previously covered in this book. Strong connections are provided among the different areas of this book through its *global structure*, which is shown in Figure 1.1. The results presented in this book serve as basis for the development of individual components of a multimedia system and suggest some general parameters one must keep in mind.

1.3 Global Structure

This book aims to achieve a complete and balanced view. Figure 1.1 shows the global view of this book with the main topics covered in it. Figure 1.1 was developed after many iterations of topic structuring. It shows schematically the main fields of multimedia systems. The basic idea of the figure is to express the interactions among the components through spatial proximity.

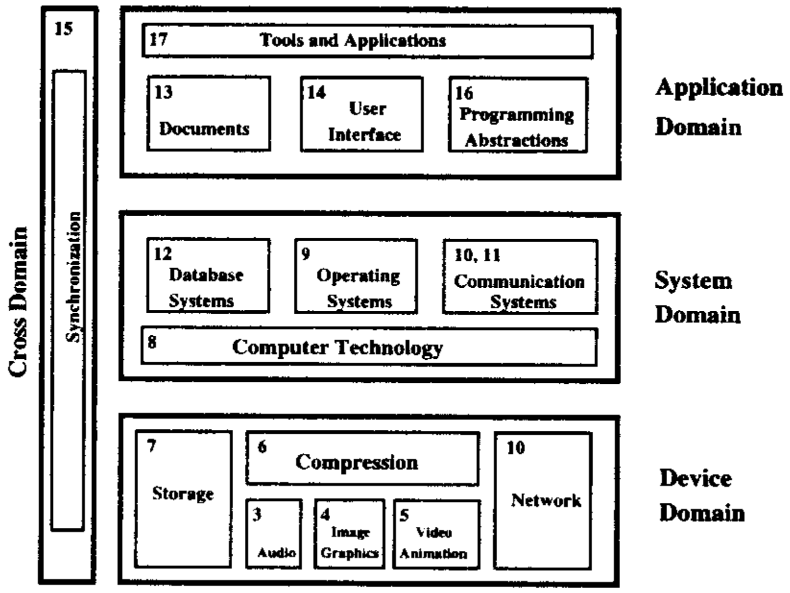

Figure 1.1: *The main topics covered in this book with chapter number information.*

The following areas can be distinguished:

- **Device Domain**

  Basic concepts for the processing of *digital audio* and *video* data are based on digital signal processing. Hence, these concepts are described and some possible practical implementations are presented. Different methods for the processing of *image*, *graphics* and *animation* are described. The audio techniques section includes music (MIDI) and speech processing. The understanding of video techniques is built mainly on TV development, including digital representation and HDTV. The originated data rates of these media demand, because of the current quality requirements and available technology, corresponding *compression* methods. The corresponding hardware and some software are briefly described.

The diminishing cost of *optical storage* space has contributed significantly to the current development of computer technology. Almost all developments are based on CD-DA (Compact Disc-Digital Audio), known from home electronics.

On the other hand, networks, with their higher bandwidth and their capacity for transmitting all media types, have led to *networked multimedia systems* Not such a long time ago, local and distributed multimedia systems consisted of a set of external analog devices controlled by a computer. Today, development tends toward full digital working systems.

- **System Domain**

  The interface between the device domain and the system domain is specified by the *computer technology.* To utilize the device domain, several system services are needed. Basically, three services exist. These services are mostly implemented in software:

  - The *operating system* serves as an interface between computer hardware/system software and all other software components. It provides the user with a programming and computational environment, which should be easy to operate. In its function as an interface, the operating system provides different services that relate to the computer resources, such as: processor, main memory, secondary storage, input and output devices and network.
  - The *database system* allows a structured access to data and a management of large databases.
  - The *communication system* is responsible for data transmission according to the timing and reliability requirements of the networked multimedia application.

- **Application Domain**

  The services of the system domain are offered to the application domain through proper *programming abstractions.* Moreover, such abstractions can be, for example, part of a multimedia operating system, programming language or object-oriented class hierarchy.

Another topic embedded in the application domain is *document handling.* A document consists of a set of structured information, represented in different media, and generated or recorded at the time of presentation.

Many functions of *document handling* and other *applications* are accessible and presented to the user through a *user interface.*

- **Cross Domain**

  It turns out that some aspects, such as synchronization aspects, are difficult to locate in one or two components or domains. The reason is that *synchronization*, being the temporal relationship among various media, relates to many components across all domains.

1.4 Multimedia Literature

All individual topics mentioned in this work have been considered to some extent, and often in more detail, though in isolation, in the literature. Likewise, the groundwork for signal processing, audio, video, graphics, image and animation techniques and various networks have been published in relevant works. In this book, all the parts of an integrated multimedia system will be considered.

Some publications, such as [BD92, EH92, Gia92, Koe94], are composed of individual articles or chapters originated by different authors, with the goal of presenting the topics of user interfaces or applications from different points of view. In contrast to the present work, there was no attempt to provide a coherent integral presentation of the multimedia area.

Other publications serve as individual product descriptions, e.g., [BS92]. They present mostly the properties of individual products without giving a global view. In [Bur93] and [Ste92], besides the application and product descriptions, there is also some groundwork for audio techniques, video techniques, compression and optical storage media. But, in contrast to the present study, all scientific aspects, as well as an overall, integrated view, are neglected. This is the case also in [PF92, V. 93, Wod92].

Hypertext and hypermedia are documented in a large number of publications, most of which offer hypermedia document descriptions. A very good presentation is given in [Nie90a]. We consider hypermedia a part of multimedia.

Besides a number of national and international workshops in the multimedia area, new conferences such as ACM Multimedia Conference (first in August 1993, Anaheim, CA) and IEEE Multimedia Conference (first in May 1994, Boston, MA) evolved. Also, in addition to many product and commercial only oriented magazines, dedicated publications, such as the ACM Springer journal *"Multimedia Systems"* (end of 1993), *"Multimedia Tools and Applications"* (beginning of 1995) and the *"IEEE Multimedia Magazine"* (beginning of 1994) are being published.

Chapter 2

Multimedia: Media and Data Streams

The following chapter introduces terminology and gives a sense of the commonality of the elements of multimedia. The introduction of terminology begins with a clarification of the notion *multimedia*, followed by a description of media and the important properties of multimedia systems. Subsequently, characteristics of *data streams* in such systems and the introduction of the notion *Logical Data Unit* (LDU) follow.

One way of defining multimedia can be found in the meaning of the composed word.

- *Multi-* [lat.: much] many; much; multiple.
- *Medium* [lat.: middle] An intervening substance through which something is transmitted or carried on; A means of mass communication such as newspaper, magazine, or television (from American Heritage Electronic Dictionary, 1991).

This description is derived from the common forms of human interaction. It is not very exact and has to be adapted to computer processing. Therefore, we discuss in the next section the notion *medium* in more detail with respect to computer processing.

2.1 Medium

In general, one describes medium as a means for distribution and presentation of information. Examples of a medium are text, graphics, speech and music. In the same way, one can also add water and atmosphere to it (according to the above medium description from the American Heritage Dictionary).

Media can be classified with respect to different criteria [ISO93a]. We classify media according to perception, representation, presentation, storage, transmission, and information exchange.

2.1.1 The Perception Medium

Perception media help humans to sense their environment. The central question is: *How do humans perceive information in a computer environment?* The answer is that the perception of information occurs mostly through *seeing* or *hearing* the information, although tactile perception increases its presence in a computer environment.

There is a primary difference between *seeing* and *hearing* information when using a computer. For the perception of information through seeing, the visual media such as *text, image* and *video* are used. For the perception of information through hearing, auditory media such as *music, noise* and *speech* are relevant.

The difference among media can be further refined. For example, video can be further decomposed into different video scenes, which again are composed of individual images.

2.1.2 The Representation Medium

Representation media are characterized by internal computer representations of information. The central question is: *How is the computer information coded?* The answer is that various formats are used to represent media information in a computer. For example:

- A text character is coded in ASCII or EBCDIC code.
- Graphics are coded according to CEPT or CAPTAIN videotext standard. The graphics standard GKS can also serve as a basis for coding,
- An audio stream can be represented using a simple PCM (Pulse Coding Method) with a linear quantization of 16 bits per sample.
- An image can be coded as a facsimile (the group 3 according to the ISO Standard Specification) or in JPEG format.
- A combined audio/video sequence can be coded in different TV standard formats (e.g., PAL, SECAM, NTSC), and stored in the computer using an MPEG format.

2.1.3 The Presentation Medium

Presentation media refer to the tools and devices for the input and output of information. The central question is: *Through which medium is information delivered by the computer, or introduced into the computer?* The media, e.g., paper, screen and speaker are used to deliver the information by the computer (output media); keyboard, mouse, camera and microphone are the input media.

2.1.4 The Storage Medium

Storage media refer to a data carrier which enables storage of information. However, the storage of data is not limited only to the available components of a computer. Therefore, paper is also a storage medium. The central question is: *Where will the information be stored?* Microfilm, floppy disk, hard disk. and CD-ROM are examples of storage media.

2.1.5 The Transmission Medium

The transmission medium characterizes different information carriers, that enable continuous data transmission. Therefore, storage media are excluded from this kind of medium. The central question is: *Over what will the information be transmitted?* The answer is that information is transmitted over networks, which use wire and cable transmission, such as coaxial cable and fiber optics, as well as free air space transmission, which is used for for wireless traffic.

2.1.6 The Information Exchange Medium

The information exchange medium includes all information carriers for transmission, i.e., all storage and transmission media. The central question is: *Which information carrier will be used for information exchange between different places?* The answer is that information can flow through *intermediate* storage media, where the storage medium is transported outside of computer networks to the destination, through *direct* transmission using computer networks, or through *combined* usage of storage and transmission media (e.g., electronic mailing system).

2.1.7 Representation Values and Representation Spaces

The above classification of media can be used as a basis for characterizing the notion *medium* in the context of information processing. Here, the description of perception medium comes closest to our notion of a medium: the media appeal to the human senses. Each medium defines *representation values* and *representation spaces* [HD90, HS91], which involve the five senses.

Examples of visual representation spaces are paper or screen. During a computer-controlled slide show with simultaneous projection of the computer screen content, the whole movie screen counts as a representation space. Stereo and quadraphony determine the acoustic representation spaces. Representation spaces can also be considered part of the above described presentation media for information output.

Representation values determine the information representation of different media: while the *text* medium visually represents a sentence through a sequence of characters, this sentence will be represented by the *speech* medium in the form of a pressure wave. Some representation values are self-contained by their media. In other words, they can be properly interpreted by the recipient. Examples here are temperature, taste, and smell. Other media require a predefined symbol set, which the users must agree upon. Text, speech and gestures are examples of such media.

Representation values can be considered either as a continuum or a sequence of discrete values. Pressure wave fluctuations do not appear as discrete values; instead they determine the acoustic signals. Electromagnetic waves for human eye perception are not discrete values either; rather they are a continuum. Characters of a text and audio sample values in electronic form are sequences of discrete values.

2.1.8 Representation Dimensions

Each representation space consists of one or more *representation dimensions.* A computer screen has two spatial dimensions; holography and stereophony require an additional spatial dimension. Time can occur inside each representation space as an additional dimension, as it has central meaning to multimedia systems. Media are divided into two types with respect to time in their representation space:

1. Some media, such as text and graphics, are time-independent. Information in these media consist exclusively of a sequence of individual elements or of a continuum without a time component. Such media are known as *time-independent* (or *discrete*). Note, the notion 'discrete' is sometimes confusing, because a medium can also be discrete in value but continuous in time.). The text of a book is, for example, a discrete medium. Processing of discrete media should happen as fast as possible, but this processing is not time critical because the validity (and therefore correctness) of the data does not depend on any time condition.

2. The values of other media, such as sound and full-motion video, change over time. Information is expressed not only in its individual value, but also by the time of its occurrence. The semantics depend on the level of the relative change of the discrete values or of the continuum. Such media are *time-dependent.* Also, representation values caused by tactile or temperature sensors with threshold detectors are time-dependent, and therefore also belong to the time-dependent media.

Processing of these media is time-critical because the validity (and therefore correctness) of the data depends on a time condition. For example, a transmitted audio sample delivered too late is invalid if the following samples to the sample in question have already been played back.

Individual representation values occur in audio and video as a continuous sequence. We understand *video* as a sequence of plain images occurring periodically, as well as audio being a sequence of samples with periodic behavior. We call these media *continuous media*. Using this division, time-dependent representation values, which occur aperiodically, are not considered continuous media. Control commands for real-time systems are an example. In multimedia systems, we must also consider non-continuous sequences of representation values. Such sequences occur, for example, by transmission of information (e.g., mouse pointer position) in a cooperative application within a shared window.

Examples of continuous media are: *video* coming from natural source (e.g., video taken by a camera during a live video transmission) or from an artificial source (e.g., video disc); *audio*, which is mostly stored as a sequence of digitalized sound-wave samples; and *signals of different sensors*, such as those that sense air pressure, temperature, humidity, pressure or radioactivity.

These notions of time-dependent, discrete and continuous media do not have any connection to internal representation. They relate to the impression of the viewer or listener. For example, a movie as a representative of continuous media often consists of a sequence of discrete values, which change in representation space according to a time function. The inertia of the human eye only leads to the impression of continuity if a sequence of at least 16 individual images per second is provided.

2.2 Main Properties of a Multimedia System

2.2.1 Multimedia System Definition

If we derive a multimedia system from the meaning of the words in the American Heritage Dictionary, then a multimedia system is any system which supports more than a single kind of media. This characterization is insufficient because it only deals with a *quantitative* evaluation of the system. For example, each system processing text and graphics would be classified as a multimedia system according to this narrow definition. Such systems already existed before the multimedia notion was used in a computer environment. Hence, the notion *multimedia* implies a new quality in a computer environment.

We understand multimedia more in a *qualitative* rather than a quantitative way. Therefore, the kind rather than the number of supported media should determine if a system is a multimedia system. It should be pointed out that this definition is controversial. Even in the standardization bodies, e.g., ISO, a weaker interpretation is often used.

A multimedia system distinguishes itself from other systems through several properties. We elaborate on the most important properties such as combination of the media, media-independence, computer control and integration.

2.2.2 Combination of Media

Not every arbitrary combination of media justifies the usage of the term *multimedia.* A simple text processing program with incorporated images is often called a multimedia application because two media are processed through one program. But one should talk about multimedia only when both continuous and discrete media are utilized. A text processing program with incorporated images is therefore not a multimedia application.

2.2.3 Independence

An important aspect of different media is their level of *independence* from each other. In general, there is a request for independence of different media, but multimedia may require several levels of independence. On the one hand, a computer-controlled video recorder stores audio and video information, but there is an inherently tight connection between the two types of media. Both media are coupled together through the common storage medium of the tape. On the other hand, for the purpose of presentations, the combination of DAT recorder (Digital Audio Tape) signals and computer-available text satisfies the request for media-independence.

2.2.4 Computer-supported Integration

The media-independence prerequisite provides the possibility of combining media in arbitrary forms. Computers are the ideal tool for this purpose. The system should be capable of computer-controlled media processing. Moreover, the system should be programmable by a system programmer or even a user. Simple input or output of different media through one system (e.g., a video recorder) does not satisfy the requirement for a computer-controlled solution. Computer-controlled data of independent media can be integrated to accomplish certain functions. For such a purpose, timing, spatial and semantic synchronization relations will be included. A text processing program that supports text, table calculations and video clips does not satisfy the demand for integration if program supporting the connection between the data cannot be established. A high integration level is accomplished if changing the content of a table row causes corresponding video scene and text changes.

Such flexible processing of media is not obvious – even in many of the best available multimedia products. Therefore, this aspect must be emphasized in terms of an *integrated multimedia system*. Simply put, in such systems, everything can be presented with video and sound that is presented with text and graphics today [AGH90]. For example, in conventional systems, a text message can be sent to other users; but, a multimedia system with a high level of integration allows this function also for audio messages or even for a combination of audio and text.

2.2.5 Communication Systems

Communication-capable multimedia systems must be approached. A reason for this is that most of today's computers are interconnected; considering multimedia functions from only the local processing viewpoint would be a restriction, if not a step back. Another reason is that distributed environments enable particularly interesting multimedia applications. Here multimedia information cannot only be created, processed, presented and stored, but also distributed above the single computer's boundary.

2.3 Multimedia

Considering the first explanation of multimedia at the beginning of this chapter, it is apparent that the notion is insufficient. We derive the following definition for multimedia from the American Heritage Dictionary definitions and the above considerations with respect to the medium (Section 2.1) and to the main properties of a multimedia system (Section 2.2):

A multimedia system is characterized by computer-controlled, integrated production, manipulation, presentation, storage and communication of independent information, which is encoded at least through a continuous (time-dependent) and a discrete (time-independent) medium.

Multimedia is very often used as an attribute of many systems, components, products, ideas, etc., without satisfying the above presented characteristics. From this viewpoint our definition is deliberately restrictive.

Thus, two notions of multimedia can be distinguished:

- **"Multimedia", strictly speaking:**

  This notion was explained in Section 2.2 and will be used further. In this context, continuous media will always be included in a multimedia system. At the same time important timely marginal conditions (through the continuous media) for the processing of discrete media will be introduced. They have barely been considered in computer use until now.

- **"Multimedia", in the broader sense:**

  Often the notion *multimedia* is used to describe the processing of individual images and text, although no continuous medium is present. Many of the processing tasks in this environment will also be necessary in the multimedia system according to the restrictive definition. In any case, if more media are processed together, one can talk about multimedia according to this second notion.

2.4 Traditional Data Streams Characteristics

In Sections 2.1, 2.2, and 2.3 we clarified the multimedia notion from the local computer-based point of view. But the presented work also includes the consideration of multimedia communication systems. Therefore, we need to specify the multimedia notion from the communication point of view.

In distributed multimedia communication systems, data of discrete and continuous media are transmitted and information exchange takes place. Moreover, in each digital system, transmitted information is divided into *individual units* (in general, these are packets) and subsequently sent away from one system component (the source) to another (the destination). The source and destination can be located either on the same computer or on different machines. A sequence of individual packets transmitted in a time-dependent fashion is called a *data stream* (The term "data stream" will be used here as a synonym for "data flow".). Packets can carry information of either continuous or discrete media. An example of a continuous media data stream is the transmission of speech in a telephone system. The retrieval of a document from a database can be seen as setting up a discrete media data stream.

Transmission of information carrying different media leads to data streams with very different features. The attributes of *asynchronous*, *synchronous*, and *isochronous* data transmission come from the fields of computer communication and switching. They are also used, for example, in FDDI (Fiber Distributed Data Interface) networks for the description of different data transmission modes with respect to end-to-end delay of individual packets (see Figures 10.5 and 10.6).

2.4.1 Asynchronous Transmission Mode

The *asynchronous transmission mode* provides for communication with no timely restrictions. Packets reach the receiver as fast as possible.

Protocols of the worldwide Internet for electronic mail transmission are an example. In local area networks, Ethernet is a further example. All information of discrete media can be transmitted as an asynchronous data stream. Data of discrete media can also include time restrictions through the timely connection to continuous media synchronization. In this case an asynchronous transmission might not be appropriate. If an asynchronous mode is chosen for transmission of continuous media, additional techniques must be applied to provide the time restrictions.

2.4.2 Synchronous Transmission Mode

The *synchronous transmission mode* defines a maximum end-to-end delay for each packet of a data stream. This upper bound will never be violated. Moreover, a packet can reach the receiver at any arbitrary earlier time. Thus, an important claim of multimedia applications is satisfied: a maximal end-to-end delay can be guaranteed.

Additionally, an audio connection can be established over a local area network which supports synchronous transmission mode. The uncompressed transfer of video data in a retrieval mode is characterized by a high data rate and relatively high maximal end-to-end delay. Here the typical data rate is 140 Mbit/s and a maximal delay can be 1 second. In extreme cases packets arrive at the receiver 1 second too early and have to be stored intermediately. In our example, a receiver would need a temporary storage of about 17.5 Mbytes.

2.4.3 Isochronous Transmission Mode

The *isochronous transmission mode* defines, besides a maximum end-to-end delay for each packet of a data stream, a minimum end-to-end delay. This means that the delay jitter (in short, "jitter") of individual packets is bounded.

In this case, the necessary storage of video data at the receiver, mentioned in the above example, would be strongly reduced. These demands on intermediate storage must be considered in all components along the data route between source(s) and sink(s).

2.5 Data Stream Characteristics for Continuous Media

The following section describes data stream characteristics that relate to any audio and video data transfer in a multimedia systems (*multimedia data streams*). Moreover, we consider the effects of compression on data stream characteristics during data transfer. These data stream characteristics apply to distributed as well as local environments.

2.5.1 The Time Interval Between a Complete Transmission of Consecutive Packets

This first property relates to the time interval between a complete transmission of consecutive packets. Based on the availability of packets, we distinguish among the following possibilities:

- If the time interval between two consecutive packets is constant, a data stream is called *strongly periodic*. Therefore, in an ideal case, jitter has the value zero. Figure 2.1 shows such a data stream. An example is PCM-coded speech in

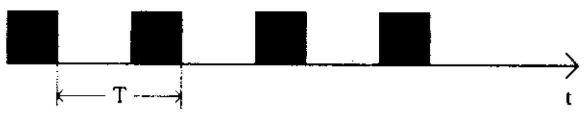

Figure 2.1: *Strongly periodic stream, (T-time limit between two consecutive packets), i.e., time intervals are of the same length between two consecutive packets.*

  traditional telephone switching systems.

- The duration of the time interval between two consecutive packets can be described through a periodical function with finite period, but the time interval between consecutive packets is not constant (otherwise it would be a strongly periodic data stream). The data stream is called *weakly periodic*. This case is shown in Figure 2.2.

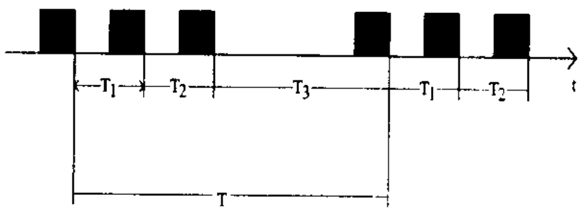

Figure 2.2: *Weakly periodic stream, i.e., time intervals between consecutive packets are of periodic nature.*

- All other possibilities of transmission with respect to time interval are known as *aperiodic data streams*. Figure 2.3 shows such a data stream.

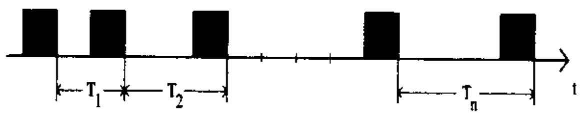

Figure 2.3: *Aperiodic stream, i.e., the sequence of time intervals is neither strongly nor weakly periodic.*

An example of an aperiodic data stream can be found in a cooperative application with shared windows. Very often, the status and actual coordinates of the user's mouse must be distributed among all participants of the multimedia conference. If this information is transmitted periodically, extremely high redundancy is present. Thus, given that an optimal system should transmit information only when necessary, after an initialization phase, data are exchanged only when a change in position or status occurs.

2.5.2 Variation of Consecutive Packet Amount

A second characteristic of data streams is the variation of the amount of consecutive packets.

- If the amount of data stays constant during the lifetime of a data stream, one calls the data stream *strongly regular*. Such a data stream is shown in Figure 2.4. This feature is typical for uncompressed digital data transmission

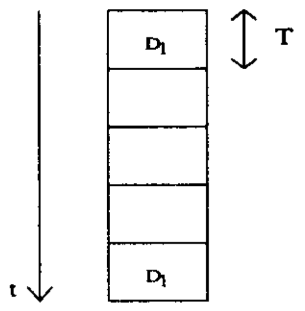

Figure 2.4: *Strongly regular stream, i.e., constant data size of all packets.*

Examples are the video stream taken from a camera in uncompressed form and the audio stream from an audio CD.

- If the amount of data varies periodically (with time), this is a *weakly regular* data stream. An example of a weakly regular data stream is a compressed video stream which uses a compression method as follows: individual images are coded and compressed as an individual, whole unit, which represents a relatively large packet inside the data stream (bounded packet length of network transmission is left out in this consideration.). Packets will be periodically transmitted, e.g., every two seconds. Inbetween the two second periods, additional packets will be sent which include the information about the difference of the two consecutive compressed images.

  An example of a compression method which works similarly to the above description is the MPEG compression method (see Section 6.7). MPEG differentiates among I, P and B images in a compressed video stream. I-images represent compressed individual images, while P- and B-images take into account image differences. With this approach the data rate is reduced essentially. There is no constant bit rate for individual I, P, B compressed packets, but the I:B:P relation of the created data amount for every image is known (often used value of the I:B:P relation is 10:1:2 for individual images.). Such a data stream can be characterized on average over a long time period as *weakly regular* (Figure 2.5).

- Data streams are *irregular* if the amount of data is neither constant nor changes according to a periodic function (see Figure 2.6). Transmission and processing

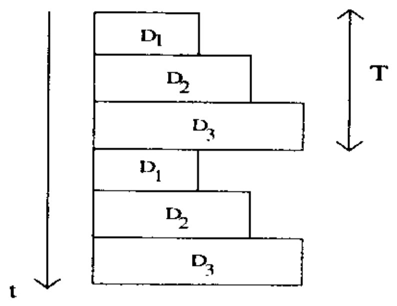

Figure 2.5: *Weakly regular stream, i.e., data size of the packets changes periodically.*

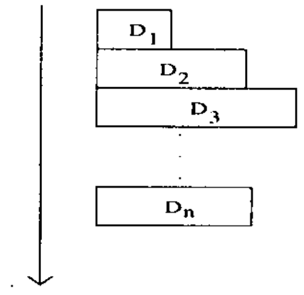

Figure 2.6: *Irregular data stream, i.e., data size of the packets is neither constant nor changing periodically.*

is more complicated in this case. In the case when a compression method is applied to the data stream, the data stream has a variable bit rate, and the size of an individual packet (derived from an individual image) is determined from the content of the previous changed image. The size of the created information unit is therefore dependent on the video sequence and the data stream is *irregular.*

2.5.3 Contiguous Packets

A third property characterizes *continuity*, or the connection between consecutive packets. Are consecutive packets transmitted directly one after another, or is there a gap between the packets? This can be seen as utilization of a certain system resource, such as a network.

- Figure 2.7 shows a *connected* information transfer. All packets are trans-

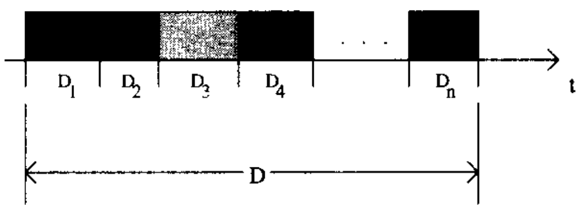

Figure 2.7: *Continuous stream, i.e., the packets are transmitted without intermediate gaps.*

  mitted successively without a gap. Necessary additional information (e.g., error control codes) of the data is considered. In this case, the considered system resource is 100% utilized. A connected data stream allows maximal data throughput and reaches optimal utilization of the system resource. A B-channel of ISDN with transmission of 64 kbit/s audio data is an example.

- The transmission of a connected data stream through a channel with a higher capacity leads to gaps between individual packets. A data stream with gaps between information units is called an *unconnected data stream.* An example is shown in Figure 2.8. However, it is not important if gaps exist among all

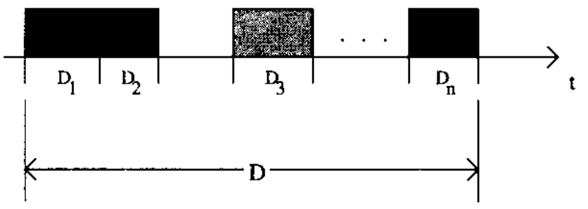

Figure 2.8: *Discrete stream, i.e., gaps exist among the packets.*

  packets or if the duration of the gaps varies. For example, the transmission of a data stream, coded with the JPEG method, with 1.2 Mbit/s throughput on average, will lead to gaps among individual packets on an FDDI network.

In the following example, the properties described above should be made clear: an NTSC video signal is captured from a video camera and digitized in a computer, yet no compression is done. The created data stream is strongly periodic, strongly regular and connected, as shown in Figure 2.4. There are no gaps among the packets. During the digitizing process, the DVI_RTV method for compression, using the ActionMedia II^{TM}card, is performed. The resulting data stream (considered over a longer period of time) is now weakly periodic, weakly regular, and, through transmission over a 16 Mbit/second Token Ring, unconnected.

2.6 Information Units

Continuous media consist of a time-dependent sequence of individual information units. Such an information unit is called a *Logical Data Unit* (LDU), which is close to a *Protocol Data Unit* (PDU). The meaning of the information and data amount of an LDU can be different:

1. Consider for example the symphony "*The bear*" by Joseph Haydn. It consists of four sentences: *vivace assai, allegretto, minuet* and *finale vivace*. Each sentence is an independent, self-contained part of this composition, consisting of a sequence of notes for different instruments. The notes are represented in a digital system as a sequence of samples (no compression is considered in our example.). With CD-DA quality, there are 44,100 samples per second, which are coded with 16 bits per sample. On a CD the samples are grouped into units of 1/75 second duration. One could take as the LDU the whole symphony, individual sentences, individual notes, grouped samples of 1/75 second duration or just individual samples. The particular application determines what is considered to be the LDU. For example, applications using output functions of the whole symphony will take the whole symphony as the LDU. Other applications use functions which consider the smallest meaningful units (in our case, notes). A digital system considers samples as the LDUs.

2. An example of an uncompressed video sequence consisting of individual video clips, which present a specific scene, is shown in Figure 2.9. Such a scene is comprised of a sequence of images. An image can be divided, for example, into 16x16 groups of pixels. Each pixel consists again of luminance and chrominance values. The image is therefore not the only possible LDU of a video sequence. A scene or a pixel can also be an LDU. In a video sequence, coded

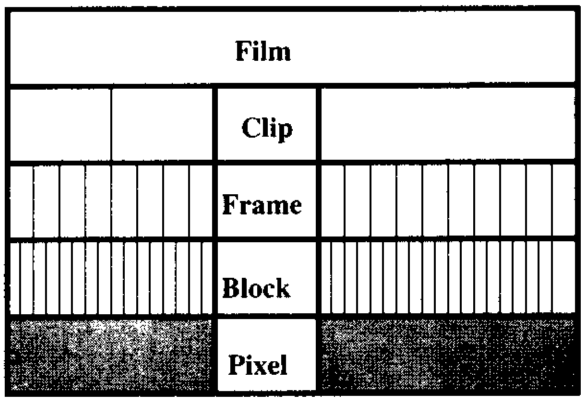

Figure 2.9: *Granularity of a motion picture sequence.*

with MPEG or DVI, existent redundancies can be used through applying an interframe compression method. The smallest self-contained meaningful units here are image sequences.

The notion of *granularity* characterizes the hierarchical division of audio or video data streams into their components. In our examples, the most general names and best-known information units are the symphony and the movie. Yet there exists also another classification of LDU with respect to duration. *Closed LDUs* have a pre-defined duration. An example of such an LDU stream is a data stream of audio samples in the computer. If the duration is not known in advance, we encounter an *open LDU*. An example of such an LDU stream is a data stream sent from a camera or microphone to the computer.

Chapter 3

Sound / Audio

Sound is a physical phenomenon produced by the vibration of matter, such as a violin string, or a block of wood. As the matter vibrates, pressure variations are created in the air surrounding it. This alteration of high and low pressure is propagated through the air in a wave-like motion. When a wave reaches the human ear, a sound is heard.

Sound methodology and audio techniques engage in processing these sound waves (acoustic signals). Important topics in this area are coding, storage on recorders or digital audio tapes, music and speech processing.

In this chapter a discussion of sound, music coding and speech processing is presented. Basic concepts and formats of sound, as well as representation of sound in the computer [Boo87, Tec89] are presented. Because multimedia applications use audio in the form of music and/or speech, music and its MIDI standard, as well as speech synthesis, speech recognition and speech transmission [Loy85, Fla72, FS92, O'S90, Fal85, Bri86, Ace93, Sch92], are described.

The topic of audio data storage on optical discs is presented in Chapter 7, for the reason that the principles and technology used are not restricted to audio. The compression of audio/video signals is also described separately in Chapter 6, because similar methods are used for compressing the data of different media. Further, the commonalities among the media are emphasized by being treated together.

3.1 Basic Sound Concepts

Sound is produced by the vibration of matter. During the vibration, pressure variations are created in the air surrounding it. The pattern of the oscillation is called a *waveform* (Figure 3.1 [Tec89]).

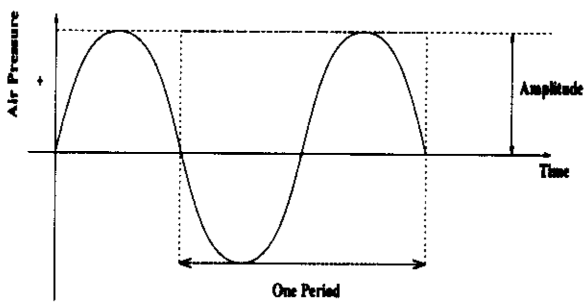

Figure 3.1: *Oscillation of an air pressure wave.*

The waveform repeats the same shape at regular intervals and this portion is called a *period.* Since sound waves occur naturally, they are never perfectly smooth or uniformly periodic. However, sounds that display a recognizable periodicity tend to be more musical than those that are nonperiodic. Examples of periodic sound sources are musical instruments, vowel sounds, the whistling wind and bird songs. Nonperiodic sound sources include unpitched percussion instruments, coughs and sneezes and rushing water.

Frequency

The *frequency* of a sound is the reciprocal value of the period; it represents the number of periods in a second and is measured in *hertz* (Hz) or *cycles per second* (cps). A convenient abbreviation, kHz (kilohertz), is used to indicate thousands of oscillations per second: 1 kHz equals 1000 Hz [Boo87]. The frequency range is divided into:

Infra-sound	from 0 to 20 Hz
Human hearing frequency range	from 20 Hz to 20 kHz
Ultrasound	from 20 kHz to 1 GHz
Hypersound	from 1 GHz to 10 THz

Multimedia systems typically make use of sound only within the frequency range of human hearing. We will call sound within the human hearing range *audio* and the waves in this frequency range *acoustic signals* [Boo87]. For example, speech is an acoustic signal produced by humans; music signals have a frequency range between 20 Hz and 20 kHz. Besides speech and music, we denote any other audio signal as *noise*.

Amplitude

A sound also has an *amplitude*, a property subjectively heard as loudness. The amplitude of a sound is the measure of the displacement of the air pressure wave from its mean, or quiescent state.

3.1.1 Computer Representation of Sound

The smooth, continuous curve of a sound waveform is not directly represented in a computer. A computer measures the amplitude of the waveform at regular time intervals to produce a series of numbers. Each of these measurements is a *sample*. Figure 3.2 illustrates one period of a digitally sampled waveform.

The mechanism that converts an audio signal into digital samples is the *Analog-to-Digital Converter (ADC)*. The reverse conversion is performed by a *Digital-to-Analog Converter (DAC)*. The AM79C30A Digital Subscriber Controller chip is an example of an ADC and is available on SPARCstations^{TM}. Desktop SPARC^{TM}systems include a built-in speaker for audio output. DAC is also available as a standard UNIX^{TM}device. For example, SPARCserver 6xx systems do not have an internal speaker, but support an external microphone and speaker.

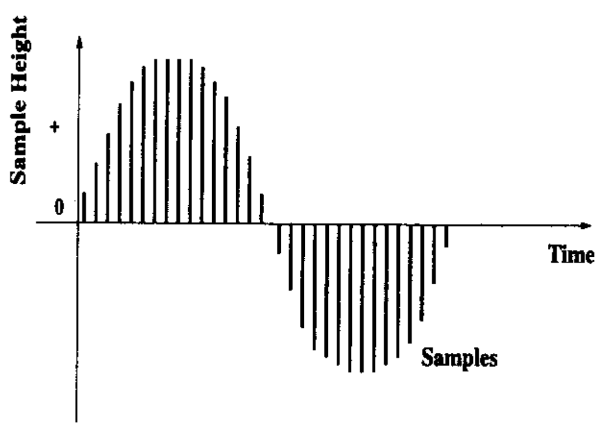

Figure 3.2: *Sampled waveform.*

Sampling Rate

The rate at which a continuous waveform (Figure 3.1) is sampled is called the *sampling rate*. Like frequencies, sampling rates are measured in Hz. The CD standard sampling rate of 44100 Hz means that the waveform is sampled 44100 times per second. This seems to be above the frequency range the human ear can hear. However, the bandwidth (which in this case is 20000 Hz - 20 Hz = 19980 Hz) that digitally sampled audio signal can represent, is at most equal to half of the CD standard sampling rate (44100 Hz). This is an application of the Nyquist sampling theorem. ("For lossless digitization, the sampling rate should be at least twice the maximum frequency responses.") Hence, a sampling rate of 44100 Hz can only represent frequencies up to 22050 Hz, a boundary much closer to that of human hearing.

Quantization

Just as a waveform is sampled at discrete times, the value of the sample is also discrete. The resolution or *quantization* of a sample value depends on the number of bits used in measuring the height of the waveform. An 8-bit quantization yields 256 possible values; 16-bit CD-quality quantization results in over 65536 values.

Figure 3.3 presents a 3-bit quantization. The sampled waveform with a 3-bit quan-

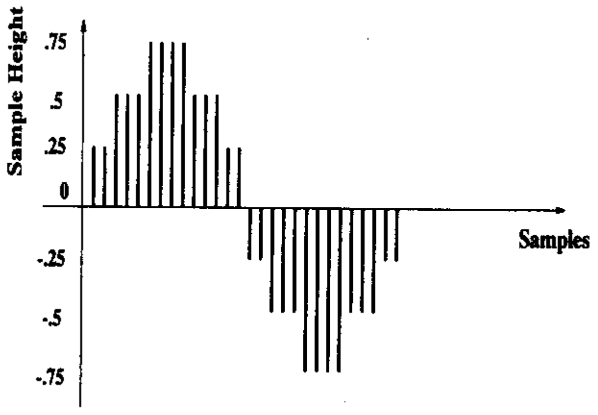

Figure 3.3: *Three-bit quantization.*

tization results in only eight possible values: .75, .5, .25, 0, -.25, -.5, -.75 and -1. The shape of the waveform becomes less discernible with a lowered quantization, i.e., the lower the quantization, the lower the quality of the sound (the result might be a buzzing sound).

Sound Hardware

Before sound can be processed, a computer needs input/output devices. Microphone jacks and built-in speakers are devices connected to an ADC and DAC, respectively for the input and output of audio.

3.1.2 Audio Formats

The AM79C30A Digital Subscriber Controller provides *voice-quality* audio. This converter uses an 8-bit μ-law encoded quantization and a sampling rate of 8000 Hz. This representation is considered fast and accurate enough for *telephone-quality* speech input.

CD-quality audio is generated if the stereo DAC operates at 44100 samples per second with a 16-bit *linear PCM (Pulse Code Modulation)* encoded quantization [JB89].

The above examples of telephone-quality and CD-quality audio indicate that important format parameters for specification of audio are: *sampling rate* (e.g., 8012.8 samples/second) and *sample quantization* (e.g., 8-bit quantization).

3.2 Music

The relationship between music and computers has become more and more important, specially considering the development of MIDI (Music Instrument Digital Interface) and its important contributions in the music industry today. The MIDI interface between electronic musical instruments and computers is a small piece of equipment that plugs directly into the computer's serial port and allows the transmission of music signals. MIDI is considered to be the most compact interface that allows full-scale output.

3.2.1 MIDI Basic Concepts

MIDI is a standard that manufacturers of electronic musical instruments have agreed upon. It is a set of specifications they use in building their instruments so that the instruments of different manufacturers can, without difficulty, communicate musical information between one another [Loy85].

A MIDI interface has two different components:

- *Hardware* connects the equipment. It specifies the physical connection between musical instruments, stipulates that a *MIDI port* is built into an instrument, specifies a *MIDI cable* (which connects two instruments) and deals with electronic signals that are sent over the cable.

- A *data format* encodes the information traveling through the hardware. A MIDI data format does not include an encoding of individual samples as the audio format does (Section 3.1.2). Instead of individual samples, an instrument-connected data format is used. The encoding includes, besides the instrument specification, the notion of the beginning and end of a note, basic frequency and sound volume. MIDI data allow an encoding of about 10 octaves, which corresponds to 128 notes.

The MIDI data format is digital; the data are grouped into *MIDI messages*. Each MIDI message communicates one *musical event* between machines. These musical events are usually actions that a musician performs while playing a musical instrument. The action might be pressing keys, moving slider controls, setting switches and adjusting foot pedals.

When a musician presses a piano key, the MIDI interface creates a MIDI message where the beginning of the note with its stroke intensity is encoded. This message is transmitted to another machine. In the moment the key is released, a corresponding signal (MIDI message) is transmitted again. For ten minutes of music, this process creates about 200 Kbytes of MIDI data, which is essentially less than the equivalent volume of a CD-audio coded stream in the same time.

If a musical instrument satisfies both components of the MIDI standard, the instrument is a *MIDI device* (e.g., a synthesizer), capable of communicating with other MIDI devices through *channels*. The MIDI standard specifies 16 channels. A MIDI device (musical instrument) is mapped to a channel. Music data, transmitted through a channel, are reproduced at the receiver side with the synthesizer instrument. The MIDI standard identifies 128 instruments, including noise effects (e.g., telephone, air craft), with unique numbers. For example, 0 is for the Acoustic Grand Piano, 12 for the marimba, 40 for the violin, 73 for the flute, etc.

Some instruments allow only one note to be played at a time, such as the flute. Other instruments allow more than one note to be played simultaneously, such as the organ. The maximum number of simultaneously played notes per channel is a main property of each synthesizer. The range can be from 3 to 16 notes per channel.

To tune a MIDI device to one or more channels, the device must be set to one of the MIDI *reception modes*. There are four modes:

- Mode 1: Omni On/Poly;
- Mode 2: Omni On/Mono;

- Mode 3: Omni Off/Poly;
- Mode 4: Omni Off/Mono

The first half of the mode name specifies how the MIDI device monitors the incoming MIDI channels. If Omni is turned on, the MIDI device monitors all the MIDI channels and responds to all channel messages, no matter which channel they are transmitted on. If Omni is turned off, the MIDI device responds only to channel messages sent on the channel(s) the device is set to receive.

The second half of the mode name tells the MIDI device how to play notes coming in over the MIDI cable. If the option Poly is set, the device can play several notes at a time. If the mode is set to Mono, the device plays notes like a monophonic synthesizer – one note at a time.

3.2.2 MIDI Devices

Through the MIDI interface, a computer can control output of individual instruments. On the other hand, the computer can receive, store or process coded musical data through the same interface. The data are generated with a *keyboard* and reproduced through a sound generator. A *sequencer* can store data. Further, it may also modify the musical data. In a multimedia system, the sequencer is a computer application.

The heart of any MIDI system is the MIDI *synthesizer* device. A typical synthesizer looks like a simple piano keyboard with a panel full of buttons, but it is far more (more detailed information on synthesizers can be found in [Boo87].). Most synthesizers have the following common components:

- *Sound Generators*

  Sound generators do the actual work of synthesizing sound; the purpose of the rest of the synthesizer is to control the sound generators. The principal purpose of the generator is to produce an audio signal that becomes sound when fed into a loudspeaker. By varying the voltage oscillation of the audio signal, a sound generator changes the quality of the sound – its pitch, loudness and tone color – to create a wide variety of sounds and notes.

Internally, sound generation can be done in different ways. One way is to store the acoustic signals as MIDI data in advance. Afterwards, the stored MIDI data are transformed with a digital-analog adapter into acoustic signals. Individual notes are composed in a timely fashion. Another method is to create acoustic signals synthetically.

- *Microprocessor*

  The microprocessor communicates with the keyboard to know what notes the musician is playing, and with the control panel to know what commands the musician wants to send to the microprocessor. The microprocessor then specifies note and sound commands to the sound generators; in other words, the microprocessor sends and receives MIDI messages.

- *Keyboard*

  The keyboard affords the musician's direct control of the synthesizer. Pressing keys on the keyboard signals the microprocessor what notes to play and how long to play them. Some synthesizer keyboards can also signal to the microprocessor how loud to play the notes and whether to add *vibrato* or other effects to the notes. The sound intensity of a tone depends on the speed and acceleration of the key pressure. The keyboard should have at least five octaves with 61 keys.

- *Control Panel*

  The control panel controls those functions that are not directly concerned with notes and durations (controlled by the keyboard). Panel controls include: a slider that sets the overall volume of the synthesizer, a button that turns the synthesizer on and off, and a menu that calls up different patches for the sound generators to play.

- *Auxiliary Controllers*

  Auxiliary controllers are available to give more control over the notes played on the keyboard. Two very common variables on a synthesizer are *pitch bend* and *modulation*. Pitch bend controllers can bend pitch up and down, adding *portamento* (a smooth, uninterrupted glide in passing from one tone to another) to notes; modulation controllers can increase or decrease effects such as *vibrato*.

- *Memory*

  Synthesizer memory is used to store patches for the sound generators and settings on the control panel. Many synthesizers also have a slot for *external memory cartridges*. By using several memory cartridges, the musician can plug in a different cartridge each time s/he wants a set of new sounds for the synthesizer.

There are many other MIDI devices that augment the standard synthesizer in a MIDI system. Examples are drum machines which specialize in percussion sounds and rhythms, the master keyboard which increases the quality of the synthesizer keyboard, guitar controllers, guitar synthesizers, drum pad controllers and so on.

An important MIDI device is a *sequencer*, which can be a drum machine, computer or dedicated sequencer. A sequencer was used originally as a storage server for generated MIDI data. Today, a sequencer, being a computer, becomes additionally a music editor. Data can be modified in a proper way because of their digital data representation. There are several possibilities to represent musical data. The most common representation and manipulation of data are musical notes. The musical piece appears on the screen in the form of a sheet of music. The sequencer transforms the notes into MIDI messages (Sections 3.2.1, 3.2.3). Another representation is a direct input of MIDI messages. Here, the user specifies required musical events per channel with their time dependencies. This input depends on the keyboard type.

3.2.3 MIDI Messages

MIDI messages transmit information between MIDI devices and determine what kinds of musical events can be passed from device to device. The format of MIDI messages consists of the *status byte* (the first byte of any MIDI message), which describes the kind of message, and *data bytes* (the following bytes). MIDI messages are divided into two different types:

- *Channel Messages*

  Channel messages go only to specified devices. There are two types of channel messages:

  - *Channel voice messages* send actual performance data between MIDI devices, describing keyboard action, controller action and control panel changes. They describe music by defining pitch, amplitude, timbre, duration and other sound qualities. Each message has at least one and usually two data bytes that accompany the status byte to describe these sound qualities. Examples of channel voice messages are *Note On*, *Note Off*, *Channel Pressure*, *Control Change*, etc.
  - *Channel mode messages* determine the way that a receiving MIDI device responds to channel voice messages. They set the MIDI channel receiving modes for different MIDI devices, stop spurious notes from playing and affect local control of a device. Examples of such messages are *Local Control*, *All Notes Off*, *Omni Mode Off*, etc.

- *System Messages*

  System messages go to all devices in a MIDI system because no channel numbers are specified. There are three types of system messages:

  - *System real-time messages* are very short and simple, consisting of only one byte. They carry extra data with them. These messages synchronize the timing of MIDI devices in performance; therefore, it is important that they be sent at precisely the time they are required. To avoid delays, these messages are sent in the middle of other messages, if necessary. Examples of such messages are *System Reset*, *Timing Clock (MIDI clock)*, etc.
  - *System common messages* are commands that prepare sequencers and synthesizers to play a song. The various messages enable you to select a song, find a common starting place in the song and tune all the synthesizers if they need tuning. Examples are *Song Select*, *Tune Request*, etc.
  - *System exclusive messages* allow MIDI manufacturers to create customized MIDI messages to send between their MIDI devices. This coding starts

with a *system-exclusive-message*, where the manufacturer is specified, and ends with an *end-of-exclusive message*.

3.2.4 MIDI and SMPTE Timing Standards

MIDI reproduces traditional note length using *MIDI clocks*, which are represented through *timing clock* messages. Using a MIDI clock, a receiver can synchronize with the clock cycles of the sender. For example, a MIDI clock helps keep separate sequencers in the same MIDI system playing at the same tempo. When a master sequencer plays a song, it sends out a stream of 'Timing Clock' messages to convey the tempo to other sequencers. The faster the Timing Clock messages come in, the faster the receiving sequencer plays the song. To keep a standard timing reference, the MIDI specifications state that 24 MIDI clocks equal one quarter note.

As an alternative, the *SMPTE timing standard* (Society of Motion Picture and Television Engineers) can be used. The SMPTE timing standard was originally developed by NASA as a way to mark incoming data from different tracking stations so that receiving computers could tell exactly what time each piece of data was created [Boo87]. In the film and video version promoted by the SMPTE, the SMPTE timing standard acts as a very precise clock that stamps a time reading on each frame and fraction of a frame, counting from the beginning of a film or video. To make the time readings precise, the SMPTE format consists of *hours:minutes:seconds:frames:bits* (e.g., 30 frames per second), uses a 24-hour clock and counts from 0 to 23 before recycling to 0. The number of frames in a second differs depending on the type of visual medium. To divide time even more precisely, SMPTE breaks each frame into 80 bits (not digital bits). When SMPTE is counting bits in a frame, it is dividing time into segments as small as one twenty-five hundredth of a second.

Because many film composers now record their music on a MIDI recorder, it is desirable to synchronize the MIDI recorder with video equipment. A SMPTE *synchronizer* should be able to give a time location to the MIDI recorder so it can move to that location in the MIDI *score* (pre-recorded song) to start playback or recording. But MIDI recorders cannot use incoming SMPTE signals to control their recording and playback. The solution is a MIDI/SMPTE synchronizer that converts SMPTE into MIDI, and vice versa. The MIDI/SMPTE synchronizer lets the user specify different tempos and the exact points in SMPTE timing at which each tempo is to start, change, and stop. The synchronizer keeps these tempos and timing points in memory. As a SMPTE video deck plays and sends a stream of SMPTE times to the synchronizer, the synchronizer checks the incoming time and sends out MIDI clocks at a corresponding tempo.

3.2.5 MIDI Software

Once a computer is connected to a MIDI system, a variety of MIDI applications can run on it. Digital computers afford the composer or sound designer unprecedented levels of control over the evolution and combination of sonic events.

The software applications generally fall into four major categories:

- *Music recording and performance applications*

  This category of applications provides functions such as recording of MIDI messages as they enter the computer from other MIDI devices, and possibly editing and playing back the messages in performance.

- *Musical notations and printing applications*

  This category allows writing music using traditional musical notation. The user can then play back the music using a performance program or print the music on paper for live performance or publication.

- *Synthesizer patch editors and librarians*

  These programs allow information storage of different synthesizer patches in the computer's memory and disk drives, and editing of patches in the computer.

- *Music education applications*

  These software applications teach different aspects of music using the computer monitor, keyboard and other controllers of attached MIDI instruments.

The main issue in current MIDI-based computer music systems is *interactivity.* Music is a temporal art, and any computer program dealing with music must have sophisticated facilities for representing time and for scheduling processes to occur at a particular point in time. This capability of music applications became possible because of increased computational speeds (e.g., the computational speeds needed to execute compositional algorithms in real-time are available). Therefore, current computer music systems are able to modify their behavior in response to input from other performing musicians [Row93].

The processing chain of interactive computer music systems can be conceptualized in three stages:

- The *sensing stage*, when data are collected from controllers reading gesture information from human performers on stage.
- The *processing stage*, when the computer reads and interprets information coming from the sensors and prepares data for the response stage.
- The *response stage*, when the computer and some collection of sound-producing devices share in realizing a musical output.

Commercial manufacturers dominate in providing MIDI devices, such as MIDI controllers and synthesizers, for sensing and response stages. The processing stage has commercial entries as well, most notably MIDI sequencers. It is in processing, however, that individual conceptions of interactive music are most readily expressed, in any of a variety of programming languages with temporal and MIDI extensions.

Commercial interactive music systems appeared in the mid-1980s. Two groundbreaking efforts in this field were *M* and *Jam Factory* [Zic87]. Among the breakthroughs implemented by these programs was the graphic control panel, which allowed access to the values of global variables affecting musical output. Manipulating the grapl. c controls had an immediately audible effect. The sensing performed by *M* and *Jam Factory* centered around reading manipulations of the control panel and interpreting an incoming stream of MIDI events. Responses were sent out as MIDI.

In 1990, Opcode Systems released *Max*$^{\mathrm{TM}}$, a graphical programming environment for interactive music systems. *Max* is an object-oriented programming language, in which programs are realized by manipulating graphic objects on a computer screen and making connection between them [DZ90].

An interactive computer music system, emphasizing composition and performance, is the MIT system *Cypher* [Row93]. The program has two main components: a listener and a player. The listener characterizes performances represented by streams of MIDI data, which could be coming from a human performer, another computer program or even *Cypher* itself. The player generates and plays musical material.

NeXT^{TM}Computer has a music system based on MIDI that combines the synthesis power and generality of a mainframe computer with the performance flexibility of a keyboard synthesizer. The system, *Music Kit*, helps the composer or performer construct applications that create, organize, process and render music data [JB89].

3.3 Speech

Speech can be "perceived," "understood" and "generated" by humans and also by machines. A human adjusts himself/herself very efficiently to diffrrent speakers and their speech habits. Despite different dialects and pronunciation, the speech can be well understood by humans. The brain can recognize the very fine line between speech and noise. For this purpose, both ears are used, because filtering with only one ear is substantially more difficult for the listener. The human speech signal comprises a subjective lowest spectral component known as the *pitch*, which is not proportional to frequency. The human ear is most sensitive in the range from 600 Hz to 6000 Hz. Fletscher and Munson have shown that the human ear is substantially less sensitive to low and very high frequencies than to frequencies around 1 kHz. Speech signals have two properties which can be used in speech processing:

- Voiced speech signals show during certain time intervals almost periodic behavior. Therefore, we can consider these signals as *quasi-stationary signals* for around 30 milliseconds.
- The spectrum of audio signals shows characteristic maxima, which are mostly 3-5 frequency bands. These maxima, called *formants*, occur because of resonances of the vocal tract.

For description and modeling of human speech generation, see [All85, BN93].

A machine can also support speech generation and recognition. With computers, one can synthetically generate speech, where the generated signals do not sound quite natural but can be easily understood. An example of such an artificial sounding voice can be heard at the Atlanta (Georgia, USA) airport. On the other hand, a voice can sound natural but may be very difficult to understand. Speech recognition often uses matching rules or statistically based methods. Today, workstations and personal computers can recognize 25,000 possible words. Problems are caused when dialects, emotional pronunciation and environmental noises are part of the audio signal. There are, and will continue to be in the near future, considerable differences between the speech generation and recognition efficiencies/capabilities of the human brain and a high-performance computer [Ace93, Mam93].

In the following two sections we describe in more detail some crucial issues of computer-generated speech and recognition.

3.3.1 Speech Generation

Speech generation research has a long history. By the middle of the 19th century, Helmholtz had already built a mechanical vocal tract coupling together several mechanical resonators with which sound could be generated. In 1940, Dudley produced the first speech synthesizer through imitation of mechanical vibration using electrical oscillation [Fal85].

An important requirement for speech generation is *real-time signal generation.* With such a requirement met, a speech output system could transform text into speech automatically without any lengthy preprocessing. Some applications only need a limited vocabulary; an example is the spoken time announcement of a telephone answering service. However, most applications need a large vocabulary, if not an unlimited vocabulary.

Generated speech must be *understandable* and must sound *natural.* The requirement of understandable speech is a fundamental assumption, and the natural sound of speech increases user acceptance.

Basic Notions

For further discussion we introduce some notions of importance:

- The lowest periodic spectral component of the speech signal is called the *fundamental frequency.* It is present in a voiced sound.
- A *phone* is the smallest speech unit, such as the *m* of *mat* and the *b* of *bat* in English, that distinguish one utterance or word from another in a given language.
- *Allophones* mark the variants of a phone. For example, the aspirated *p* of *pit* and the unaspirated *p* of *spit* are allophones of the English phoneme *p.*
- The *morph* marks the smallest speech unit which carries a meaning itself. Therefore, *consider* is a morph, but *reconsideration* is not.
- A *voiced sound* is generated through the vocal cords. *m*, *v* and *l* are examples of voiced sounds. The pronunciation of a voiced sound depends strongly on each speaker.
- During the generation of an *unvoiced sound,* the vocal cords are opened. *f* and *s* are unvoiced sounds. Unvoiced sounds are relatively independent from the speaker.

Exactly, there are:

- Vowels – a speech sound created by the relatively free passage of breath through the larynx and oral cavity, usually forming the most prominent and central sound of a syllable (e.g., *u* from hunt);
- Consonants – a speech sound produced by a partial or complete obstruction of the air stream by any of the various constrictions of the speech organs (e.g., voiced consonants, such as *m* from *mother*, fricative voiced consonants, such as *v* from *voice*, fricative voiceless consonants, such as *s* from *nurse*, plosive consonants, such as *d* from *daily* and affricate consonants, such as *dg* from *knowledge*, or *ch* from *chew*).

Reproduced Speech Output

The easiest method of speech generation/output is to use prerecorded speech and play it back in a timely fashion [BN93]. The speech can be stored as PCM (Pulse Code Modulation) samples. Further data compression methods, without using language typical properties, can be applied to recorded speech (see Chapter 6).

Time-dependent Sound Concatenation

Speech generation/output can also be performed by sound concatenation in a timely fashion [Ril89]. Individual speech units are composed like building blocks, where the composition can occur at different levels. In the simplest case, the individual phones are understood as speech units. Figure 3.4 shows the individual phones of the word *crumb*. It is possible with just a few phones to create an unlimited vocabulary.

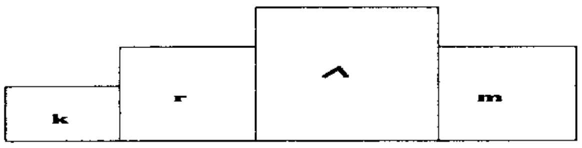

Figure 3.4: *Phone sound concatenation.*

However, transitions between individual phones prove to be extremely problematic. Therefore, the phones in their environment, i.e., the allophones, are considered in the second level. But the transition problem is not solved sufficiently on this level either. Two phones can constitute a diphone (from di-phone). Figure 3.5 shows the word *crumb*, which consists of an ordered set of diphones.

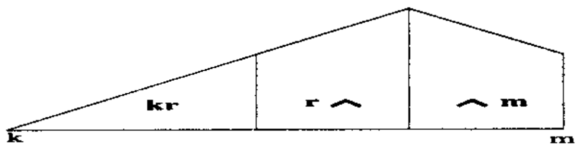

Figure 3.5: *Diphone sound concatenation.*

To make the transition problem easier, syllables can be created. The speech is generated through the set of syllables. Figure 3.6 shows the syllable sound of the word *crumb*. The best pronunciation of a word is achieved through storage of the

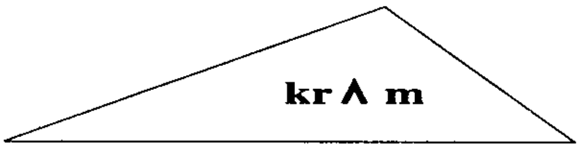

Figure 3.6: *Syllable sound.*

whole word. This leads toward synthesis of the speech sequence (Figure 3.7).

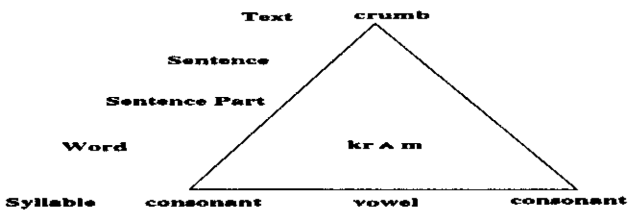

Figure 3.7: *Word sound concatenation.*

Transitions between individual sound units create an essential problem, called *coarticulation*, which is the mutual sound influence throughout several sounds. This influence between individual sound units arises because physical constraints, such as mass and speed of the articulator in the vocal tract, influence the articulation of consecutive phones.

Additionally, *prosody* should be considered during speech generation/output. Prosody means the stress and melody course. For example, pronunciation of a question differs strongly from a statement. Therefore, prosody depends on the semantics of the speech and this has to be taken into consideration during time-dependent sound concatenation [Wai88].

Frequency-dependent Sound Concatenation

Speech generation/output can also be based on a frequency-dependent sound concatenation, e.g., through a formant-synthesis [Ril89]. Formants are frequency maxima in the spectrum of the speech signal. Formant synthesis simulates the vocal tract through a filter. The characteristic values are the filter's middle frequencies and their bandwidths. A pulse signal with a frequency, corresponding to the fundamental speech frequency, is chosen as a simulation for voiced sounds. On the other hand, unvoiced sounds are created through a noise generator.

Individual speech elements (e.g., phones) are defined through the characteristic values of the formants. Similar problems to the time-dependent sound concatenation exist here. The transitions, known as *coarticulation*, present the most critical problem. Additionally, the respective *prosody* has to be determined.

New sound-specific methods provide a sound concatenation with combined time and frequency dependencies. Initial results show that new methods generate fricative and plosive sounds with higher quality.

Human speech can be generated using a multi-pole lattice filter. The first four or five formants, occurring in human speech are modeled correctly with this filter type. Further, unvoiced sounds, created by vocal chords, are simulated through a noise and tone generator. The method used for the sound synthesis in order to simulate human speech is called the *Linear-Predictive Coding (LPC) method.* This method is very similar to the formant synthesis described above. A further possibility to simulate human speech consists of implementing a *tube model.* Here, a simplified mechanical tube model approximates the human tube as the speech generation system.

Using speech synthesis, an existent text can be transformed into an acoustic signal. Figure 3.8 shows the typical components of the system. In the first step, *transcrip-*

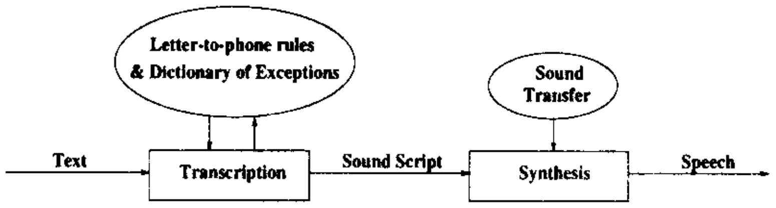

Figure 3.8: *Components of a speech synthesis system with time-dependent sound concatenation.*

tion is performed, in which text is translated into sound script. Most transcription methods work here with *letter-to-phone rules* and a *Dictionary of Exceptions* stored in a library. The generation of such a library is work-extensive, but using the interactive control of the user it can be improved continuously. The user recognizes the formula deficiency in the transcription and improves the pronunciation manual; therefore, his/her knowledge becomes part of the letter-to-phone rules and the Dictionary of Exceptions. The solution can be either individual or generally accessible rules and a Dictionary of Exceptions.

In the second step, the sound script is translated into a speech signal. Time or frequency-dependent concatenation can follow. While the first step is always a software solution, the second step is most often implemented with signal processors or even dedicated processors.

Besides the problems of coarticulation and prosody, ambiguous pronunciation must be considered. Pronunciation can be performed correctly only with additional knowledge of the content, i.e., it is semantic-dependent. An example is the word *lead.* It can be used as a noun to describe a metal, but when used as a verb (with a different pronunciation as the noun) it means "to guide people."

3.3.2 Speech Analysis

Speech analysis/input deals with the research areas shown in Figure 3.9 [Bri86]:

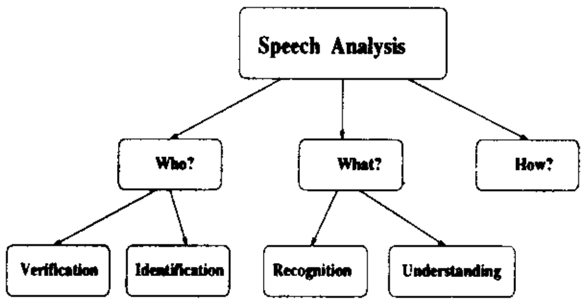

Figure 3.9: *Research areas of speech analysis.*

- Human speech has certain characteristics determined by a speaker. Hence, speech analysis can serve to analyze *who* is speaking, i.e., to *recognize a speaker* for his/her *identification* and *verification.* The computer identifies and verifies the speaker using an acoustic fingerprint. An acoustic fingerprint is a digitally stored speech probe (e.g., certain statement) of a person; for example, a company that uses speech analysis for identification and verification of employees. The speaker has to say a certain sentence into a microphone. The computer system gets the speaker's voice, identifies and verifies the spoken statement, i.e., determines if the speech probe matches the speaker's spoken statement.

- Another main task of speech analysis is to analyze *what* has been said, i.e., to recognize and understand the speech signal itself. Based on speech sequence, the corresponding text is generated. This can lead to a speech-controlled typewriter, a translation system or part of a workplace for the handicapped.
- Another area of speech analysis tries to research speech patterns with respect to *how* a certain statement was said. For example, a spoken sentence sounds differently if a person is angry or calm. An application of this research could be a lie detector.

Speech analysis is of strong interest for multimedia systems. Together with speech synthesis, different media transformations can be implemented.

The primary goal of speech analysis is to correctly determine individual words with probability ≤ 1. A word is recognized only with a certain probability. Here, environmental noise, room acoustics and a speaker's physical and psychological conditions play an important role.

For example, let's assume extremely bad individual word recognition with a probability of 0.95. This means that 5% of the words are incorrectly recognized. If we have a sentence with three words, the probability of recognizing the sentence correctly is $0.95 \times 0.95 \times 0.95 = 0.857$. This small example should emphasize the fact that speech analysis systems should have a very high individual word recognition probability. Figure 3.10 shows schematically a speech recognition system. The system is divided into system components according to a basic principle: "Data Reduction Through Property Extraction". First, speech analysis occurs where properties must be determined. Properties are extracted by comparison of individual speech element characteristics with a sequence of in advance given speech element characteristics. The characteristics are quantified where the concrete speech elements are present.

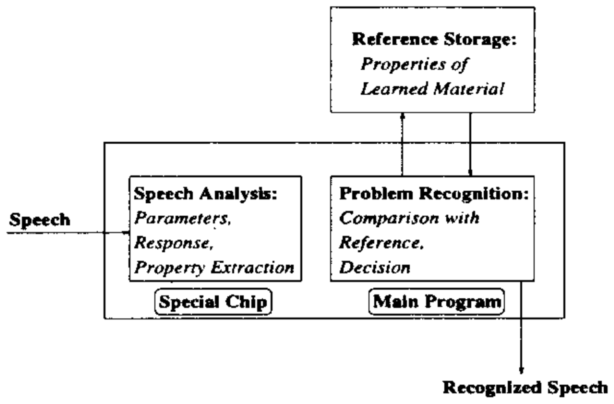
Reference Storage:
Properties of Learned Material
Speech
Speech Analysis:
Parameters, Response, Property Extraction
Special Chip
Problem Recognition:
Comparison with Reference, Decision
Main Program
Recognized Speech

Figure 3.10: *Speech recognition system: task division into system components, using the basic principle "Data Reduction Through Property Extraction."*

Second, the speech elements are compared with existent references to determine the mapping to one of the existent speech elements. The identified speech can be stored, transmitted or processed as a parametrized sequence of speech elements.

Concrete implementations mostly use dedicated building blocks or signal processors for characteristics extraction. Usually, the comparison and decision are executed through the main system processor. The computer's secondary storage contains the letter-to-phone rules, a Dictionary of Exceptions and the reference characteristics. The concrete methods differ in the definition of the characteristics. The principle of "Data Reduction Through Property Extraction," shown in Figure 3.10, can be applied several times to different characteristics. The system which provides recognition and understanding of a speech signal (Figure 3.11) applies this principle several times as follows:

- In the first step, the principle is applied to a sound pattern and/or word model. An acoustical and phonetical analysis is performed.
- In the second step, certain speech units go through syntactical analysis; thereby, the errors of the previous step can be recognized. Very often during the first step, no unambiguous decisions can be made. In this case, syntactical analysis provides additional decision help and the result is a *recognized speech.*

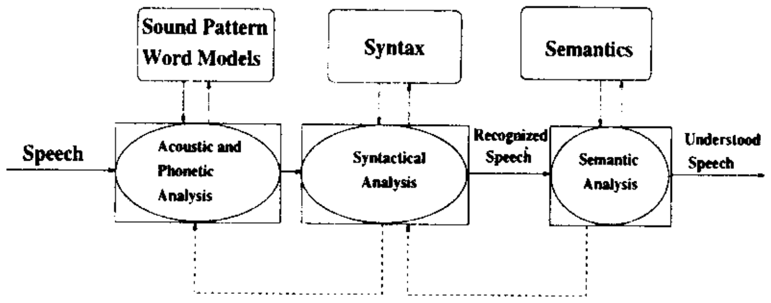

Figure 3.11: *Components of speech recognition and understanding.*

- The third step deals with the semantics of the previously recognized language. Here the decision errors of the previous step can be recognized and corrected with other analysis methods. Even today, this step is non-trivial to implement with current methods known in artificial intelligence and neural nets research. The result of this step is an *understood speech.*

These steps work mostly under the consideration of time and/or frequency-dependent sounds. The same criteria and speech units (formants, phones, etc.) are considered as in speech generation/output (discussed in Section 3.3.1).

There are still many problems into which speech recognition research is being conducted:

- A specific problem is presented by room acoustics with existent environmental noise. The frequency-dependent reflections of a sound wave from walls and objects can overlap with the primary sound wave.
- Further, word boundaries must be determined. Very often neighboring words flow into one another.
- For the comparison of a speech element to the existing pattern, time normalization is necessary. The same word can be spoken quickly or slowly. However, the time axis cannot be modified because the extension factors are not proportional to the global time interval. There are long and short voiceless sounds (e.g., *s*, *sh*). Individual sounds are extended differently and need a minimal time duration for their recognition.

Speech recognition systems are divided into *speaker-independent recognition systems* and *speaker-dependent recognition systems*. A speaker-independent system can recognize with the same reliability essentially fewer words than a speaker-dependent system because the latter is trained in advance. Training in advance means that there exists a training phase for the speech recognition system, which takes a half an hour. Speaker-dependent systems can recognize around 25,000 words; speaker-independent systems recognize a maximum of about 500 words, but with a worse recognition rate. These values should be understood as gross guidelines. In a concrete situation, the marginal conditions must be known. (e.g., Was the measurement taken in a sound deadening room?, Does the speaker have to adapt to the system to simplify the time normalization?, etc.)

3.3.3 Speech Transmission

The area of speech transmission deals with efficient coding of the speech signal to allow speech/sound transmission at low transmission rates over networks. The goal is to provide the receiver with the same speech/sound *quality* as was generated at the sender side. This section includes some principles that are connected to speech generation and recognition.

- *Signal Form Coding*

  This kind of coding considers no speech-specific properties and parameters. Here, the goal is to achieve the most efficient coding of the audio signal. The data rate of a PCM-coded stereo-audio signal with *CD-quality* requirements is:

  $rate = 2 * \frac{44100}{s} * \frac{16bit}{8bit/byte} = 176,400\ bytes/s = 1,411,200\ bits/s$

  Telephone quality, in comparison to CD-quality, needs only 64 Kbit/s. Using *Difference Pulse Code Modulation (DPCM)*, the data rate can be lowered to 56 Kbits/s without loss of quality. *Adaptive Pulse Code Modulation (ADPCM)* allows a further rate reduction to 32 Kbits/s.

- *Source Coding*

Parameterized systems work with source coding algorithms. Here, the specific speech characteristics are used for data rate reduction.

Channel vo-coder is an example of such a parameterized system (Figure 3.12). The channel vo-coder is an extension of a sub-channel coding. The signal is

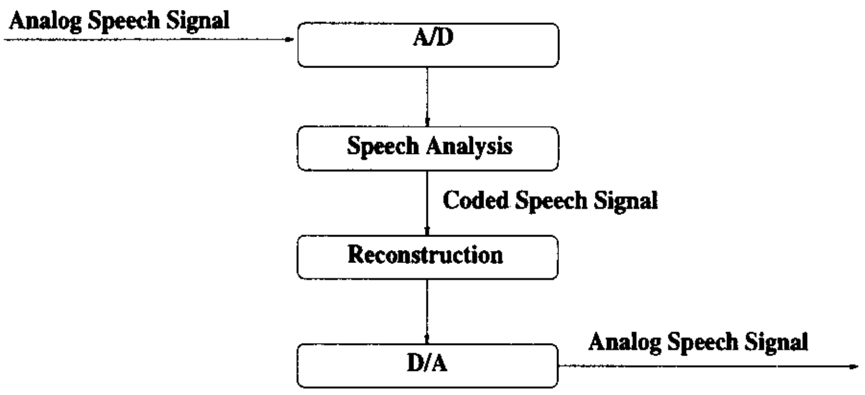

Figure 3.12: *Source coding in parametrized systems: components of a speech transmission system.*

divided into a set of frequency channels during speech analysis because only certain frequency maxima are relevant to speech. Additionally, the differences between voiced and unvoiced sounds are taken into account. Voiceless sounds are simulated by the noise generator. For generation of voiced sounds, the simulation comes from a sequence of pulses. The rate of the pulses is equivalent to the a priori measured basic speech frequency. The data rate of about 3 Kbits/s can be generated with a channel vo-coder; however the quality is not always satisfactory.

Major effort and work on further data rate reduction from 64 Kbits/s to 6 Kbits/s is being conducted, where the compressed signal quality should correspond, after a decompression, to the quality of an uncompressed 64 Kbits/s signal.

- *Recognition/Synthesis Methods*

  There have been attempts to reduce the transmission rate using pure *recognition/synthesis methods*. Speech analysis (recognition) follows on the sender side of a speech transmission system and speech synthesis (generation) follows on the receiver side (see Figure 3.13).

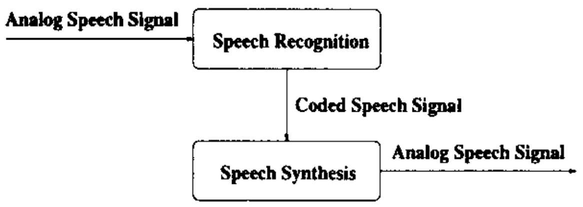

Figure 3.13: *Recognition/synthesis systems: components of a speech transmission system.*

Only the characteristics of the speech elements are transmitted. For example, the speech elements with their characteristics are the formants with their middle frequency bandwidths. The frequency bandwidths are used in the corresponding digital filter. This reduction brings the data rate down to 50 bits/s. The quality of the reproduced speech and its recognition rate are not acceptable by today's standards.

- *Achieved Quality*

  The essential question regarding speech and audio transmission with respect to multimedia systems is how to achieve the minimal data rate for a given quality. The published function from Flanagan [Fla72] (see Figure 3.14) shows the dependence of the achieved quality of compressed speech on the data rate. One can assume that for telephone quality, a data rate of 8 Kbits/s is sufficient.

  Figure 3.15 shows the dependence of audio quality on the number of bits per sample value. For example, excellent CD-quality can be achieved with a reduction from 16 bits per sample value to 2 bits per sample value. This means that only 1/8 of the actual data needs to be transmitted.

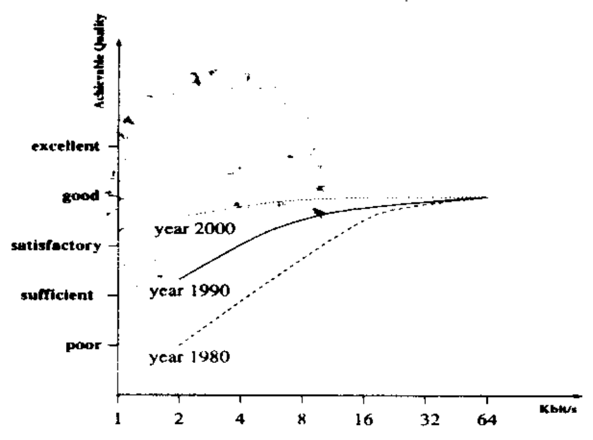

Figure 3.14: *Dependence of the achieved speech quality on the data rate.*

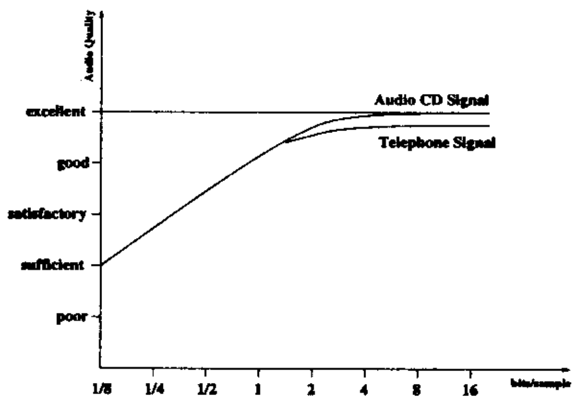

Figure 3.15: *Dependence of audio quality on the number of bits per sample value.*

Chapter 4

Images and Graphics

An *image* is a spatial representation of an object, a two-dimensional or three-dimensional scene or another image. It can be real or virtual. An image may be abstractly thought of as a continuous function defining usually a rectangular region of a plane. For example, for optic or photographic sensors, an image is typically proportional to the radiant energy received in the electromagnetic band to which the sensor or detector is sensitive. In this case, the image is called an *intensity image*. For range finder sensors, an image is a function of the line-of-sight distance from the sensor position to an object in the three-dimensional world; such an image is called a *range image*. For tactile sensors, an image is proportional to the sensor deformation caused by the surface of or around an object.

A recorded image may be in a photographic, analog video signal or digital format. In computer vision, an image is usually a recorded image such as a video image, digital image or a picture. In computer graphics, an image is always a digital image. In multimedia applications, all formats can be presented.

In this chapter, the digital image will be discussed; this is the computer representation most important for processing in multimedia systems. We present basic concepts of image representation. Further, computer processing of images is described with an overview of topics on image generation, recognition and transmission.

4.1 Basic Concepts

An image might be thought of as a function with resulting values of the light intensity at each point over a planar region. For digital computer operations, this function needs to be sampled at discrete intervals. The sampling quantizes the intensity values into discrete levels.

4.1.1 Digital Image Representation

A digital image is represented by a matrix of numeric values each representing a quantized intensity value. When I is a two-dimensional matrix, then $I(r,c)$ is the intensity value at the position corresponding to row r and column c of the matrix.

The points at which an image is sampled are known as *picture elements*, commonly abbreviated as *pixels.* The pixel values of intensity images are called *gray scale levels* (we encode here the "color" of the image). The intensity at each pixel is represented by an integer and is determined from the continuous image by averaging over a small neighborhood around the pixel location. If there are just two intensity values, for example, black and white, they are represented by the numbers 0 and 1; such images are called *binary-valued* images. When 8-bit integers are used to store each pixel value, the gray levels range from 0 (black) to 255 (white). An example of such an image is shown in Figure 4.1.

It is common to use a square sampling grid with pixels equally spaced along the two sides of the grid. The distance between grid points obviously affects the accuracy with which the original image is represented, and it determines how much detail can be resolved. The *resolution* depends on the imaging system as well.

Digital pictures are often very large. For example, suppose we want to sample and quantize an ordinary (525-line) television picture (NTSC) with a VGA (Video Graphics Array) video controller, so that it can be redisplayed without noticeable degradation. We must use a matrix of 640×480 pixels, where each pixel is represented by an 8-bit integer. This pixel representation allows 256 discrete gray levels. Hence, this image specification gives an array of 307,200 8-bit numbers, and a total of 2,457,600 bits. In many cases, even finer sampling is necessary.

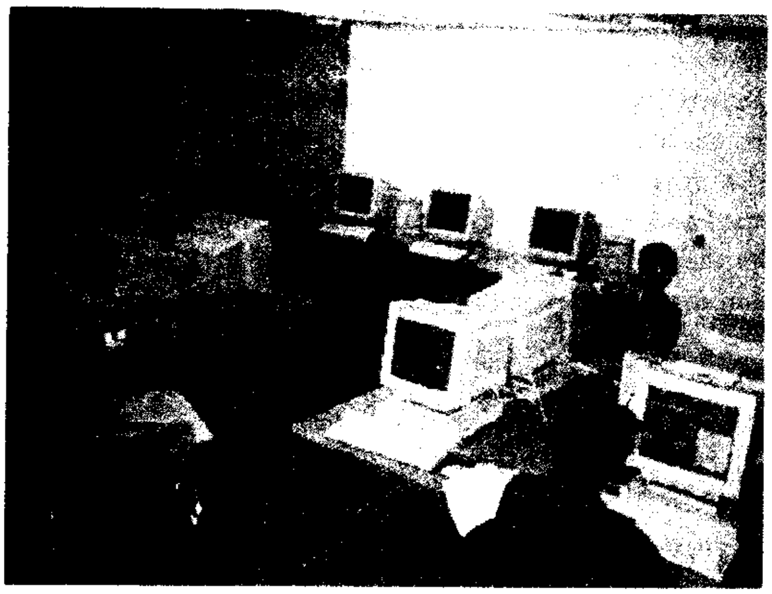

Figure 4.1: *An example of an image with 256 gray levels.*

4.1.2 Image Format

There are different kinds of image formats in the literature. We shall consider the image format that comes out of an image frame grabber, i.e., the *captured image format*, and the format when images are stored, i.e., the *stored image format.*

Captured Image Format

The image format is specified by two main parameters: *spatial resolution,* which is specified as *pixels* x *pixels* and *color encoding,* which is specified by bits per pixel. Both parameter values depend on hardware and software for input/output of images. As an example we will present image formats supported on SPARC and IRIS computers.

For image capturing on a SPARCstation, the *VideoPix*$^{\mathrm{TM}}$card and its software [Sun90] can be used. The spatial resolution is 320 x 240 pixels. The color can be encoded with 1-bit (a binary image format), 8-bit (color or grayscale) or 24-bit (color-*RGB*). Another video frame grabber is the *Parallax XVideo*, which includes a 24-bit frame buffer and 640 × 480 pixels resolution. The new multimedia kit in the new SPARCstations includes the *SunVideo*™card (Sun Microsystems, Inc.), a color video camera and the CDware for CD-ROM discs. The SPARCstation 10 M offers 24-bit image manipulation. The new *SunVideo* card is a capture and compression card and its technology captures and compresses 30 frames/second in real time under the Solaris 2.3 operating system. Further, *SunVideo* offers capture and compression of video at resolution (320 × 240) pixels in several formats [Moo94]:

CellB	30 fps (frames per second)
JPEG	30 fps
MPEG1 I frames	30 fps
MPEG1 IP frames	17 fps
Capture YUV	30 fps
Capture RGB-8	30 fps
Capture RGB-24	12 fps

IRIS™stations provide high-quality images through, for example, add-on *IndigoVideo* or *IndyVideo*™video digitizers and corresponding software [Roy94]. The IRIS video board *VINO*™is supported only on Indy™systems. It does not include image compression. *VINO* offers image resolution of 640 × 480 pixels at about four frames per second. Speed can be increased at the cost of resolution. The resulting formats are *RGB* and *YUV* formats (described in Section 5.1.1.).

Stored Image Format

When we store an image, we are storing a two-dimensional array of values, in which each value represents the data associated with a pixel in the image. For a bitmap, this value is a binary digit. For a color image, the value may be a collection of:

- Three numbers representing the intensities of the red, green and blue components of the color at that pixel.
- Three numbers that are indices to tables of red, green and blue intensities.

- A single number that is an index to a table of color triples.
- An index to any number of other data structures that can represent a color, including XYZ color system.
- Four or five spectral samples for each color.

In addition, each pixel may have other information associated with it; for example, three numbers indicating the normal to the surface drawn at that pixel. Thus, we consider an image as consisting of a collection of *Red, Green and Blue channels* (*RGB* channels), each of which gives some single piece of information about the pixels in the image.

If there is enough storage space, it is convenient to store an image in the form of *RGB* triples. Otherwise, it may be worth trying to compress the channels in some way. When we store an image in the conventional manner, as a collection of channels, information about each pixel, i.e., the value of each channel at each pixel, must be stored. Other information may be associated with the image as a whole, such as width and height, as well as the depth of the image, the name of the creator etc. The need to store such properties has prompted the creation of flexible formats such as RIFF (Resource Interchange File Format) and BRIM (derived from RIFF) [Mei83], which are used in general attribute value database systems. RIFF includes formats for bitmaps, vector drawings, animation, audio and video. In BRIM, an image always has a width, height, creator and history field, which describes the creation of the image and modifications to it.

Some current image file formats for storing images include GIF (Graphical Interchange Format), X11 Bitmap, Sun Rasterfile, PostScript, IRIS, JPEG, TIFF (Tagged Image File Format) and others.

4.1.3 Graphics Format

Graphics image formats are specified through *graphics primitives* and their *attributes.* To the category of graphics primitives belong lines, rectangles, circles and ellipses, text strings specifying two-dimensional objects (2D) in a graphical image or, e.g., polyhedron, specifying three-dimensional objects (3D). A graphics package determines which primitives are supported. Attributes such as line style, line width, and color affect the outcome of the graphical image.

Graphics primitives and their attributes represent a higher level of an image representation, i.e., the graphical images are not represented by a pixel matrix. This higher level of representation needs to be converted at some point of the image processing into the lower level of the image representation; for example, when an image is to be displayed. The advantage of the higher level primitives is the reduction of data to be stored per one graphical image, and easier manipulation of the graphical image. The disadvantage is the additional conversion step from the graphical primitives and their attributes to its pixel representation. Some graphics packages like SRGP (Simple Raster Graphics Package) provide such a conversion, i.e., they take the graphics primitives and attributes and generate either a *bitmap* or *pixmap*. A bitmap is an array of pixel values that map one by one to pixels on the screen; the pixel information is stored in 1 bit, so we get a binary image. Pixmap is a more general term describing a multiple-bit-per-pixel image. Low-end color systems have eight bits per pixel, allowing 256 colors simultaneously. More expensive systems have 24 bits per pixel, allowing a choice of any of 16 million colors. Refresh buffers with 32 bits per pixel and a screen resolution of 1280×1024 pixels are available even on personal computers. Of the 32 bits per pixel, 24 bits are devoted to representing color and 8 bits to control purposes. Beyond that, buffers with 96 bits (or more) per pixel are available at 1280×1024 resolution on high-end systems [FDFH92]. SRGP does not convert the graphical image into primitives and attributes after generating a bitmap/pixmap. In this case, after the conversion phase, the graphics format is presented as a digital image format.

Packages such as PHIGS (Programmer's Hierarchical Interactive Graphics System) and GKS (Graphical Kernel System) [FDFH92] take graphical images specified through primitives and attributes, generate a graphical image in the form of pixmap and after image presentation, continue to work based on the object primitive/attribute representation. In this case, the graphical image format is presented after the generation phase as a *structure*, which is a logical grouping of primitives, attributes and other information.

4.2 Computer Image Processing

Computer graphics concern the pictorial *synthesis* of real or imaginary objects from their computer-based models. The related field of image processing treats the converse process: the *analysis* of scenes, or the *reconstruction* of models from pictures of 2D or 3D objects. In the following sections, we describe basic principles of image synthesis (generation) and image analysis (recognition). The literature on computer graphics and image processing presents further and detailed information [FDFH92, KR82, Nev82, HS92].

4.2.1 Image Synthesis

Image synthesis is an integral part of all computer user interfaces and is indispensable for visualizing 2D, 3D and higher-dimensional objects. Areas as diverse as education, science, engineering, medicine, advertising and entertainment all rely on graphics. Let us look at some representative samples:

- *User Interfaces*

  Applications running on personal computers and workstations have user interfaces that rely on desktop window systems to manage multiple simultaneous activities, and on point-and-click facilities to allow users to select menu items, icons and objects on the screen.

- *Office Automation and Electronic Publishing*

  The use of graphics for the creation and dissemination of information has increased enormously since the advent of desktop publishing on personal computers. Office automation and electronic publishing can produce both traditional printed documents and electronic documents that contain text, tables, graphs and other forms of drawn or scanned-in graphics. Hypermedia systems that allow browsing networks of interlinked multimedia documents are proliferating.

- *Simulation and Animation for Scientific Visualization and Entertainment*

Computer-produced animated movies and displays of time-varying behavior of real and simulated objects are becoming increasingly popular for scientific and engineering visualization. We can use them to study abstract mathematical entities and models of such phenomena as fluid flow, relativity and nuclear and chemical reactions. Cartoon characters will increasingly be modeled in computers as 3D shape descriptions whose movements are controlled by computers rather than by the figures drawn manually by cartoonists. Television commercials featuring flying logos and more exotic visual trickery have become common, as have elegant special effects in movies.

Interactive computer graphics are the most important means of producing images (pictures) since the invention of photography and television; it has the added advantage that we can make pictures not only of concrete, "real world" objects, but also of abstract, synthetic objects such as mathematical surfaces in 4D.

Dynamics in Graphics

Graphics are not confined to static pictures. Pictures can be dynamically varied; for example, a user can control animation by adjusting the speed, portion of the total scene in view, amount of detail shown, etc. Hence, dynamics is an integral part of graphics (*dynamic graphics*). Much of interactive graphics technology contains hardware and software for user-controlled *motion dynamics* and *update dynamics*:

- Motion Dynamics

  With motion dynamics, objects can be moved and enabled with respect to a stationary observer. The objects can also remain stationary and the view around them can move. In many cases, both the objects and the camera are moving. A typical example is a flight simulator which contains a mechanical platform, which supports a mock cockpit and a display screen. The computer controls platform motion, gauges and the simulated world of both stationary and moving objects through which the pilot navigates.

- Update Dynamics

Update dynamics is the actual change of the shape, color, or other properties of the objects being viewed. For instance, a system can display the deformation of an in-flight airplane structure in response to the operator's manipulation of the many control mechanisms. The smoother the change, the more realistic and meaningful the result. Dynamic, interactive graphs offer a large number of user-controllable modes with which to encode and communicate information, e.g., the 2D or 3D shape of objects in a picture, their gray scale or color and the time variations of these properties.

The Framework of Interactive Graphics Systems

Images can be generated by video digitizer cards that capture NTSC (PAL) analog signals and create a digital image. These kinds of digital images are used, for example, in image processing for image recognition and in communication for video conferencing. In this section we concentrate on image generation via graphics systems. We discuss in more detail image and video generation via video digitizers in Chapter 5.

Graphical images are generated using interactive graphics systems. An example of such a graphics system is SRGP, which borrows features from Apple's *QuickDraw* integer raster graphics package [RHA$^+$85] and MIT's *X Window System*™[SGN88] for output, and from GKS and PHIGS for input. The high-level conceptual framework of almost any interactive graphics system consists of three software components: an ***application model***, an ***application program*** and a ***graphics system***, and a hardware component: ***graphics hardware***.

The application model represents the data or objects to be pictured on the screen; it is stored in an application database. The model typically stores descriptions of primitives that define the shape of components of the object, object attributes and connectivity relationships that describe how the components fit together. The model is application-specific and is created independently of any particular display system. Therefore, the application program must convert a description of the portion of the model to whatever procedure calls or commands the graphics system uses to create an image. This conversion process has two phases. First, the application program traverses the application database that stores the model to extract the portions to be viewed, using some selection or query system. Second, the extracted geometry is put in a format that can be sent to the graphics system.

The application program handles user input. It produces views by sending to the third component, the graphics system, a series of graphics output commands that contain both a detailed geometric description of *what* is to be viewed and the attributes describing *how* the objects should appear.

The graphics system is responsible for actually producing the picture from the detailed descriptions and for passing the user's input to the application program for processing. The graphics system is thus an intermediary component between the application program and the display hardware. It effects an *output transformation* from objects in the application model to a view of the model. Symmetrically, it effects an *input transformation* from user actions to application program inputs that cause the application to make changes in the model and/or picture. The graphics system typically consists of a set of output subroutines corresponding to various primitives, attributes and other elements. These are collected in a *graphics subroutine library* or *package*. The application program specifies geometric primitives and attributes to these subroutines, and the subroutines then drive the specific display device and cause it to display the image.

At the hardware level, a computer receives input from interaction devices and outputs images to display devices.

To connect this high-level conceptual framework to our model shown in Figure 1.1, the application model and application program may represent applications, as well as the user interface part in Figure 1.1. The graphics system represents programming abstractions with support from the operating system to connect to the graphics hardware. The graphics hardware belongs to the device area in Figure 1.1; therefore, this is the main focus of the follwoing discussion.

Graphics Hardware – Input

Current input technology provides us with the ubiquitous mouse, the data tablet and the transparent, touch-sensitive panel mounted on the screen. Even fancier input devices that supply, in addition to (x, y) screen location, 3D and higher-dimensional input values (degrees of freedom), are becoming common, such as *track-balls*, *space-balls* or the *data glove*.

Track-balls can be made to sense rotation about the vertical axis in addition to that about the two horizontal axes. However, there is no direct relationship between hand movements with the device and the corresponding movement in 3D space.

A space-ball is a rigid sphere containing strain gauges. The user pushes or pulls the sphere in any direction, providing 3D translation and orientation. In this case, the directions of movement correspond to the user's attempts to move the rigid sphere, although the hand does not actually move.

The data glove records hand position and orientation as well as finger movements. It is a glove covered with small, lightweight sensors. Each sensor is a short length of fiber-optic cable, with a Light-Emitting Diode (LED) at one end and a photo-transistor at the other. In addition, a Polhelmus *3SPACE* three-dimensional position and orientation sensor records hand movements. Wearing the data glove, a user can grasp objects, move and rotate them and then release them, thus providing very natural interaction in 3D [ZLB+87].

Audio communication also has exciting potential since it allows hand-free input and natural output of simple instructions, feedback, and so on.

Graphics Hardware – Output

Current output technology uses *raster displays*, which store display primitives in a refresh buffer in terms of their component pixels. The architecture of a raster display is shown in Figure 4.2. In some raster displays, there is a hardware display controller that receives and interprets sequences of output commands. In simpler, more common systems (Figure 4.2), such as those in personal computers, the display controller exists only as a software component of the graphics library package, and the refresh buffer is no more than a piece of the CPU's memory that can be read by the image display subsystem (often called the *video controller*) that produces the actual image on the screen.

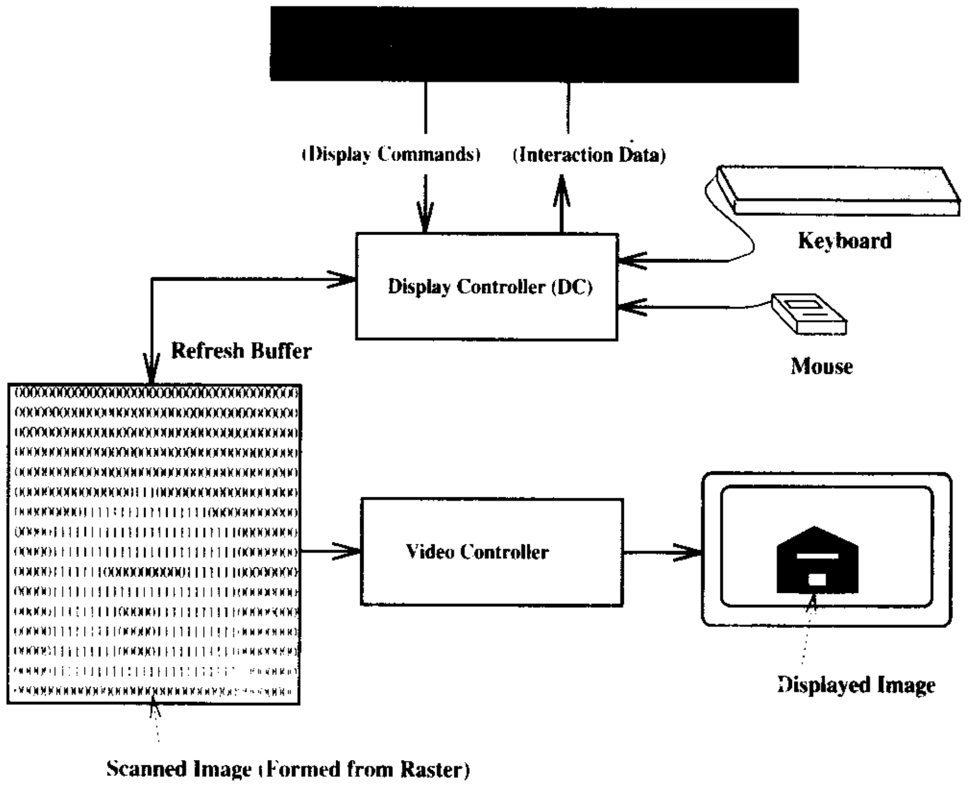

Figure 4.2: *Architecture of a raster display.*

The complete image on a raster display is formed from the *raster*, which is a set of horizontal *raster lines*, each a row of individual pixels; the raster is thus stored as a matrix of pixels representing the entire screen area. The entire image is scanned out sequentially by the video controller. The raster scan is shown in Figure 4.3. At each pixel, the beam's intensity is set to reflect the pixel's intensity; in color systems, three beams are controlled – one for each primary color (red, green, blue) – as specified by the three color components of each pixel's value (see Section 5.1).

Raster graphics systems have other characteristics. To avoid flickering of the image, a 60 Hz or higher refresh rate is used today; an entire image of 1024 lines of 1024 pixels each must be stored explicitly and a bitmap or pixmap is generated.

Raster graphics can display areas filled with solid colors or patterns, i.e., realistic

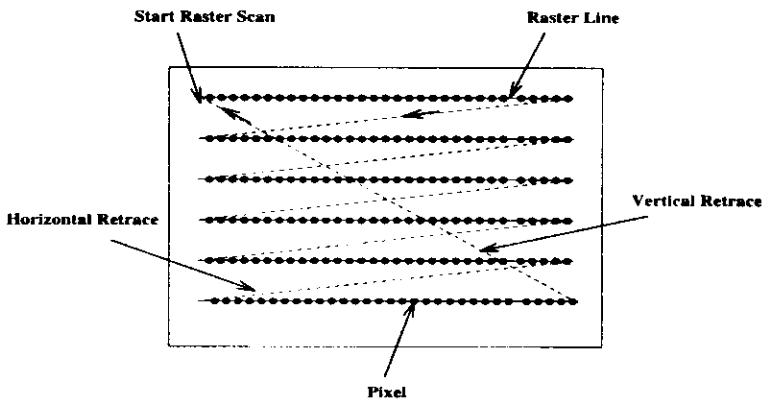

Figure 4.3: *Raster scan.*

images of 3D objects. Furthermore, the refresh process is independent of the image complexity (number of polygons, etc.) since the hardware is fast enough to read out each pixel in the buffer on each refresh cycle.

Dithering

The growth of raster graphics has made color and grayscale an integral part of contemporary computer graphics. The color of an object depends not only on the object itself, but also on the light source illuminating it, on the color of the surrounding area and on the human visual system. What we see on a black-and-white television set or display monitor is *achromatic light*. Achromatic light is determined by the attribute *quality of light*. Quality of light is determined by the intensity and luminance parameters. For example, if we have hardcopy devices or displays which are only *bi-leveled*, which means they produce just two intensity levels, then we would like to expand the range of available intensity.

The solution lies in our eye's capability for *spatial integration*. If we view a very small area from a sufficiently large viewing distance, our eyes average fine detail within the small area and record only the overall intensity of the area. This phenomenon is exploited in the technique called *halftoning*, or *clustered-dot ordered dithering* (halftoning approximation). Each small resolution unit is imprinted with a circle of black ink whose area is proportional to the blackness $1 - I$ (I=intensity) of the area in the original photograph. Graphics output devices can approximate the variable-area circles of halftone reproduction. For example, a 2×2 pixel area of a bi-level display can be used to produce five different intensity levels at the cost of halving the spatial resolution along each axis. The patterns, shown in Figure 4.4, can be filled by 2×2 areas, with the number of 'on' pixels proportional to the desired intensity. The patterns can be represented by the *dither matrix*. This technique is used on

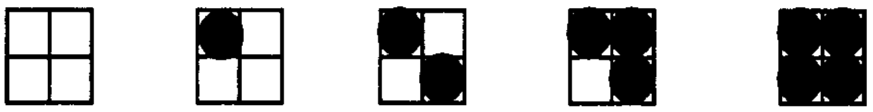

Figure 4.4: *Five intensity levels approximated with four* 2×2 *dither patterns.*

devices which are not able to display individual dots (e.g., laser printers). This means that these devices are poor at reproducing isolated 'on' pixels (the black dots in Figure 4.4). All pixels that are 'on' for a particular intensity must be adjacent to other 'on' pixels.

A CRT display is able to display individual dots; hence, the clustering requirement can be relaxed and a *dispersed-dot ordered dither* can be used. Monochrome dithering techniques can also be used to extend the number of available colors, at the expense of resolution. Consider a color display with three bits per pixel, one for red, green and blue. We can use a 2×2 pattern area to obtain 125 colors as follows: each pattern can display five intensities for each color, by using the halftone patterns in Figure 4.4, resulting in $5 \times 5 \times 5 = 125$ color combinations.

4.2.2 Image Analysis

Image analysis is concerned with techniques for extracting descriptions from images that are necessary for higher-level scene analysis methods. By itself, knowledge of the position and value of any particular pixel almost conveys no information related to the recognition of an object, the description of an object's shape, its position or orientation, the measurement of any distance on the object or whether the object is defective. Hence, image analysis techniques include computation of perceived brightness and color, partial or complete recovery of three-dimensional data in the scene, location of discontinuities corresponding to objects in the scene and characterization of the properties of uniform regions in the image.

Image analysis is important in many arenas: aerial surveillance photographs, slow-scan television images of the moon or of planets gathered from space probes, television images taken from an industrial robot's visual sensor, X-ray images and computerized axial tomography (CAT) scans. Subareas of image processing include *image enhancement, pattern detection and recognition* and *scene analysis and computer vision.*

Image enhancement deals with improving image quality by eliminating noise (extraneous or missing pixels) or by enhancing contrast.

Pattern detection and recognition deal with detecting and clarifying standard patterns and finding distortions from these patterns. A particularly important example is Optical Character R ecognition (OCR) technology, which allows for the economical bulk input of pages of typeset, typewritten or even hand-printed characters. The degree of accuracy of *handwriting recognition* depends on the input device. One possibility is that the user prints characters with a continuous-positioning device, usually a tablet stylus (a pen-based environment), and the computer recognizes them (online recognition). This is easier than recognizing scanned-in characters because the tablet records the sequence, direction and sometimes speed and pressure of strokes, and a pattern-recognition algorithm can match these factors to stored templates for each character. The recognizer may evaluate patterns without considering how the pattern has been created (a *static character recognition*) or it may focus on strokes, edges in strokes or drawing speed (a *dynamic recognizer*). A recognizer can be trained to identify different styles of block printing. The parameters of each character are calculated from samples drawn by the users. An architecture for an object-oriented character recognition engine (AQUIRE), which supports online-recognition with combined static and dynamic capabilities, is described in [KW93b]. A commercial character recognizer is described in [WB85, BW86].

Scene analysis and computer vision deal with recognizing and reconstructing 3D models of a scene from several 2D images. An example is an industrial robot sensing the relative sizes, shapes, positions and colors of objects.

Image Recognition

To fully recognize an object in an image means knowing that there is an agreement between the sensory projection and the observed image. How the object appears in the image has to do with the *spatial configuration* of the pixel values. Agreement between the observed spatial configuration and the expected sensory projection requires the following capabilities:

- Infer explicitly or implicitly an object's position and orientation from the spatial configuration.
- Confirm that the inference is correct.

To infer an object's (e.g., a cup) position, orientation and category or class from the spatial configuration of gray levels requires the capability to infer which pixels are part of the object. Further, from among those pixels that are part of the object, it requires the capability to distinguish observed object features, such as special markings, lines, curves, surfaces or boundaries (e.g., edges of the cup). These features themselves are organized in a spatial relationship on the image and the object.

Analytic inference of object shape, position and orientation depends on matching the distinguishing image features (in 2D, a point, line segment or region) with corresponding object features (in 3D, a point, line segment, arc segment, or a curved or planar surface).

The kind of object, background, imaging sensor and viewpoint of the sensor all determine whether the recognition problem is easy or difficult. For example, suppose that the object is a white planar square on a uniform black background, as shown in the digital image (Table 4.1). A simple corner feature extractor could identify the distinguishing corner points, as shown in the symbolic image (Table 4.2). The match between the image corner features and the object corner features is direct. Just relate the corners of the image square to the corners of the object square in clockwise order, starting from any arbitrary correspondence. Then, use the corresponding points to establish the sensor orientation relative to the plane of the square. If we know the size of the square, we can completely and analytically determine the position and orientation of the square relative to the position and orientation of the camera. In

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	255	255	255	255	255	0	0	0	0
0	0	0	0	255	255	255	255	255	0	0	0	0
0	0	0	0	255	255	255	255	255	0	0	0	0
0	0	0	0	255	255	255	255	255	0	0	0	0
0	0	0	0	255	255	255	255	255	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

Table 4.1: *Numeric digital intensity image of a white square (gray tone 255) on a black (gray tone 0) background and symbolic image.*

N	N	N	N	N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N	N	N	N	N
N	N	N	N	C	N	N	N	C	N	N	N	N
N	N	N	N	N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N	N	N	N	N
N	N	N	N	C	N	N	N	C	N	N	N	N
N	N	N	N	N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N	N	N	N	N

Table 4.2: *Numeric digital intensity image of image's corners shown in Table 4.1 (N = noncorner; C = corner).*

this simple instance, the unit of pixel is transformed to the unit of match between image corners and object corners. The unit of match is then transformed to the unit of object position and orientation relative to the natural coordinate system of the sensor.

On the other hand, the transformation process may be difficult. There may be a variety of complex objects that need to be recognized. For example, some objects may include parts of other objects, shadows may occur or the object reflectances may be varied, and the background may be busy.

Which kind of unit transformation must be employed depends on the specific nature of the vision task, the complexity of the image and the kind of prior information available.

Computer recognition and inspection of objects is, in general, a complex procedure, requiring a variety of steps that successively transform the iconic data into recognition information. A recognition methodology must pay substantial attention to each of the following six steps: *image formatting*, *conditioning*, *labeling*, *grouping*, *extracting* and *matching* (Figure 4.5).

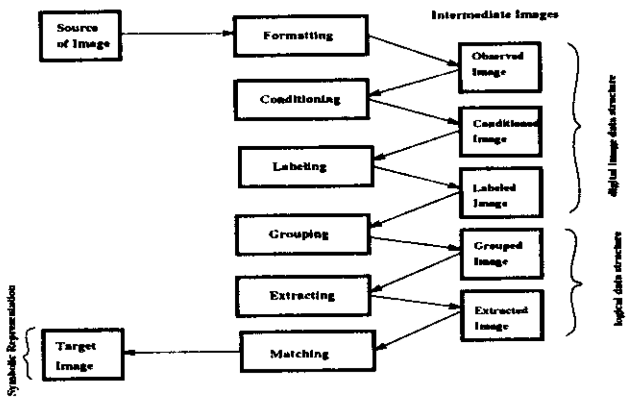

Figure 4.5: *Image recognition steps.*

Image Recognition Steps

We will give a brief overview of the recognition steps, but a deeper analysis of these steps can be found in computer vision literature, such as [Nev82, HS92], etc.

Image formatting means capturing an image from a camera and bringing it into a digital form. It means that we will have a digital representation of an image in the form of pixels. (Pixels and image formats were described in Sections 4.1.1. and 4.1.2.) An example of an observed image is shown in Figure 4.6.

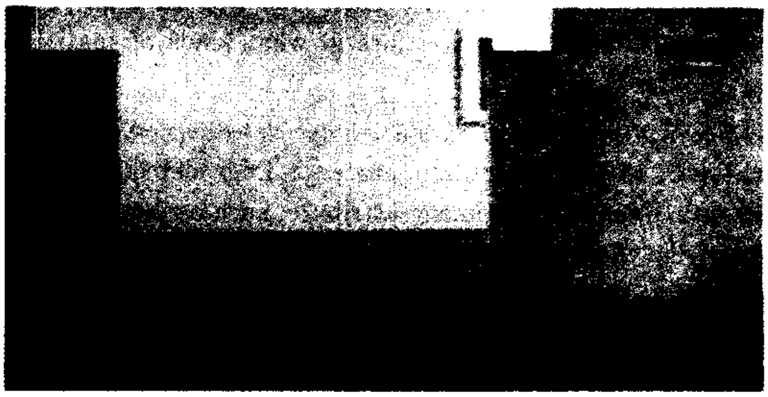

Figure 4.6: *Observed image (Courtesy of Jana Košecká. GRASP Laboratory, University of Pennsylvania, 1994).*

Conditioning, labeling, grouping, extracting and matching constitute a canonical decomposition of the image recognition problem, each step preparing and transforming the data to facilitate the next step. Depending on the application, we may have to apply this sequence of steps at more than one level of the recognition and description processes. As these steps work on any level in the unit transformation process, they prepare the data for the unit transformation, identify the next higher-level unit and interpret it. The five transformation steps, in more detail, are:

1. *Conditioning*

   Conditioning is based on a model that suggests the observed image is composed of an informative pattern modified by uninteresting variations that typically add to or multiply the informative pattern. Conditioning estimates the informative pattern on the basis of the observed image. Thus conditioning suppresses noise, which can be thought of as random unpatterned variations affecting all measurements. Conditioning can also perform background normalization by suppressing uninteresting systematic or patterned variations. Conditioning is typically applied uniformly and is context-independent.

2. *Labeling*

   Labeling is based on a model that suggests the informative pattern has structure as a spatial arrangement of events, each spatial event being a set of connected pixels. Labeling determines in what kinds of spatial events each pixel participates.

   An example of a labeling operation is *edge detection.* Edge detection is an important part of the recognition process. Edge detection techniques find local discontinuities in some image attribute, such as intensity or color (e.g., detection of cup edges). These discontinuities are of interest because they are likely to occur at the boundaries of objects. An edge is said to occur at a point in the image if some image attribute changes in value discontinuously at that point. Examples are intensity edges. An ideal edge, in one dimension, may be viewed as a step change in intensity; for example, a step between high-valued and low-valued pixels (Figure 4.7). If the step is detected, the neighboring

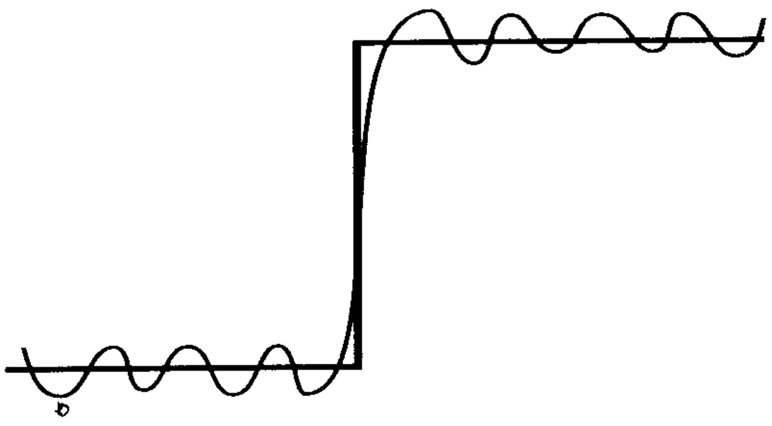

Figure 4.7: *One-dimensional edge.*

high-valued and low-valued pixels are labeled as part of an edge. An example of an image (Figure 4.6.) after edge detection is shown in Figure 4.8.

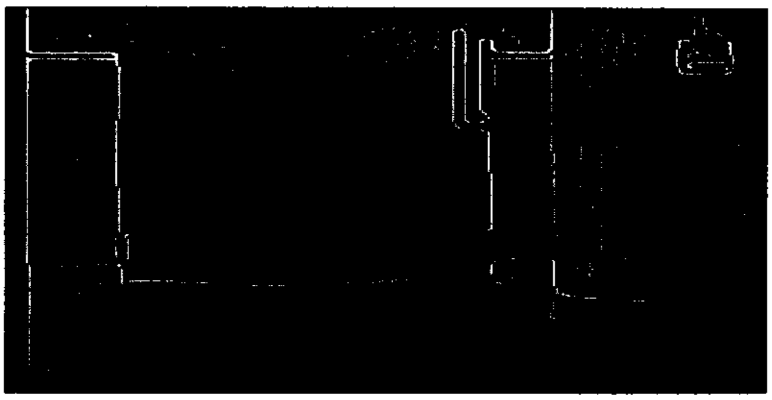

Figure 4.8: *Edge detection of the image from Figure 4.6 (Courtesy of Jana Košecká, GRASP Laboratory, University of Pennsylvania, 1994).*

Edge detection recognizes many edges, but not all of them are significant. Therefore, another labeling operation must occur after edge detection, namely *thresholding*. Thresholding specifies which edges should be accepted and which should not; the thresholding operation filters only the significant edges from the image and labels them. Other edges are removed. Thresholding the image from Figure 4.8 is presented in Figure 4.9.

Other kinds of labeling operations include corner finding and identification of pixels that participate in various shape primitives.

3. *Grouping*

   The labeling operation *labels* the kinds of primitive spatial events in which the pixel participates. The grouping operation identifies the events by collecting together or identifying maximal connected sets of pixels participating in the same kind of event. When the reader recalls the intensity edge detection viewed as a step change in intensity (Figure 4.7), the edges are labeled as step edges, and the grouping operation constitutes the step edge linking.

   A grouping operation, where edges are grouped into lines, is called *line-fitting*. A grouped image with respect to lines is shown in Figure 4.10. Again the

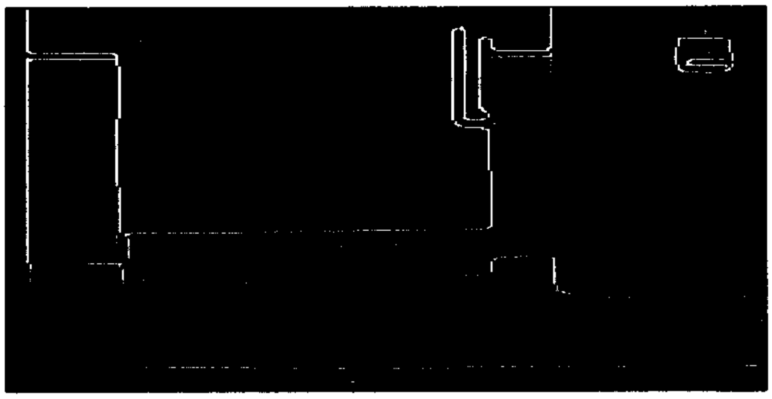

Figure 4.9: *Thresholding the image from Figure 4.8 (Courtesy of Jana Košecká, GRASP Laboratory, University of Pennsylvania, 1994).*

grouping operation *line-fitting* is performed on the image shown in Figure 4.8.

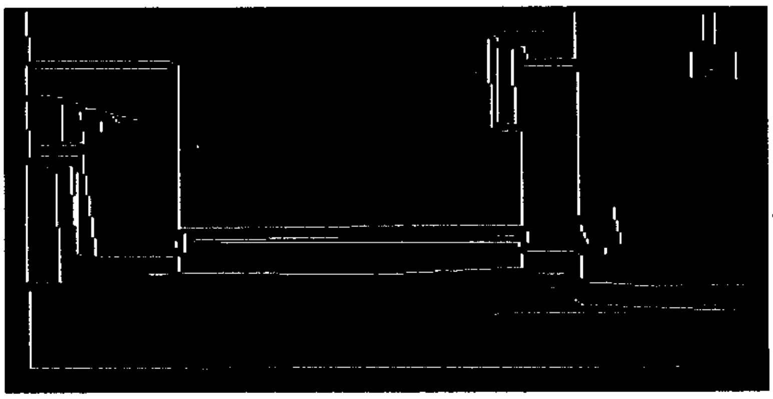

Figure 4.10: *Line-fitting of the image from Figure 4.8 (Courtesy of Jana Košecká, GRASP Laboratory, University of Pennsylvania).*

The grouping operation involves a change of logical data structure. The observed image, the conditioned image and the labeled image are all digital image data structures. Depending on the implementation, the grouping operation can produce either an image data structure in which each pixel is given an index associated with the spatial event to which it belongs or a data structure that is a collection of sets. Each set corresponds to a spatial event and contains the pairs of positions (row, column) that participate in the event. In either case, a change occurs in the logical data structure. The entities of interest prior to grouping are pixels; the entities of interest after grouping are sets of pixels.

4. *Extracting*

   The grouping operation determines the new set of entities, but they are left naked in the sense that the only thing they posses is their identity. The extracting operation computes for each group of pixels a list of properties. Example properties might include its centroid, area, orientation, spatial moments, gray tone moments, spatial-gray tone moments, circumscribing circle, inscribing circle, and so on. Other properties might depend on whether the group is considered a region or an arc. If the group is a region, the number of holes might be a useful property. If the group is an arc, average curvature might be a useful property.

   Extraction can also measure topological or spatial relationships between two or more groupings. For example, an extracting operation may make explicit that two groupings touch, or are spatially close, or that one grouping is above another.

5. *Matching*

   After the completion of the extracting operation, the events occurring on the image have been identified and measured, but the events in and of themselves have no meaning. The meaning of the observed spatial events emerges when a perceptual organization has occurred such that a specific set of spatial events in the observed spatial organization clearly constitutes an imaged instance of some previously known object, such as a chair or the letter A. Once an object or set of object parts has been recognized, measurements (such as the distance between two parts, the angle between two lines or the area of an object part) can be made and related to the allowed tolerance, as may be the case in an inspection scenario. It is the matching operation that determines the interpretation of some related set of image events, associating these events with some given three-dimensional object or two-dimensional shape.

There are a wide variety of matching operations. The classic example is *template matching*, which compares the examined pattern with stored models (templates) of known patterns and chooses the best match.

4.2.3 Image Transmission

Image transmission takes into account transmission of digital images through computer networks. There are several requirements on the networks when images are transmitted: (1) The network must accommodate bursty data transport because image transmission is bursty (The burst is caused by the large *size* of the image.); (2) Image transmission requires reliable transport; (3) Time-dependence is not a dominant characteristic of the image in contrast to audio/video transmission.

Image size depends on the image representation format used for transmission. There are several possibilities:

- *Raw image data transmission*

  In this case, the image is generated through a video digitizer and transmitted in its digital format. The size can be computed in the following manner:

  $$size = spatial\_resolution \times pixel\_quantization$$

  For example, the transmission of an image with a resolution of 640 × 480 pixels and pixel quantization of 8 bits per pixel requires transmission of 307,200 bytes through the network.

- *Compressed image data transmission*

  In this case, the image is generated through a video digitizer and compressed before transmission. Methods such as JPEG or MPEG, described in Chapter 6, are used to downsize the image. The reduction of image size depends on the compression method and compression rate.

- *Symbolic image data transmission*

  In this case, the image is represented through symbolic data representation as image primitives (e.g., 2D or 3D geometric representation), attributes and other control information. This image representation method is used in computer graphics. Image size is equal to the structure size, which carries the transmitted symbolic information of the image.

4.3 Comments

We have described in this section some characteristics of images and graphical objects. The quality of these media depends on the quality of the hardware, such as frame grabbers, displays and other input/output devices. The development of input and output devices continues at a rapid pace. A few examples should give a flavor of this development:

- *New multimedia devices*

  New *scanners* of photographical objects already provide high-quality digital images and become part of multimedia systems. An introduction of a new multimedia device (e.g., scanner) implies new multimedia format because the new medium (e.g., photographical images) can be combined with other images and other media. An example of such a new multimedia format is the Photo Image Pac File Format introduced by Kodak. This format is a new disc format that combines high-resolution images with text, graphics and sound. Hence, it enables users to design interactive Photo-CD-based presentations [Ann94b].

- *Improvements of existing multimedia devices*

  New *3D digitizers* are coming to market which enable the user to copy 3D objects of all shapes and sizes into a computer [Ann94a].