MULTIMEDIA AND ANIMATION

RAJLAXMI SAHOO

LECTURE IN COMPUTER SCIENCE AND ENGINEERING
G.I.E.T JAGATPUR , CUTTACK

MODULE-1

UNIT-1 Introduction to Multimedia

Structure

1. Objectives
2. Introduction
3. Definition of Multimedia
4. Media elements
5. Benefits of Multimedia
6. Multimedia in Learning
7. Designing multimedia applications
8. Digital imaging
9. Summary
10. Keywords
11. Exercise
12. References

1. Learning objectives

After studying this unit, you will be able to,

- define multimedia,
- mention applications of multimedia.
- describes the elements & classes of multimedia
- state benefits of multimedia.
- Explain multimedia in learning,
- design of multimedia

1. Introduction of multimedia

Multimedia systems are becoming an integral part of our heterogeneous computing and communication environment. We have seen an explosive growth of multimedia computing, communication, and applications over the last decade. The World Wide Web, conferencing, digital entertainment, and other widely used applications are using not only text and images but also video, audio, and other continuous media. In the future, all computers and networks will include multimedia devices. They will also require corresponding processing and communication support to provide appropriate services for multimedia applications in a seamless and often also ubiquitous way.

Multimedia is probably one of the most overused terms of the 90s. The field is at the crossroads of several major industries: computing, telecommunications, publishing, consumer audio-video electronics, and television/movie/broadcasting. Multimedia only brings new industrial players to the game, but adds a new dimension to the potential market.

1.2The term "Multimedia"

The word multimedia is composed of two parts: the prefix *Multi* and the root *Media*. The prefix *Multi* does not pose any difficulty; it comes from the latin word multus, which means
_numerous‖ or ―many‖.
The root *media* has a more complicated story. Media is plural form of the latin word medium. Media is noun and means ―middle, center‖.

2. The definition of multimedia

―Multimedia is the integration of multiple forms of media. It includes text, audio, animations, and video and so on.‖
Medium is ―ameans to distribute and represent information‖. Media are, for example, text, graphics, picture, voice, sound and music.

1. Classification of media

Each medium defines,

- Representation values - determine the information representation of different media

- Continuous representation values (e.g. electro-magnetic waves)
- Discrete representation values(e.g. text characters in digital form)

- Representation space determines the surrounding where the media are presented.
  - Visual representation space (e.g. paper, screen).
  - Acoustic representation space (e.g. stereo).

Following are the different classes of media,

- Perception Medium
  - How do humans perceive information in a computer?
    - Through seeing - text, images, video.
    - Through hearing - music, noise, speech.
- Representation Medium
  - How is the computer information encoded?
    - Using formats for representing and information.
    - ASCII(text), JPEG(image), MPEG(video).
- Presentation Medium
  - Through which medium is information delivered by the computer or introduced into the computer?
    - Via I/O tools and devices.
    - Paper, screen, speakers (output media).
    - Keyboard, mouse, camera, microphone (input media).
- Storage Medium
- Where will the information be stored?
  - Storage media - floppy disk, hard disk, tape, CD-ROM etc.
- Transmission Medium
- Over what medium will the information are transmitted?
  - Using information carriers that enable continuous data transmission – networks.
  - Wire, coaxial cable, fiber optics.
- Information Exchange Medium
- Which information carrier will be used for information exchange between different places?
  - Direct transmission using computer networks
  - Combined use of storage and transmission media (e.g. electronic mail).

3. Media elements

High-impact multimedia applications, such as presentations, training, and messaging, require the use of moving images such as video and image animation, as well as sound intermixed with document images and graphical text displays. Multimedia applications require dynamic handling of data consisting of a mix of text, voice, audio components, video components and image animation. Integrated multimedia applications allow the user to cut sections of all or any of these components and paste them in a new document or in another application such as an animated sequence of events, a desktop publishing system, a spreadsheet. The components of multimedia are listed as below:

- **Facsimile**
  - A facsimile is a copy or reproduction of an old book, manuscript, amp art print, that is as true to the original copy.
  - Facsimile transmission was the first practical means of transmitting document images over the telephone lines.
- **Text**
  - Text and symbols are very important for communication in any medium.
- **Document Images**
  - Document images are being used for storing business documents that must be retained for long time or may need to be accessed by a large number of people.
- **Photographic Images**
  - Photographic images are used for a wide range of applications and it can also be used as employee records for instant identification at the security level.
- **Geographical Information Systems**
  - The GIS maps are being used widely for natural resources and wild life management as well as urban planning.
- **Voice Commands and Voice Synthesis**
  - Voice commands are being used for hands-free-operation for computer programs.
  - Voice synthesis is used for presenting the results of an action to the user in a synthesized voice.
- **Audio Messages**
  - Voice messages refer to a message that could be sent to a destination using voice media.
- **Video Messages**

- ✦ Video messages refer to a message that could be sent to a destination using video transmission media.

- **Full motion stored and Live Video (FMV)**
  - ✦ Full motion video started out as a very useful idea for online training and maintenance manual.
  - ✦ The evolutionary step of FMV is video conferencing.
- **Holographic images**
  - ✦ Holographic images extend the concept of virtual reality by allowing the user to get ―inside‖ a part such as operations from the outside.
- **Fractals**
  - ✦ This technology is based on synthesizing and storing algorithms that describe the information.

4. Benefits of multimedia

Multimedia is widely used in applications like,

- **Teleconferencing**
  - VoIP(Voice over IP)
  - PC –to- PC.
  - PC-to-Telephone.
- **Audio, Video and Multimedia messages**
  - Voice mail.
  - Multimedia mail.
- **Geographical Information System.**
- **Image processing and image recognition.**
- **Video Conferencing.**
- **Universal Applications**
  - Education.
  - Science and Technology.
  - Medicine.
- **Business**
  - Advertisements.
  - Training materials.
  - Presentations.
  - Customer support services.

- **Entertainment**
  - Interactive Games.
  - Animation.
- **Enabling Technology**
  - Accessibility to web based materials
  - Teaching-learning for disabled children and adults.
- **Fine Arts and Humanities**
  - Museumtours.
  - Artexhibitions.
  - Presentations of literature.

1.5 Multimedia in learning

In many ways, ours has been a multimedia society for decades. A variety of media - print material, film strips, and visual aids - have been used in the classroom for years. Conferences and seminars have made effective use of music, lights, slide projectors and videotapes. And ubiquitous televisions have shaped a new multimedia generation.

What differentiates multimedia as the buzzword of the nineties, however, is the partnership potential of multiple media and computer technologies. Computers can now present data, text, sound, graphics, and limited motion video on the desktop. Computer based multimedia skill and knowledge applications offer benefits and value difficult to equal in non-technology implementations.

The use of multimedia in learning offers many advantages:

1.Enhancement of Text Only Messages: Multimedia enhances text only presentations by adding interesting sounds and compelling visuals.

2.Improves over Traditional Audio-Video Presentations: Audiences are more attentive to multimedia messages than traditional presentations done with slides or overhead transparencies.

3.Gains and Holds Attention: People are more interested in multimedia messages which combine the elements of text, audio, graphics and video. Communication research has shown that the combination of communication mode (aural and visual) offers greater understanding and retention of information.

4.Good for "computer-phobic": Those who are intimidated by computer keyboards and complex instructions are more comfortable with pressing buttons with a mouse or on a screen.

5. Multimedia is Entertaining as well as Educational:

Relative industries

Creative industries use multimedia for a variety of purposes ranging from fine arts, to entertainment, to commercial art, to journalism, to media and software services provided for any of the industries listed below. An individual multimedia designer may cover the spectrum throughout their career. Request for their skills range from technical, to analytical, to creative.

Commercial uses

Much of the electronic old and new media used by commercial artists is multimedia. Exciting presentations are used to grab and keep attention in advertising. Business to business, and interoffice communications are often developed by creative services firms for advanced multimedia presentations beyond simple slide shows to sell ideas or liven-up training. Commercial multimedia developers may be hired to design for governmental services and nonprofit services applications as well.

Entertainment and fine arts

In addition, multimedia is heavily used in the entertainment industry, especially to develop special effects in movies and animations (VFX, 3D animation, etc.). Multimedia games are a popular pastime and are software programs available either as CD-ROMs or online. Some video games also use multimedia features. Multimedia applications that allow users to actively participate instead of just sitting by as passive recipients of information are called *Interactive Multimedia*. In the field of Arts there are multimedia artists, whose minds are able to blend techniques using different media that in some way incorporates interaction with the viewer.

In Education, multimedia is used to produce computer-based training courses (popularly called CBTs) and reference books like encyclopedia and almanacs. A CBT lets the user go through a series of presentations, text about a particular topic, and associated illustrations in various information formats. Edutainment is the combination of education with entertainment, especially multimedia entertainment.

Learning theory in the past decade has expanded dramatically because of the introduction of multimedia. Several lines of research have evolved (e.g. Cognitive load, Multimedia learning, and the list goes on). The possibilities for learning and instruction are nearly endless.

The idea of media convergence is also becoming a major factor in education, particularly higher education. Defined as separate technologies such as voice (and telephony features), data and video that now share resources and interact with each other, synergistically creating new efficiencies, media convergence is rapidly changing the curriculum in universities all over the world. Likewise, it is changing the availability, or lack thereof, of jobs requiring this savvy technological skill.

Journalism

Newspaper companies all over are also trying to embrace the new phenomenon by implementing its practices in their work. While some have been slow to come around, other major newspapers like *The New York Times*, *USA Today* and *The Washington Post* are setting the precedent for the positioning of the newspaper industry in a globalized world.

News reporting is not limited to traditional media outlets. Freelance journalists can make use of different new media to produce multimedia pieces for their news stories. It engages global audiences and tells stories with technology, which develops new communication techniques for both media producers and consumers.

Engineering

Software engineers may use multimedia in Computer Simulations for anything from entertainment to training such as military or industrial training. Multimedia for software interfaces are often done as collaboration between creative professionals and software engineers.

Industry

In the Industrial sector, multimedia is used as a way to help present information to shareholders, superiors and coworkers. Multimedia is also helpful for providing employee training, advertising and selling products all over the world via virtually unlimited web-based technology

Mathematical and scientific research

In mathematical and scientific research, multimedia is mainly used for modeling and simulation. For example, a scientist can look at a molecular model of a particular substance and manipulate it to arrive at a new substance. Representative research can be found in journals such as the Journal of Multimedia.

Medicine

In Medicine, doctors can get trained by looking at a virtual surgery or they can simulate how the human body is affected by diseases spread by viruses and bacteria and then develop techniques to prevent it.

Document imaging

Document imaging is a technique that takes hard copy of an image/document and converts it into a digital format (e.g. Scanners).

1.6 Designing multimedia applications

We introduced a number of data element types in the earlier section that form the basic elements of multimedia applications. While the progression of graphical user interfaces opened the way for a variety of multimedia applications such as, Document imaging, Image processing, Image recognition are intended for recognizing objects by analyzing their raster images.

The rapid evolution and spread of GUIs has made it possible to implement multimedia applications widely accessible to desktop users in an office environment. In the subsequent sections, we will look at these applications and then present a view of generic multimedia applications.

1. Document Imaging

The first major step toward multimedia was originated in document image management systems. Organizations such as insurance agencies, law offices, country and state governments, including the department of defense, manage large volume documents. In fact, the Department of Defense (DOD) is among the early adopters of document image technology for applications ranging from military personal records to maintenance manuals and high speed printing systems.

The source of interest in imaging is due to its workflow management and contribution to productivity. Document imaging makes it possible to store, retrieve, and manipulate very large volume of drawings, documents, and other graphical representation of data. Imaging also provides important benefits in terms of electronic data interchange, such as in the case of sending large volume of engineering data about complex systems in electronic form rather than on paper.

Imaging is already being used for a variety of applications. An application such as medical claims processing not only speeds payment to healthcare facilities, but cuts costs of reentering

information from claim forms into a computer database. OCR systems now automatically handle the task of data entry of key fields.

2. Image Processing and Image Recognition

Unlike document image management, image processing involves image recognition, image enhancement, image synthesis, image reconstruction and image understanding. The original is not altered in document image workflow management system rather, annotations are recorded and stored separately an image processing system, on the other hand, may actually alter the contents of the image itself. Examples of image processing systems applications include recognition of images, as in factory floor quality assurance systems; image enhancement, as in satellite reconnaissance systems; image synthesis, as in law enforcement suspect identification systems; and image reconstruction, as in plastic surgery design systems.

Let us briefly review the various aspects of image processing and image recognition.

Image enhancement: Most image display systems provide some level of image enhancement. This may be a simple scanner sensitivity adjustment very much akin to the light-dark adjustment in a copier. Increasing the sensitivity and contrast makes the picture darker by making borderline pixels black or increasing the gray-level of pixels. Or it may be more complex, with capabilities built in the compression boards. These capabilities might include the following:

(i) **Image calibration-** the overall image density is calibrated, and the image pixels are adjusted to a predefined level.

(ii) **Real-time alignment-** the image is aligned in real-time for skewing caused by improper feeding of paper.

(iii) **Gray-scale normalization-** the overall level of an image is evaluated to determine if it is skewed in one direction and if it needs correction.

(iv) **RGB hue intensity adjustment-** too much color makes picture garish and fuzzy.

Automatic hue intensity adjustment brings the hue intensity within predefined ranges.

(v) **Color separation-**A picture with very little color contrast can be dull and may not bring out the details. The hardware used can detect and adjust the range of color separation.

Image Animation: Computer-created or scanned images can be displayed sequentially at controlled display speeds provide image animation that simulates real processes. Image animation is a technology that was developed by Walt Disney and brought into every home in the

form of cartoons. The basic concept of displaying the successive images at short intervals to give the perception of motion is being used successfully in designing moving parts such as automobile engines.

Image Annotation: Image Annotation can be performed in one of two ways: as a text file stored along with the image or as a small image stored with the original image.

Optical Character Recognition: Data entry has traditionally been more expensive component of data processing. OCR technology, used for data entry by scanning typed or printed words in a form, has been in use for quite some time.

3. Full motion Digital video Applications:

Full motion video has applications in the games industry and training, as well as the business world. For all business applications, some core requirements are as follows:

- Full-motion video clips should be sharable but should have only one sharable copy-user may have their own copies of design manual but storing duplicated video clips requires substantial storage.
- It should possible to attach full-motion video clips to other documents such as memos, chapter text, presentations and so on.
- Users should be able to take sections of a video clip and combine the sections with sections from other video clips to form their own video clips.
- All the normal features of a VCR metaphor, such as, rewind, forward, play, and search should be available.
- Users should be able to search to the beginning of a specific scene. That is, the fullmotion video should be indexed.
- Users should be able to place their own indexing marks to locate segments in the video clip.
- Users should be able to move and resize the window displaying the video clip.
- Users should be able to adjust the contrast and brightness of the video clip and also adjust volume of the associated sound.

4. Electronic messaging

The first-generation mail systems provided a basic text link between users and provided a valuable communications medium for users within an enterprise. These systems were the first alternative to paper-based inter office memos. The second generation of E-mail systems

expanded this capability tremendously by providing cross-platform and cross-network E-mail with a capability to attach other files ranging from editable text files to bit-mapped graphics and program executable, and embed native format graphics or text files within an electronic mail message. This increased capability has tremendous potential to change the interaction among mail-enabled workers who can exchange information much more rapidly even when they are widely distributed geographically.

The availability of other technologies, such as audio compression and decompression and fullmotion video, has opened new ways in which electronic mail can be used. What used to be a text document has given way in stages to a complex rich-text document with attachments and, more recently, to a very complex hypermedia document. With this capability in mind, electronic messaging is changing from being a communication medium to a workgroup application. A multimedia-enabled electronic messaging system requires a sophisticated infrastructure consisting of the following to support it:

- Message store and forward facility.
- Message transfer agents to route messages to their final destinations across various nodes in multilevel network.
- Message repositories where users may store them just as they would store documents in a filing document.
- Repositories for dense multimedia components such as images, video frames, audio messages, and full-motion video clips.
- Ability for multiple electronic hypermedia messages to share the same multimedia components residing in various repositories on the enterprise network.
- Dynamic access and transaction managers to allow multiple users to access, edit, and print these multimedia messages.
- Local and global directories to locate users and servers across an enterprise network.
- Automatic database synchronization of dynamic electronic messaging database.
- Automatic protocol conversions and data format conversions.
- Administrative tools to manage enterprise-wide networks.

---

7. Digital imaging

Digital imaging is the creation of digital images, typically from a physical scene. The term is often assumed to imply or include the processing, compression, storage, printing, and display of such images. The most usual method is by digital photography with a digital camera.

Methods

Digital photograph may be created directly from a physical scene by a camera or similar device. Alternatively, a digital image may be obtained from another image in an analog medium, such as photographs, photographic film, or printed paper, by an image scanner or similar device. The digitalization of analog real-world data is known as digitizing, and involves sampling and quantization.

Finally, a digital image can also be computed from a geometric model or mathematical formula. In this case the name **image synthesis** is more appropriate, and it is more often known as rendering.

Previously digital imaging depended on chemical and mechanical processes, now all these processes have converted to electronic. A few things need to take place for digital imaging to occur, the light energy converts to electrical energy- think of a grid with millions of little solar cells. Each condition generates a specific electrical charge. Charges for each of these "solar cells" are transported and communicated to the firmware to be interpreted. The firmware is what understands and translates the color and other light qualities. Pixels are what is noticed next, with varying intensities they create and cause different colors, creating a picture or image. Finally the firmware records the information for future and further reproduction.

Advantages

There are several benefits of digital imaging.

- The process enables easy access of photographs and word documents. Google is at the forefront of this _revolution,‘ with its mission to digitize the world‘s books.
- Digital imaging also benefits the medical world because it allows the electronic transmission of images to third-party providers, referring dentists, consultants, and insurance carriers via a modem.
- Digital imaging is also frequently used to help document and record historical, scientific and personal life events.
- Benefits also exist regarding photographs. Digital imaging will reduce the need for physical contact with original images. Furthermore, digital imaging creates the possibility of reconstructing the visual contents of partially damaged photographs, thus eliminating the potential that the original would be modified or destroyed.
- Another advantage to digital photography is that it has been expanded to camera phones. We are able to take cameras with us wherever as well as send photos instantly to others. It is

easy for people to us as well as help in the process of self-identification for the younger generation.

8. Summary

This chapter introduced the definition of multimedia systems- the essential components of multimedia systems, benefits of multimedia systems in different areas, and the types of applications that fall into the class of multimedia. It also includes the digital imaging process. We listed text, graphics, images, fractals, audio, and video as the components that can be found in multimedia systems. Any multimedia design address how each of components will be handled. These must be addressed by applications such as, document imaging, image processing, full motion digital video applications, electronic messaging.

We also addressed about the multimedia in learning-using of multimedia systems in different areas such as, relative industries, commercial uses, entertainment and fine arts, journalism, engineering, industry, mathematical and scientific research, medicine, document imaging.

Here, have also discussed on the methods for creating digital images and the benefits of digital imaging.

9. Keywords:

Multimedia, Text, Images, Sound, video, Elements of multimedia, Design of multimedia, Digital Imaging.

10. Exercises

1. Define multimedia.
2. List the multimedia elements.
3. Explain any two elements of multimedia system.
4. Discuss the applications of multimedia system.
5. What are the advantages of multimedia system?
6. Discuss briefly about Digital Imaging.

1.11 References

1. Ralf Steinmentz , klara Naestedt: Multimedia Fundamentals: Vol 1- Media Coding and Content processing, 2nd edition, PHI, Indian Reprint 2008.
2. Prabhat K. Andleigh, kiran Thakrar,Multimedia Systems Design, PHI,2003.

3. Digital Multimedia, Nigel Chapman

UNIT-2

Structure

1. Objectives
2. Introduction
3. Types of Multimedia
4. Hypermedia and Multimedia
5. Multimedia and WWW
6. Different types of Digital media technology
7. History of color theory
8. Different types of Web graphics
9. Picture Archiving and Communication Systems
10. Summary
11. Keywords
12. Questions
13. References

1. Learning Objectives

After studying this unit you will be able to,

- Describe types of multimedia
- Explain multimedia in different forms such as hypermedia, World Wide Web.
- Describe different kind of media technologies.
- History of color theory,
- Explain Different types of web graphics, and Picture archiving and communication.

1. Introduction

Before we get into the components and types of multimedia system, let‘s take a look at where the term multimedia originates.

Multimedia once meant slide projector and a tape recorder being played simultaneously. For instance, 50 years ago, photographic images in slide form were projected on a screen or wall while audio attempted to synchronize with the sequence or played as ―background‖ music. In 1967 pop artist Andy Warhol organized ‗multimedia‘ events called the exploding plastic inevitable, where he showed films combined with live performances that were illuminated with flashing, colored lights to create a multisensory atmosphere. The technology for joining individual media did not exist at that time.

Today, the term multimedia is associated almost exclusively with the computer and components that make up a multimedia program are digital. Various media are brought together to perform in unison on the computer as a single entity, and they are programmed or scripted using authoring software or programming languages.

2. Types of multimedia

i) Hypermedia.

ii) Interactive media.

Hypermedia: Hypermedia is the use of text, data, graphics, audio and video as elements of an extended hypertext system in which all elements are linked, where the content is accessible via hyperlinks. Text, audio, graphics, and video are interconnected to each other creating a compilation of information that is generally considered as non-linear system. The modern world wide web is the best example for the hypermedia, where the content is most of the time interactive hence non-linear. Hypertext is a subset of hypermedia, and the term was first used by Ted Nelson in 1965.

Hypermedia content can be developed using specified software such as Adobe Flash,Adobe Director and Macromedia Author ware. Some business software as Adobe Acrobat and Microsoft Office Suite offers limited hypermedia features with hyperlinks embedded in the document itself**.**

Interactive multimedia:

Any computer-delivered electronic system that allows the user to control, combine, and manipulate different types of media, such as text, sound, video, computer graphics, and animation. Interactive multimedia integrate computer, memory storage, digital (binary) data, telephone, television, and other information technologies. Their most common applications include training programs, video games, electronic encyclopedias, and travel guides. Interactive multimedia shift the user‘s role from observer to participant and are considered the next generation of electronic information systems.

Among the interactive multimedia systems under commercial development by the mid-1990s were cable television services with computer interfaces that enable viewers to interact with television programs; high-speed interactive audiovisual communications systems that rely on digital data from fiber-optic lines or digitized wireless transmissions; and virtual reality systems that create small-scale artificial sensory environments.

3. Hypermedia and Multimedia

Ted Nelson coined the term ―Hypertext during 1965. Whereas we may think of a book as a linear medium, basically meant to be read from beginning to end, a hypertext system is meant to be read nonlinearly, by following links that point to other parts of the document, or indeed to other documents.

Hypermedia is not constrained to be text-based. It can include other media, such as graphics, images, and especially the continuous media-sound and video. Apparently Ted Nelson was also first to use to this term. The World Wide Web (WWW) is the best example of a hypermedia application.

As we have seen, multimedia fundamentally means that computer information can be represented through audio, graphics, images, video, and animation in additional media (text and graphics). Hypermedia can be considered as one particular multimedia application.

Examples of typical multimedia application include: digital video editing and production systems; electronic newspapers and magazines; the WWW; online reference works, such as encyclopedias; games; groupware; home shopping; interactive TV; multimedia courseware; video conferencing; video-on-demand; and interactive movies.

4. Difference between Multimedia and Hypermedia

•Multimedia is the presentation of media as text, images, graphics, video, and audio by the use of computers or the information content processing devices (ex. Smart phones).

•Hypermedia is the use of advanced form of hypertext like interconnected systems to store and present text, graphics and other media types where the content is linked to each other by hyperlinks.

•Multimedia can be in linear or non-linear content format, but the hypermedia is only in nonlinear content format.

• Hypermedia is an application of multimedia, hence a subset of multimedia.

5. Multimedia and World Wide Web

The WWW is the largest and most commonly used hypermedia application. Its popularity is due to the amount of information available from web servers, the capacity to post such information, and ease of navigating such information with a web browser. WWW technology is maintained and developed by the World Wide Web Consortium (W3C), although the Internet Engineering Task Force (IETF) standardizes the technologies. The W3C has listed the following three goals

of WWW; universal access of web resources, effectiveness of navigating available information, and responsible use of posted material.

6. The different types of Digital media technology

(i) Audio technology

(ii) Graphics and images

(iii) Computer-based animation

(iv) Video technology

1. Audio technology

Audiology is the discipline interested in manipulating acoustic signals that can be perceived by humans. Important aspects are psychoacoustics, music, the MIDI standard, and speech synthesis and analysis. Most multimedia applications use audio in the form of music and/or speech, and voice communication is of particular significance in distributed multimedia applications.

Sound

Sound is a physical phenomenon caused by vibration of a material, such as violin string or wood log. This type of vibration triggers pressure wave fluctuations in the air around the material. The pressure wave‘s propagates in the air. The pattern of this oscillation is called wave form. When hear a sound when such a wave reaches our ears. The Analog wave patterns have two attributes,

- Volume – the height of each peak in the sound wave
- Frequency – (sometimes referred to as pitch) the distance between the peaks. The greater the distance, the lower the sound.

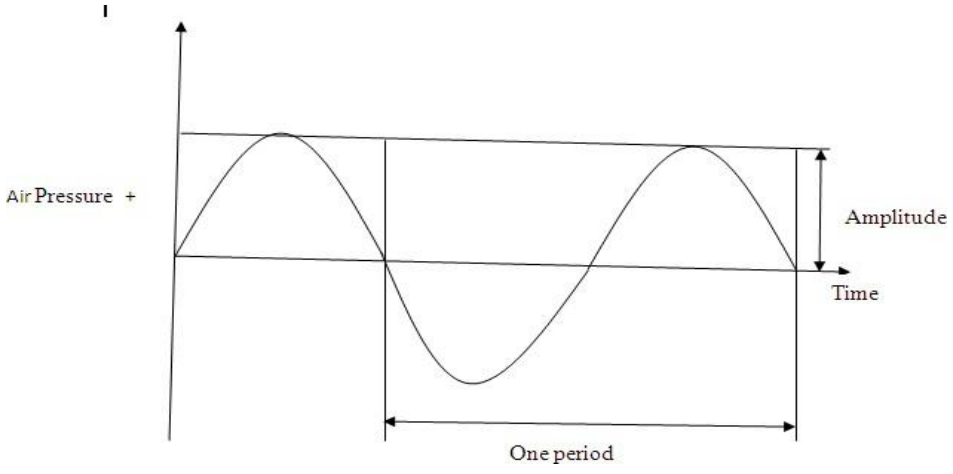


Figure 2-1 pressure wave oscillation in the air.

Frequency

A sound‘s frequency is the reciprocal value of its period. Similarly, the frequency represents the number of periods per second and is measured in hertz (Hz) or cycles per second (cps). A common abbreviation is kilohertz (kHz), which describes 1,000 oscillations per second, corresponding to 1,000Hz

Sound processes that occur in liquids, gases, and solids are classified by frequency range:

- Infrasonic: 0 to 20Hz.
- Audiosonic: 20Hz to 20kHz
- Ultrasonic: 20kHz to 1GHz
- Hypersonic: 1GHz to 10THz.

Amplitude

A sound has a property called amplitude, which humans perceive subjectively as loudness or volume. The amplitude of a sound is a measuring unit used to derive the pressure wave from its mean value.

Sound perception and psychoacoustics:

Psychoacoustics is a discipline that studies the relationship between acoustic waves at the auditory ossicle and spatial recognition of the auditor.

Two main perspectives:

(i) The physical acousticperspective.
(ii) The psychoacousticperspective.

First wave-front law: this law says that an auditor‘s judgment about the direction of the acoustic event is primarily influenced by the sound that takes the shortest and most direct way.

DigitalAudio

All multimedia file formats are capable, by definition, of storing sound information. Sound data, like graphics and video data, has its own special requirements when it is being read, written, interpreted, and compressed. Before looking at how sound is stored in a multimedia format we must look at how sound itself is stored as digital data.

All of the sounds that we hear occur in the form of analog signals. An analog audio recording system, such as a conventional tape recorder, captures the entire sound wave form and stores it in analog format on a medium such as magnetic tape.

Because computers are now digital devices it is necessary to store sound information in a digitized format that computers can readily use. A digital audio recording system does not record

the entire wave form as analog systems do (the exception being Digital Audio Tape [DAT] systems). Instead, a digital recorder captures a wave form at specific intervals, called the ***sampling rate***. Each captured wave-form snapshot is converted to a binary integer value and is then stored on magnetic tape or disk.

Storing audio as digital samples is known as ***Pulse Code Modulation (PCM)***. PCM is a simple quantizing or digitizing (audio to digital conversion) algorithm, which linearly converts all analog signals to digital samples. This process is commonly used on all audio CD-ROMs.

Differential Pulse Code Modulation (DPCM) is an audio encoding scheme that quantizes the difference between samples rather than the samples themselves. Because the differences are easily represented by values smaller than those of the samples themselves, fewer bits may be used to encode the same sound (for example, the difference between two 16-bit samples may only be four bits in size). For this reason, DPCM is also considered an audio compression scheme.

One other audio compression scheme, which uses difference quantization, ***is Adaptive Differential Pulse Code Modulation (ADPCM)***. DPCM is a non-adaptive algorithm. That is, it does not change the way it encodes data based on the content of the data. DPCM uses the sample number of bits to represent every signal level. ADPCM, however, is an adaptive algorithm and changes its encoding scheme based on the data it is encoding. ADPCM specifically adapts by using fewer bits to represent lower-level signals than it does to represent higher-level signals. Many of the most commonly used audio compression schemes are based onADPCM.

Digital audio data is simply a binary representation of a sound. This data can be written to a binary file using an audio file format for permanent storage much in the same way bitmap data is preserved in an image file format. The data can be read by a software application, can be sent as data to a hardware device, and can even be stored as a CD-ROM.

The quality of an audio sample is determined by comparing it to the original sound from which it was sampled. The more identical the sample is to the original sound, the higher the quality of the sample. This is similar to comparing an image to the original document or photograph from which it was scanned.

The quality of audio data is determined by three parameters:

- Sample resolution
- Sampling rate

- Number of audio channels sampled

The ***sample resolution*** is determined by the number of bits per sample. The larger the sampling size, the higher the quality of the sample. Just as the apparent quality (resolution) of an image is reduced by storing fewer bits of data per pixel, so is the quality of a digital audio recording reduced by storing fewer bits per sample. Typical sampling sizes are 8 bits and 16 bits.

The ***sampling rate*** is the number of times per second the analog wave form was read to collect data. The higher the sampling rate, the greater the quality of the audio. A high sampling rate collects more data per second than a lower sampling rate, therefore requiring more memory and disk space to store. Common sampling rates are 44.100 kHz (higher quality), 22.254 kHz (medium quality), and 11.025 kHz (lower quality). Sampling rates are usually measured in the signal processing terms hertz (Hz) or kilohertz (kHz), but the term samples per second (samples/second) is more appropriate for this type of measurement.

A sound source may be sampled using one channel (monaural sampling) or two channels (stereo sampling). Two-channel sampling provides greater quality than mono sampling and, as you might have guessed, produces twice as much data by doubling the number of samples captured. Sampling one channel for one second at 11,000 samples/second produces 11,000 samples.

Sampling two channels at the same rate, however, produces 22,000 samples/second.

MIDI Standard

Musical Instrument Digital Interface (MIDI) is an industry standard for representing sound in a binary format. MIDI is not an audio format, however. It does not store actual digitally sampled sounds. Instead, MIDI stores a description of sounds, in much the same way that a vector image format stores a description of an image and not image data itself. Sound in MIDI data is stored as a series of control messages. Each message describes a sound event using terms such as pitch, duration, and volume. When these control messages are sent to a MIDI-compatible device (the MIDI standard also defines the interconnecting hardware used by MIDI devices and the communications protocol used to interchange the control information) the information in the message is interpreted and reproduced by the device.

MIDI data may be compressed, just like any other binary data, and does not require special compression algorithms in the way that audio data does.

2.4.2 Graphics and images

Graphics and images are both non-textual information that can be displayed and printed. They may appear on screens as well as on printers but can‘t be displayed with devices only capable of handling characters.

Graphics are normally created in a graphics application and internally represented as an assemblage of objects such as lines, curves, or circles. Attributes such as style, width, and color define the appearance of graphics. The objects graphics are composed of can be individually deleted, added, moved and modified later. In contrast, images can be from the real world or virtual and are not editable in the sense of above.

Graphics/image data types

A wide variety of graphic/image file formats exist. Many are dependent upon particular hardware/operating system platforms, while others are cross-platform independent formats. This introduction will only touch upon some of the most common formats. While not all formats are cross-platform, there are conversion applications which will recognize and translate formats from other systems. For example, the table 2-1 shows a list of file formats used in the popular Macromedia Director.

File export		**File Import**
Image	**Palette**	**Image**
.BMP, .DIB	.PAL	.BMP
.GIF, .JPEG	.ACT	
.PNG, .PSD		
.TIFF, .WMF		

Table 2-1: Different Image/graphics file formats

There are two methods for representing and storing graphic/image data in uncompressed form,

Bit-Map or Raster Images

A bit-map representation stores the image by storing data about every dot/point of an image.

These image points are termed **pixels** (a contraction for picture element).

Vector / Structured Images

A vector graphic file format does NOT store information about every pixel of a graphic. Instead a vector graphic file format is composed of analytical geometry formula representations for basic geometric shapes, (e.g., line, rectangle, ellipse, etc.).

1-Bit Images: Images consist of pixels or picture elements in digital images. A 1-bit image consists of on and off only and thus is the simplest type of image. Each pixel is stored as a single bit. Hence, such an image is also referred to as a binary image. It is also called a 1-bit monochrome image, since it contains no color. The following figure shows a 1-bit monochrome image

Figure 2-2: Representation of 1-bit image

Figure 2-3: shows an example for1-bit Image

8-bit Gray-level image (256 gray value image)

Now consider an 8-bit image that is, one for which each pixel has a gray value between 0 and 255. Each pixel is represented by a single byte, for example, a dark pixel might have a value of 10, and a bright one might be 230. The entire image can be thought of as a 2-dimensional array of pixel values. We refer to such an array as a bitmap,- a representation of the graphics/image data that parallels the manner in which it is stored in video memory.

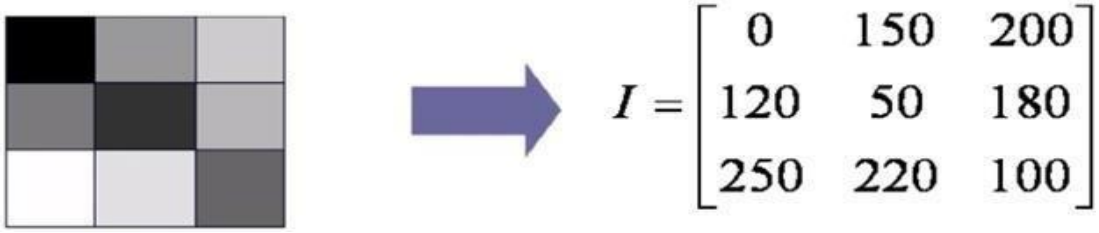


Figure 2-4: Representation of 8-bit image

Figure 2-5: Example of a 256 gray value image

Image resolution refers to the number of pixels in a digital image. Fairly high resolution for such an image might be 1,600 * 1,200, whereas the lower resolution might be 640*480. Notice that here we are using an aspect-ratio of 4:3. We don‘t have to adopt this ratio, but it has been found to look natural.

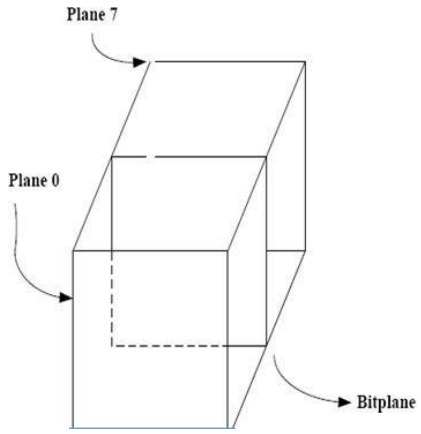


Figure 2-6: Representation of Bit-planes

Each bit-plane can have a value of 0 or 1 at each pixel, but, together, all the bit-planes make up a single byte that stores values between 0 and 255.

Color Image Data Types

- The most common data types for graphics and image file formats: 24-bit color and 8-bit color.
- Some formats are restricted to particular hardware/operating system platforms, while others are ―cross-platform‖ formats.

- Even if some formats are not cross-platform, there are conversion applications that will recognize and translate formats from one system to another.
- Most image formats incorporate some variation of a **compression** technique due to the large storage size of image files. Compression techniques can be classified into either **lossless** or **lossy**.

24-Bit color image

In a color 24-bit color image, each pixel is represented by three bytes, usually representing RGB (Red, Green, and Blue). Since each value is in the range 0-255, this format supports 256*256*256, or a total of 16,777,216, possible combined colors.

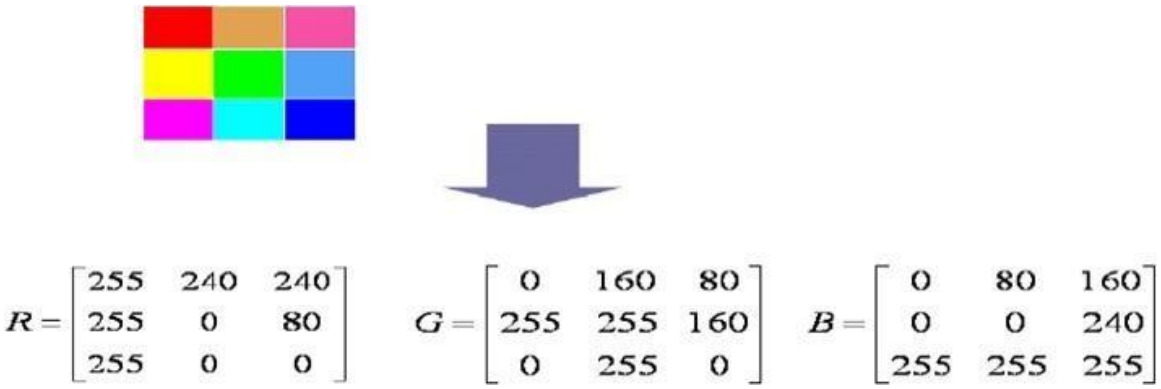


Figure 2-7: Representation of 24-bit color image

However, such flexibility does result in a storage penalty: a 640*480 24-bit color image would require 921.6 Kilobytes of storage capacity without compression.

An important point to note is that many 24-bit color images are actually stored as 32-bit images, with the extra byte of data for each pixel storing an ***alpha value*** representing special-effect information.

(a) 24 bit image"forestfire.bmp"

(b) From R channel

(c) From G channel **(d) From Bchannel**

Figure 2-8: Example of 24-bit color using with RGB color channels

8-Bit color Image

If space is a concern, reasonably accurate color image can be obtained by quantizing the color information to collapse it. Many systems can make use of only 8 bits of color information in producing screen image. Even if a system has the electronics too actually use 24-bit information, backward compatibility demands that we understand 8-bit color image files.

Such image files use the concept of a *lookup table* to store color information. Basically, the image stores not color but instead just a set of bytes, each of which index into a table with 3-byte values that specify the color for a pixel with that lookup table index.

(a) 24-bit color image **(b) 8-bit color image**

Figure 2-9: Shows samples of 24-bitand 8-bit color images

Color Lookup tables (LUTs)

The idea used in 8-bit color images is to store only the index for code value, for each pixel. Then, e.g., if a pixel stores the value 25, the meaning is to go to row 25 in a color look-up-table (LUT).

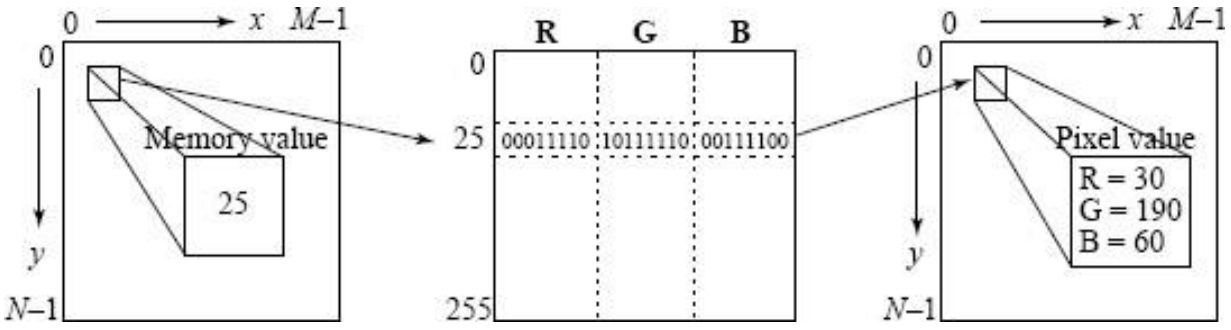


Figure 2.10: Color Lookup Table for 8-bit image

A **Color-picker** consists of an array of fairly large blocks of color (or a semi-continuous range of colors) such that a mouse-click will select the color indicated.

- In reality, a color-picker displays the palette colors associated with index values from 0 to 255.
- Each block of the color-picker corresponds to one row of the color LUT.
- If the user selects the color block with index value 2, then the colormeant is cyan, with RGB values (0*;* 255*;* 255).

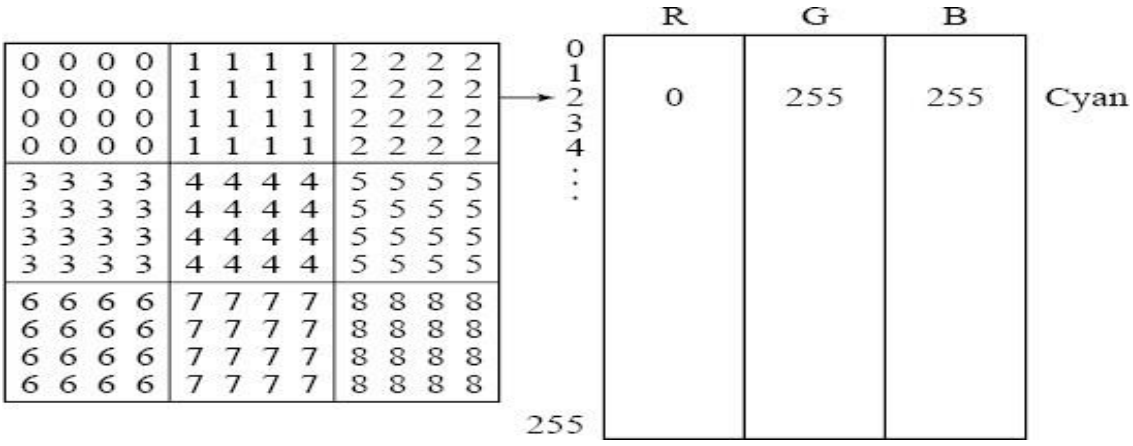


Figure 2.11: Color picker for 8-bit color. Each block of the color picker corresponds to one row color of the color LUT

3. Video technology

In addition to audio technology, TV and video technology form the basis of the processing of continuous data in multimedia systems.

Video data can be generated in two different ways:

- By recording the real world.
- Synthesis based on a description.

Basics

The human eye is the human receptor for taking in still pictures and motion pictures. Its inherent properties determine, in conjunction with neuronal processing, some of the basic requirements underlying video systems.

Representation of Video signals

In conventional black-and-white television sets, the video signal is usually generated by means of a CRT.

The representation of a video signal comprises three aspects; Visual representation, transmission, and Digitization.

(i) Visual representation

A key goal is to present the observer with as realistic as possible a representation of a scene. In order to achieve this goal, the TV picture has to accurately convey the spatial and temporal content of the scene. Important measures for this are:

- **Vertical details and viewing distance**

The geometry of a television image is based on the ratio of the picture width W to the picture height H (W/H), called the ***aspect ratio***. The Conventional aspect ratio is 4/3=1.33.The angular field of view is determined by the viewing distance, D, and is calculated as D/H.

- **Horizontal Detail and Picture Width**

The picture width used for television is 4/3 times the picture height. The horizontal field of view can be determined using the aspect ratio.

- **Total detail content of a picture**

The vertical resolution is equal to the number of picture elements of the picture height, while the number of horizontal picture elements is equal to the product of the vertical resolution and the aspect ratio.

- **Depth perception**

Depth is a result of composing a picture by each eye (from different angles). In a flat TV picture, a considerable portion of depth perception is derived from the Perspective appearance of the subject matter. Further, the choice of focal length of the camera lens and changes in depth focus influence depth perception.

- **Luminance and Chrominance**

RGB can be converted to a luminance (brightness signal) and two color difference signals (chrominance) for TV signal transmission

 TemporalAspects of Illumination

Another property of human visual perception is the limit of motion resolution. In contrast to the continuous pressure waves of an acoustic signal, a discrete sequence of still images can be perceived as a continuous sequence. The impression of motion is generated by a rapid succession of barely differing still pictures (frames). Between frames, the light is cut off briefly. Two conditions must visual reality met in order to represent a visual reality through motion picture. First, the rate of repetition of the images must be high enough to ensure continuity of movements from frame to frame. Second, rate must be high enough so that the continuity of perception is not disrupted by the dark intervals between pictures

 Continuity of Motion

It is known that Continuous motion is only perceived as such if the frame rate is higher than 15 frames per second. To make motion appear smooth in a recorded film (not synthetically generated), a rate of 30 frames per second is needed. Films recorded with 24 frames per second look strange when large objects close to the viewer move quickly.

- NTSC (National Television Systems Committee) Standard
  - Original: 30 frames/second
  - Currently: 29.97 frames/second
- PAL (Phase Alternating Line) Standard – 25 frames per second.

(ii) Signal Formats

Video signals are often transmitted to the receiver over a signal television channel. In order to encode color, consider the decomposition of a video signal into three sub-signals. For reasons of transmission, a video signal is comprised of a luminance signal and two chrominance signals. In NTSC and PAL systems, the component transfer of chrominance and luminance in a single channel is accomplished by specifying the chrominance carrier to be an odd multiple of half the line-scanning frequency.

(iii) Color Encoding

The components of a composite video signal are normally decoded into three separate signals representing the three channels of a color space model, such as RGB, YUV, or YIQ. Although the RGB model is quite commonly used in still imaging, the YUV, YIQ, or YCbCr models are more often used in motion-video imaging. TV practice uses YUV or similar color

models because the U and V channels can be down sampled to reduce data volume without materially degrading image quality.

(iv) Composite Signal

A video signal contains several different components which are mixed together in the same signal. This type of signal is called a composite video signal and is not really useful in highquality computer video. Therefore, a standard composite video signal is usually separated into its basic components before it is digitized.

(v) Computer video format

Current video digitization hardware differ in:

- Resolution of digital images (frames)
- Quantization
- Framerate

Some examples of well-known computer video formats are presented here. Each system supports various resolutions and color representations.

- Color Graphics Adapter (CGA)
  - Resolution: 320 x 200
  - 2 bits / pixel
  - The necessary storage capacity per frame is thus: 16,000 bytes.
- Enhanced Graphics Adapter (EGA)
  - Resolution: 640 x 350
  - 4 bits / pixel
  - The necessary storage capacity per frame is thus: 112,000 bytes.
- Video Graphics Array (VGA)
  - Resolution: 640 x 480
  - 8 bits / pixel
  - The necessary storage capacity per frame is thus: 307,200 bytes.
- Super Video Graphics Array (SVGA)
  - Resolution: 1024 x 768, 1280 x 1024, 1600 x 1280

- 8 bits / pixel
- The necessary storage capacity per frame is thus: 786,432 bytes.

2.4.4 Computer-basedAnimation

Somewhere between the motionless world of still images and the real-time world of video images lies the flip-book world of computer animation. All of the animated sequences seen in educational programs, motion CAD renderings, and computer games are computer-animated (computer-generated) sequences.

Traditional cartoon animation is little more than a series of artwork cells, each containing a slight positional variation of the animated subjects. When a large number of these cells are displayed in sequence and at a fast rate, the animated figures appear to the human eye to move.

A computer-animated sequence works in exactly the same manner. A series of images is created of a subject; each image contains a slightly different perspective on the animated subject. When these images are displayed (played back) in the proper sequence and at the proper speed (frame rate), the subject appears to move.

Computerized animation is actually a combination of both still and motion imaging. Each frame, or cell, of an animation is a still image that requires compression and storage. An animation file, however, must store the data for hundreds or thousands of animation frames and must also provide the information necessary to play back the frames using the proper display mode and frame rate.

Animation file formats are only capable of storing still images and not actual video information. It is possible, however, for most multimedia formats to contain animation information, because animation is actually a much easier type of data than video to store.

The image-compression schemes used in animation files are also usually much simpler than most of those used in video compression. Most animation files use a delta compression scheme, which is a form of Run-Length Encoding (RLE) that stores and compresses only the information that is different between two images (rather than compressing each image frame entirely). RLE is relatively easy to decompress on the fly Storing animations using a multimedia format also produces the benefit of adding sound to the animation. Most animation formats cannot store sound directly in their files and must rely on storing the sound in a separate disk file which is read by the application that is playing back the animation.

Animations are not only for entertaining kids and adults. Animated sequences are used by CAD programmers to rotate 3D objects so they can be observed from different perspectives; mathematical data collected by an aircraft or satellite may be rendered into an animated fly-by sequence. Movie special effects benefit greatly by computer animation.

2.7 History of Color Theory

~~Color is a subject that can make your head spin. It's such a complex entity that we take for~~ granted everyday as people and designers. The truth is there is a lot of science and color theory history behind it. This article briefly details some of the rich and interesting history behind color.

Color Theory

A major portion of art and design either relies on or utilizes color in some way and, at a first glance, color seems really easy to wield. But if you've tried serious coloring you might have realized that it difficult to get the colors to mesh or print correctly. This is because the way the eye perceives light as color and the way that substances combine to make color are different. Color theory is incredibly involved and has a lot different factors in a lot of different factors that make up color. Color theory has developed over time as different mediums such as pigments, inks, and other forms of media became more complex and easier to produce. There are currently 3 sets of primary colors depending on what materials are being used.

Color

Understanding how colors are defined in graphics data is important to understanding graphics file formats. In this section, we touch on some of the many factors governing how colors are perceived.

How We See Color?

The eye has a finite number of color receptors that, taken together, respond to the full range of light frequencies (about 380 to 770 nanometers). As a result, the eye theoretically supports only the perception of about 10,000 different colors simultaneously (although, as we have mentioned, many more colors than this can be perceived, though not resolved simultaneously).

The eye is also biased to the kind of light it detects. It's most sensitive to green light, followed by red, and then blue. It's also the case that the visual perception system can sense contrasts between adjacent colors more easily than it can sense absolute color differences, particularly if those colors are physically separated in the object being viewed. In addition, the ability to discern

colors varies from person to person; it's been estimated that one out of every twelve people has some form of color blindness.

Furthermore, the eye is limited in its ability to resolve the color of tiny objects. The size of a pixel on a typical CRT display screen, for example, is less than a third of a millimeter in diameter. When a large number of pixels are packed together, each one a different color, the eye is unable to resolve where one pixel ends and the next one begins from a normal viewing distance. The brain, however, must do something to bridge the gap between two adjacent differently colored pixels and will integrate average, ignore the blur, or otherwise adapt to the situation. For these reasons and others, the eye typically perceives many fewer colors than are physically displayed on the output device.

How ColorsAre Represented?

Several different mathematical systems exist which are used to describe colors. This section describes briefly the color systems most commonly used in the graphics file formats.

For purposes of discussion here, colors are always represented by numerical values. The most appropriate color system to use depends upon the type of data contained in the file. For example, 1-bit, gray-scale, and color data might each best be stored using a different color model.

Color systems used in graphics files are typically of the ***tri-chromatic colorimetric*** variety, otherwise known as ***primary 3-color systems***. With such systems, a color is defined by specifying an ordered set of three values. Composite colors are created by mixing varying amounts of three**,** which results in the creation of a new color. Primary colors are those which cannot be created by mixing other colors. The totalities of colors that can be created by mixing primary colors make up the color space or ***color gamut***.

Additive and subtractive color systems

Color systems can be separated into two categories: ***additive*** color systems and ***subtractive*** color systems. Colors in additive systems are created by adding colors to black to create new colors. The more color that is added, the more the resulting color tends towards white. The presence of all the primary colors in sufficient amounts creates pure white, while the absence of all the primary colors creates pure black. Additive color environments are self-luminous. Color on monitors, for instance, is additive.

Color subtraction works in the opposite way. Conceptually, primary colors are subtracted from white to create new colors. The more color that is subtracted, the more the resulting color tends towards black. Thus, the presence of all the primary colors theoretically creates pure black, while the absence of all primary colors theoretically creates pure white. Another way of looking at this process is that black is the total absorption of all light by color pigments. Subtractive environments are reflective in nature, and color is conveyed to us by reflecting light from an external source. Any color image reproduced on paper is an example of the use of a subtractive color system.

No color system is perfect. As an example, in a subtractive color system the presence of all colors creates black, but in real-life printing the inks are not perfect. Mixing all ink colors usually produces a muddy brown rather than black. The blacks we see on paper are only approximations of the mathematical ideal, and likewise for other colors.

The different types of color models are described below:

RGB (Red-Green-Blue)

RGB is perhaps the most widely used color system in image formats today. It is an additive system in which varying amounts of the colors red, green, and blue are added to black to produce new colors. Graphics files using the RGB color system represent each pixel as a color triplet-three numerical values in the form (R, G, B), each representing the amount of red, green, and blue in the pixel, respectively. For 24-bit color, the triplet (0, 0, 0) normally represents black, and the triplet (255,255,255) represents white. When the three RGB values are set to the same value-for example, (63, 63, 63) or (127,127,127), or (191,191,191) -the resulting color is a shade of gray.

CMY (Cyan-Magenta-Yellow)

CMY is a subtractive color system used by printers and photographers for the rendering of colors with ink or emulsion, normally on a white surface. It is used by most hard-copy devices that deposit color pigments on white paper, such as laser and ink-jet printers. When illuminated, each of the three colors absorbs its complementary light color. Cyan absorbs red; magenta absorbs green; and yellow absorbs blue. By increasing the amount of yellow ink, for instance, the amount of blue in the image is decreased.

As in all subtractive systems, we say that in the CMY system colors are subtracted from white light by pigments to create new colors. The new colors are the wavelengths of light reflected, rather than absorbed, by the CMY pigments. For example, when cyan and magenta are absorbed, the resulting color is yellow. The yellow pigment is said to "subtract" the cyan and magenta components from the reflected light. When all of the CMY components are subtracted, or absorbed, the resulting color is black. Almost, whether it's possible to get a perfect black is debatable. Certainly, a good black color is not obtainable without expensive inks.

In light of this, the CMY system has spawned a practical variant, CMYK, with K standing for the color black. To compensate for inexpensive and off-specification inks, the color black is tacked onto the color system and treated something like an independent primary color variable. For this reason, use of the CMYK scheme is often called 4-color printing, or process color. In many systems, a dot of composite color is actually a grouping of four dots, each one of the CMYK colors. This can be readily seen with a magnifying lens by examining a color photograph reproduced in a glossy magazine.

CMYK can be represented as either a color triple, like RGB, or as four values. If expressed as a color triple, the individual color values are just the opposite of RGB. For a 24-bit pixel value, for example, the triplet (255,255,255) is black, and the triplet (0, 0, 0) is white. In most cases, however, CMYK is expressed as a series of four values.

In many real-world color composition systems, the four CMYK color components are specified as percentages in the range of 0 to 100.

HSV (Hue, Saturation, and Value)

HSV is one of many color systems that vary the degree of properties of colors to create new colors, rather than using a mixture of the colors themselves. Hue specifies "color" in the common use of the term, such as red, orange, blue, and so on. Saturation (also called chrome) refers to the amount of white in a hue; a fully (100 percent) saturated hue contains no white and appears pure. By extension, a partly saturated hue appears lighter in color due to the admixture of white. Red hue with 50 percent saturation appears pink, for instance. Value (also called brightness) is the degree of self-luminescence of a color--that is, how much light it emits. A hue with high intensity is very bright, while a hue with low intensity isdark.

HSV (also called HSB for Hue, Saturation, and Brightness) most closely resembles the color system used by painters and other artists, who create colors by adding white, black, and gray to pure pigments to create tints, shades, and tones. A tint is a pure, fully saturated color combined with white, and a shade is a fully saturated color combined with black. A tone is a fully saturated color with both black and white (gray) added to it. If we relate HSV to this color mixing model, saturation is the amount of white, value is the amount of black, and hue is the color that the black and white are added to.

The HLS (Hue, Lightness, and Saturation) color model is closely related to HSV and behaves in the same way.

There are several other color systems that are similar to HSV in that they create color by altering hue with two other values. These include:

- HSI (Hue, Saturation, and Intensity)
- HSL (Hue, Saturation, and Luminosity)
- HBL (Hue, Brightness, and Luminosity)

None of these is widely used in graphics files.

YUV (Y-signal, U-signal, and V-signal)

The YUV model is a bit different from the other colorimetric models. It is basically a linear transformation of RGB image data and is most widely used to encode color for use in television transmission. (Note, however, that this transformation is almost always accompanied by a separate quantization operation, which introduces nonlinearities into the conversion.) Y specifies gray scale or luminance. The U and V components correspond to the chrominance (color information). Other color models based on YUV include YCbCr and YPbPr.

Black, White, and Gray

Black, white, and gray are considered neutral (achromatic) colors that have no hue or saturation. Black and white establish the extremes of the range, with black having minimum intensity, gray having intermediate intensity, and white having maximum intensity. One can say that the gamut of gray is just a specific slice of a color space, each of whose points contains an equal amount of the three primary colors, has no saturation, and varies only in intensity.

White, for convenience, is often treated in file format specifications as a primary color. Gray is usually treated the same as other composite colors. An 8-bit pixel value can represent 256

different composite colors or 256 different shades of gray. In 24-bit RGB color, (12,12,12), (128,128,128), and (199,199,199) are all shades of gray.

	RGB	CMY	HSV
Red	255,0,0	0,255,255	0,240,120
Yellow	255,255,0	0,0,255	40,240,120
Green	0,255,0	255,0,255	80,240,120
Cyan	0,255,255	255,0,0	120,240,120
Blue	0,0,255	255,255,0	160,240,120
Magenta	255,0,255	0,255,0	200,240,120
Black	0,0,0	255,255,255	160,0,0
White	255,255,255	0,0,0	160,0,240

Table 2-2: Equivalent RGB, CMY, and HSV values

2.8 Different types of Web Graphics

(i) GIF (Graphic Interchange Format)

Graphic **I**mage **F**ile format uses a CLUT (color lookup table) to define the colors as though they were individual color chips, and only supports up to 256 colors per image. Although it can simulate continuous-tone colors by dithering, that‘s generally best left to the JPEG or PNG formats. GIF87a was the original Web graphic file format. The current version, GIF89a, supports 1-bit (jagged-edge) transparency, comments, and simple animation. GIF is rarely a good choice for non-Web use. Technically GIF and its LZW compression algorithm are ―lossless,‖ but since it supports indexed color only (8-bit orless).

(ii) JPEG (Joint Photographic Experts Group)

Since June 1982, Working Group 8 (WG8) of ISO has been working on standards for the compression and decompression of still images. In June 1987, ten different techniques for coding color and gray-scaled still images were presented. An adaptive transformation coding technique based on the Discrete Cosine Transform (DCT) achieved the best subjective results.

JPEG applies to color and gray-scaled still images. Video sequences can also be handled through a fast coding and decoding of still images, a technique is called Motion JPEG.

(iii) PNG (Portable Network Graphics)

PNG, relatively recent substitute for GIFs (and some JPEGs) online. Many technical advantages, such as….

- Lossless compression (which means you could use it as an editable format, although you probably shouldn‘t in most cases).
- Multi-bit transparency map (alpha channel), even for photo-type images.
- Metadata for color management (gamma and ICC color profile), although this is something of a tease since most browsers don‘t support those things.
- Can hold either RGB data (like a JPEG) or indexed-color data (like a GIF) — but not CMYK, since that‘s designed for the Web, not for print.

(iv) PS (PostScript)

PostScript is a page-description language that some programs can generate and some printers (the expensive kind) can print from. A *.ps* is a simple text file that results when you tell a program to send its PostScript instructions to a file on your hard drive instead of to a printer; it‘s therefore called a ―printto disk file‖.

(v) EPS (Encapsulated PostScript)

EPS is essentially a PostScript file in an ―envelope.‖It usually but not always includes a rasterized preview in TIFF , plus some metadata. EPS was originally the native format of Illustrator, back in the primordial days of PostScript.

(vi) PDF (Portable Document Format)

Portable **D**ocument **F**ormat, also known as ―Adobe Acrobat format.‖ Not really a ―graphic file format,‖ since it‘s designed to contain entire pages including graphics, type, vector shapes,

and overall layout; but this is include it here because it can, in fact, be used purely as a graphic file format (to contain one or more images). For example, at one time the Macintosh used this format to store screenshot images.

(vii) PSD (Photoshop Document)

PSD stands for ―Photoshop document‖ It‘s an application-specific proprietary format, but because of Photoshop‘s dominant position in the pixel-editing world, PSD has become something of a quasi-standard. A number of other programs, even some that don‘t come from Adobe, support PSD as an additional file format — but usually as read-only or for import/export purposes, not as their true native file format.

(viii) TIFF (Tagged Image File Format)

Tagged image File Format (TIFF) is another popular image file format. Developed by the Aldus Corporation in the 1980s, it was later supported by Microsoft. Its support for attachment of additional information (referred to as ―tags‖)provides great deal flexibility. The most important tag is a format signifier: what type of compression etc. is in use in the stored image. For example, TIFF can store many different types of images: 1-bit, grayscale, 8-bit, 24-bit RGB and so on.

2.9 Picture archiving and communication system

In medical imaging, "electronic picture archiving and communication systems (PACS) have been developed in an attempt to provide economical storage, rapid retrieval of images, access to images acquired with multiple modalities, and simultaneous access at multiple sites". Electronic images and reports are transmitted digitally via PACS; this eliminates the need to manually file, retrieve, or transport film jackets. The universal format for PACS image storage and transfer is DICOM (Digital Imaging and Communications in Medicine). Non-image data, such as scanned documents, may be incorporated using consumer industry standard formats like PDF (Portable Document Format), once encapsulated in DICOM. A PACS consists of four major components: the imaging modalities such as CT and MRI, a secured network for the transmission of patient information, workstations for interpreting and reviewing images, and archives for the storage and retrieval of images and reports. Combined with available and emerging Web technology, PACS has the ability to deliver timely and efficient access to images, interpretations, and related data.

PACS breaks down the physical and time barriers associated with traditional film-based image retrieval, distribution, and display.

Types of images

Most PACSs handle images from various medical imaging instruments, including ultrasound (US), magnetic resonance (MR), positron emission tomography (PET), computed tomography (CT), endoscopy (ENDO),mammograms (MG), direct radiography (DR), computed radiography (CR)ophthalmology, etc. (see DICOM Application areas).

Uses

PACS has four main uses:

- Hard copy replacement: PACS replaces hard-copy based means of managing medical images, such as film archives. With the decreasing price of digital storage, PACSs

  provide a growing cost and space advantage over film archives in addition to the instant access to prior images at the same institution. Digital copies are referred to as Soft-copy.
- Remote access: It expands on the possibilities of conventional systems by providing capabilities of off-site viewing and reporting (distance education, telediagnosis). It enables practitioners in different physical locations to access the same information simultaneously for teleradiology.
- Electronic image integration platform: PACS provides the electronic platform for radiology images interfacing with other medical automation systems such as Hospital Information System (HIS), Electronic Medical Record (EMR), Practice Management Software, and Radiology (RIS).

- Radiology Workflow Management: PACS is used by radiology personnel to manage the workflow of patient exams.

2.10 Summary

This chapter described about the different types of multimedia, such as, hypermedia and interactive media. Hypermedia is the use of text, data, graphics, audio and video as elements of an extended hypertext system in which all elements are linked, where the content is accessible

via hyperlinks, where as the interactive multimedia is allows the user to control, combine, and manipulate different types of media, such as text, sound, video, computer graphics, and animation.

We also discussed in brief about different kind of media technology, such as Audio technology, Images and graphics, video technology and computer based animation. In audio technology, we have discussed about the representation of sound in the computer, compression and decompression of analog signals to digital signal by using PCM, DPCM, and ADPCM. In graphics and images we described a few characteristics of graphics and images. The video technology, it includes all about the representation of video signals.

In the color theory, we described the representation of colors by using RGB, CMY and HSV and so on. The different types of web graphics for using to store the images in different formats, such as GIF, JPEG, PS, PSD, TIFF and so on.

11. Keywords

Hypermedia, Interactive media, Audio, sound, frequency, Amplitude, Digital audio, Sampling. Sampling rate, PCM, DPCM, ADPCM, MIDI, 1-bit image, Gray-scaled image, Color image, CLUT, Video technology, Aspect ratio, Animation, RGB, HSV, YUV, GIF, JPEG, PNG, PS, PSD, EPS, Picture Archiving.

12. Exercises

1. Explain the different types of multimedia.
2. Differentiate between multimedia and hypermedia?
3. Discuss briefly the different types of media technologies.
4. Explain the different types of color models.
5. Explain different types of web graphics.
6. Discuss briefly about picture archiving and communication system.

2.13 References

1. Ralf Steinmentz , klara Naestedt: Multimedia Fundamentals: Vol 1- Media Coding and Content processing, 2nd edition, PHI, Indian Reprint 2008.
2. Prabhat K. Andleigh, kiran Thakrar,Multimedia Systems Design, PHI,2003.
3. Ze-Nian Li-Mark S Drew, Fundamentals of Multimedia, PHI, New Delhi .2011.
4. Donald Hearn and M. Pauline Baker, computer graphics, 3rd edition, Pearson.

UNIT-3 Introduction to Imaging Graphics, and photography

Structure

1. Objectives
2. Airborne Imaging
3. Popular file formats
4. Pixel phone
5. Pixel Art
6. Graphics chipset
7. Multimedia and Graphics
8. Advantages and Disadvantages of Graphics
9. Summary
10. Keywords
11. Questions
12. References

1. Learning objectives

- Explain Airborne Imaging methodology
- Describe popular file formats like Graphics Interchange Format, Joint Photographic Experts Group.
- Describe about Pixel and Pixel Art and graphics chipset.
- Discuss different types of graphics and the advantages and disadvantages of graphics.

2. Airborne Imaging:

Airborne imaging is one kind of digital imaging technique, which is used to digitize the images that are taken by a hyper-spectral camera. Hyper-spectral imaging, like other spectral imaging, collects and processes information from across the electromagnetic spectrum. This means that the camera is able to scan the biochemical composition of crops, and deliver an overview of every constituent present.

For example, an airborne camera capable of photographing the condition of certain crops over many acres of land could provide agriculturalists with the information they need to improve

production. This is because, instead of simply recording an image of fields from above, the camera is capable of looking directly into the crops themselves.

2. Popular File Formats

There are many different file types associated with graphics, however, only a few types are suitable for web use. The most widely supported web image formats JPEG, GIF, and PNG. The type of image dictates which image format is best to use.

1. Graphics Interchange Format (GIF)

The Graphics Interchange Format (GIF) was developed by CompuServe Information Service in 1987. Three variations of the GIF format are in use. The original specification, GIF87a, became a standard because of its many advantages over other formats. Creators of drawing programs quickly discovered how easy it was to write a program that decodes and displays GIF images. GIF images are compressed to 20 to 25 percent of their original size with no loss in image quality using a compression algorithm called LZW. The next update to the format was the GIF89a specification. GIF89a added some useful features, including transparent GIF‘s.

Unlike the original GIF specifications, which support only 256 colors, the GIF24 update supports 24-bit colors, which enable you to use more than 16 million colors. One drawback to use 24-bit color image can be displayed on an 8-bit screen, it must be *dithered,* which requires processing time and may also distort the image. GIF24 uses a compression technique called PNG.

2. JPEG (Joint Photographic Experts Group)

The most important current standard for image compression is JPEG. This standard was developed by the Joint Photographic Experts Group. As you might have guessed, it works well for natural image types (photography). Natural image types, like photography, have smooth variations of colors which mean the JPEG format also works well for images that contain gradients and varying tones and colors.

An important point about JPEG‘s is that they are generally a ***lossy*** format, unless you use another variation that is lossless. Lossy compression is simply a form of encoding that discards (loses) some of its data. There are ―lossless‖ versions of JPGs, but those variations are not common

though and don‘t have widespread support. If you need to store image data in a lossless format, it‘s best to use a format like TIFF or PSD.

Now for the technical part: The process of a JPEG divides the image into 8×8 pixel blocks and runs them through a DCT (Discrete Cosine Transform) calculation followed by quantization (JPEG 2000, a newer version, uses a wavelet transform instead of DCT).

3.3 Pixel phone

In digital imaging, a **pixel**, or a **picture element** is a physical point in a raster image, or the smallest addressable element in a display device; so it is the smallest controllable element of a picture represented on the screen. The address of a pixel corresponds to its physical coordinates.

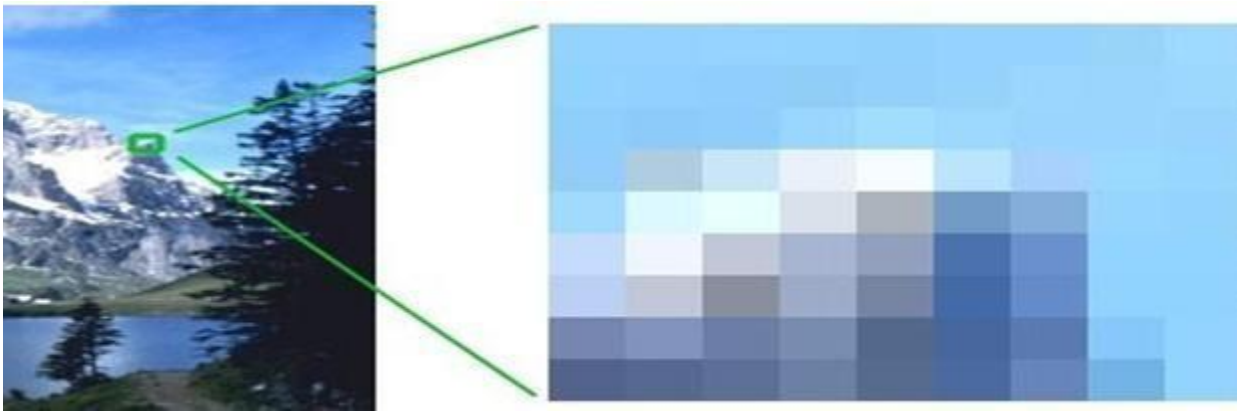


FIG 3-1 this example shows an image with a portion greatly enlarged, in which the individual pixels are rendered as small squares and can easily be seen.

Each pixel is sample of an original image; more samples typically provide more accurate representations of the original. The intensity of each pixel is variable. In color image systems, a color is typically represented by three or four component intensities such as red, green, and blue or cyan, magenta, yellow, and black as discussed in previous unit.

A pixel is generally thought of as the smallest single component of a digital image. However, the definition is highly context-sensitive. For example, there can be ―printed pixels" in a page, or pixels carried by electronic signals, or represented by digital values.

We can distinguish the pixels in two ways; i.e., ***Pixels on a display device*** and ***pixels in a digital camera.***

The pixels can be measured in printer devices in the form of dpi (dots per inch) or ppi (pixel per inch) sometimes used interchangeably, where dpi is a measure of the printer‘s density of dot placement (Example: ink droplet printer).

The more pixels used to represent an image, the closer the result can resemble the original. The number of pixels in an image is sometimes called the *resolution*, though resolution has a more specific definition. Pixel counts can be expressed as a single number, as in a "three-megapixel" digital camera, which has a nominal three million pixels, or as a pair of numbers, as in a "640 by 480 display", which has 640 pixels from side to side and 480 from top to bottom (as in a VGA display), and therefore has a total number of 640 × 480 = 307,200 pixels or 0.3 megapixels.

1. Bits per pixel

The number of distinct colors that can be represented by a pixel depends on the number of bits

---

per pixel (bpp). A 1 bpp image uses 1-bit for each pixel, so each pixel can be either on or off. Each additional bit doubles the number of colors available, so a 2 bpp image can have 4 colors, and a 3 bpp image can have 8 colors:

- 1 bpp, 2^1 = 2color(monochrome)
- 2 bpp, 2^2 = 4colors
- 3 bpp, 2^3 = 8colors
    -
    -
    -
- 8 bpp, 2^8 = 256colors
- 16 bpp, 2^{16} = 65,536 colors(High-color)
- 24 bpp, $2^{24} \approx$ 16.8 millioncolors(True-color)

3.4 Pixel Art

Pixel art is a form of digital art, created through the use of raster graphics software, where images are edited on the ***pixel level.*** Graphics in most old computer console, graphic calculator and mobile phone video games are mostly pixel art.

Image filters (such as blurring or alpha-blending) or tools with automatic anti-aliasing are considered not valid tools for pixel art, as such tools calculate new pixel values automatically, contrasting with the precise manual arrangement of pixels associated with pixel art.

3.4.1 Techniques

Drawings usually start with what is called the *Line-art,* which is the basic line that defines the character, building or anything else the artist is intending to draw. Line-arts are usually traced over scanned drawings and are often shared among other pixel artists. Other techniques, some resembling *paintings*, also exist.

The limited palette often implemented in pixel art usually promotes *dithering* to achieve different shades and colors, but due to the nature of this form of art this is done completely by hand.
Hand-made anti-aliasing is also used.

2. Saving and compression

Pixel art is preferably stored in a file format utilizing ***lossless*** data compression, such as ~~run~~length encoding . GIF and PNG are two file formats commonly used for storing pixel art. The JPEG format is avoided because its ***lossy compression*** algorithm is designed for smooth continuous-tone images and introduces visible artifacts in the presence of dithering.

3. Categories

Pixel art is commonly divided into two-categories: ***Isometric and Non-Isometric***. The ***Isometric*** kind is commonly seen in games to provide a three-dimensional view without using any real three-dimensional processing. The ***Non-isometric*** pixel art is any pixel art that does not fall in the isometric category, such as views from the top, side, front, bottom views. These are also called Plano-metric views.

3.4.4 Scaling

When pixel art is displayed at a higher resolution than the source image, it is often *scaled* using the ***nearest neighbor interpolation algorithm.*** This avoids blurring caused by other algorithms, such as ***bilinear*** and ***bi-cubic interpolation***—which interpolate between adjacent pixels and work best on continuous tones, but not sharp edges or lines. Nearest-neighbor interpolation preserves these sharp edges, but it makes diagonal lines and curves look blocky, an effect called pixilation.

5. Uses

PixelArt is widely used in different areas as follows:

- Pixel art was very often used in older computer and console video games. With the increasing use of *3D graphics* in games, pixel art lost some of its use.
- Sometimes pixel art is used for advertising too. One such company that uses pixel art to advertise is ***Bell.*** The group ***eBay*** specializes in isometric pixel graphics for advertising.
- *Icons* for operating systems with limited graphics abilities are also pixel art. The limited number of colors and resolution presents a challenge when attempting to convey complicated concepts and ideas in an efficient way. On the *Microsoft Windows* desktop icons are raster images of various sizes, the smaller of which are not necessarily scaled from the larger ones and could be considered *pixel art.*
- Modern pixel art has been seen as a reaction to the 3D graphics industry by amateur game/graphic hobbyists.

3.5 Graphics Chipset

In order to have images appear on your computer screen, you need something that can convert binary data (that's all of those 0's and 1's) into a picture. Computers either have graphics capabilities already built in to do that, or you have to install a graphics card.

For computers that do not already have graphics capabilities, the graphics card is where that translation from binary code to image takes place. A graphics card receives information sent from the processor (CPU) using software applications. The processor is the "central processing unit" (CPU or microprocessor).

A graphics card uses four main components to complete its tasks:

MOTHERBOARD: A motherboard is what allows all of the parts of a computer to receive electric power and communicate with one another. On most computers, the motherboard has sockets and slots where processors and the system's main memory are stored. It has memory chips and chipsets (which are simply groups of chips) and a clock generator which is a circuit that produces a clock signal. A motherboard has power connectors which receive and distribute electric power from the computer's power supply (usually just an electric cord). Most motherboards also have connectors for input devices for such things as a mouse or keyboard. The

graphics card uses the motherboard to receive electric power and to receive data from the computer's processor.

PROCESSOR: A processor (CPU, microprocessor) is an aptly named chip that processes data. The graphics card uses the processor to decide what to do with a pixel, like what color it should be and where it should be placed in order to make an image on the screen.

MEMORY: Whether data comes from permanent storage on a hard drive or from other input sources like keyboards, the data goes into random access memory (RAM) where the processor retrieves it. RAM is a temporary storage for data that allows the processor to access the data quickly. It would greatly slow down a computer if the processor had to access the hard drive for every piece of information it needed. The graphics card uses memory to hold information about each pixel (its color and location on the screen) and temporarily stores the final images.

MONITOR: Most of us know what the monitor is. It is the piece of the computer you are looking at right now to see this article. A graphics card uses a monitor so that you can see the final result.

A graphics card is a printed circuit board, similar to the motherboard that is another component connected to the computer's motherboard. The connection to the motherboard is how it is powered and how it communicates with the processor. Some graphics cards require more power than a motherboard can provide, so they also connect directly to a computer's power supply.

The graphics card houses its own processor and memory (RAM) and has a basic Input/output System chip (BIOS) that stores the card's settings and performs diagnostics on the memory at startup. The graphics card processor is called a graphics processing unit (GPU).

The graphics processor (GPU) creates images that it stores in the memory (RAM). The memory connects to a digital-to-analog converter (called the DAC or RAMDAC). As the name suggests, the converter takes the data from the memory and converts it into an analog signal for the monitor which is sends through a cable. The analog output from the RAMDAC goes to the monitor to display the image.

6. Multimedia and graphics

Graphics is one of the integral parts of the multimedia systems. Computer graphics remains one of the most exciting and rapidly growing areas of modern technology. Computer-graphics

methods are routinely applied in the design of most products, in training simulators, in motion pictures, in data analysis, in scientific studies, in medical procedures, and in numerous other applications. A great variety of techniques and hardware devices are now in use or under development for these diverse application areas.

1. Graphics definition

―Graphics are visual representations on some surfaces, such as, a wall, canvas, screen, paper, inform and so on‖.

The graphics can be categorized in two types: ***Raster graphics and Vector graphics***.

1. Raster graphics

Raster graphics image or Bitmap is a dot matrix data structure representing a generally rectangular grid of pixels or points of color, viewable via a monitor, paper, or other display medium. Raster images are stored in image files with varying formats.

2. Vector graphics

One way to describe an image using numbers is to declare its contents using position and size of geometric forms and shapes like lines, curves, rectangles and circles; such images are called vector images.

List of Raster and Vector graphics,

Drawings generally involve making marks on a surface by applying pressure from a tool, or moving a tool across a surface. Graphical drawing is an instrumental guided drawing.

Line Art is a rather, non-specific term, sometimes used for any image that consists of distinct straight and curved lines placed against a background, without gradations in shade (darkness) or Hue (color) to represent two-dimensional or three-dimensional objects. Line Art is usually monochromic, although lines may be of different colors.

Illustration is a visual representation such as drawings,, painting, photograph or other work of art that stresses subject more than form. The aim of illustration is to decorate a story, poem or picture of textual information.

Graph is a type of information graphic that represents tabular numeric data. Charts are often used to make it easier to understand large quantities of data and the relationship between different parts of the data.

Diagram is a simplified and structured visual representation of concepts, ideas, constructions, relations, and statistical data etc, used to visualize and clarify the topic.

7.Advantages and disadvantages of graphics

Advantages

The computer graphics are being used in many areas like,

- High quality graphics displays of personal computer provide one of the most natural means of communicating with a computer.
- It has an ability to show moving pictures, and thus it is possible to produce animations with computer graphics.
- With computer graphics use can also control the animations by adjusting the speed, the portion of the total scene in view, the geometric relationship of the objects in a scene to one another, the amount of detail shown and so on.
- The computer graphics also provides facility called update dynamics. With update dynamics it is possible to change the shape, color or other properties of the objects being viewed.
- With the recent development of digital signal processing (DSP) and audio synthesis chip the interactive graphics can now provide audio feedback along with the graphical feedbacks to make the simulated environment even more realistic.

Disadvantages

1. It is time consuming to make decisions, must be made in advance for layout, color, materials, etc.
2. Technical in nature - audience knowledge to interpret, or understand.
3. Costly - depending on the medium used (poster board, transfer letters, etc.).

1. Computer graphics applications

Graphics is widely used in many applications like,

- **User interfaces:** It is now well established fact that graphical interfaces provide an attractive and easy interaction between users and computers.
- **Plotting graphs and charts:** In industry, business, government, and educational organizations, computer graphics are commonly used to create 2D and 3D graphs of mathematical, physical, and economic functions in the form of histograms, bars, and piecharts. These charts and graphs very useful in decision making.
- **Computer-aided drafting and design:** The computer-aided drafting is used to design components and system electrical, mechanical, electro-mechanical and electronic devices such as automobile bodies, structures of building, ships very large scale integrated chips, optical systems and computer networks.
- **Simulation and Animation:** Use of graphics in simulation makes mathematical models and mechanical systems more realistic and easy to study. The interactive graphics supported by an animation software proved their in use production of animated movies and cartoon films.
- **Art and commerce:** there is a lot of development in the tools provided by computer graphics. This allows user to create artistic pictures which express message and attract attentions. Such pictures are very useful in advertisements.
- **Cartography:** Computer graphics is also used to represent geographic maps, weather maps, oceanographic maps, population density maps and so on.
- **Education training:** Computer graphics is also used to generate models of physical aids. Models of physical systems, physiological systems, population trends or equipment, such as color-coded diagram can help trainees to understand the operation of the system.

---

8. Summary

This unit described about the Airborne imaging, in which the images that are taken by hyperspectral camera are converted into digital images. This means that the camera is able to scan the biochemical composition of crops, and deliver an overview of every constituent present.

In this unit we also discussed about the popular image format such as, GIF, JPEG. The GIF images are compressed to 20 to 25 percent of their original size with no loss in image quality using a compression algorithm called LZW. An important point about JPEG‘s is that they are generally a **lossy** format, unless you use another variation that is lossless.

We also discussed about the Pixels, representation of pixels in digital images as well as the pixels on output devices. Pixel Art is a form of digital art, created through the use of raster graphics software, where images are edited on the pixel level.

This unit is also described about working of Multimedia chipsets in computer systems. Graphics are visual representations on some surfaces, such as, a wall, canvas, screen, paper, inform and so on.

9. Keywords

~~Airborne imaging, GIF, JPEG, Pixel phone, Pixel Art, Graphics chipset, Graphics, Raster images,~~ vector images.

10. Exercises

1. What is Airborne imaging? Explain briefly.
2. Discuss briefly about GIF and JPEG image formats.
3. What is pixel phone?
4. Explain briefly about Pixel Art.
5. What is graphics? What are the advantages and disadvantages of graphics?
6. Discuss the applications of computer graphics.

11. References

1. Ralf Steinmentz ,klaraNaestedt: Multimedia Fundamentals: Vol 1- Media Coding and Content–processing, 2nd edition, PHI, Indian Reprint 2008.
2. Prabhat K. Andleigh, kiranThakrar,Multimedia Systems Design, PHI,2003.
3. Ze-Nian Li-Mark S Drew, Fundamentals of Multimedia, PHI, New Delhi .2011.
4. Donald Hearn and M. Pauline Baker, computer graphics, 3rd edition, Pearson.

UNIT-4 Vector Graphics

Structure

1. Objectives
2. Vector graphics
3. Bitmap graphics
4. Different types of Digital Media Technology
5. Input devices

6. Video display devices
7. Summary
8. Keywords
9. Questions
10. References

11. Objectives

After studying this unit you will be able to

- Explain vector graphics and bitmap graphics.
- Discuss different Digital Media Technologies such as, JPEG, GIF.
- Explain various input and output devices which are required for multimedia systems.

1. Vector graphics

Vector graphics is the use of geometrical primitives such as points, lines, curves, and shapes or polygons, which are all based on mathematical expressions, to represent images in computer graphics. Vector graphics are based on vectors (also called paths, or strokes) which lead through locations called control points. Each of these points has a definite position on the x and y axes of the work plan. Each point, as well, is a variety of database, including the location of the point in the work space and the direction of the vector (which is what defines the direction of the track). Each track can be assigned a color, a shape, a thickness and also a fill.

This does not affect the size of the files in a substantial way because all information resides in the structure; it describes how to draw the vector.

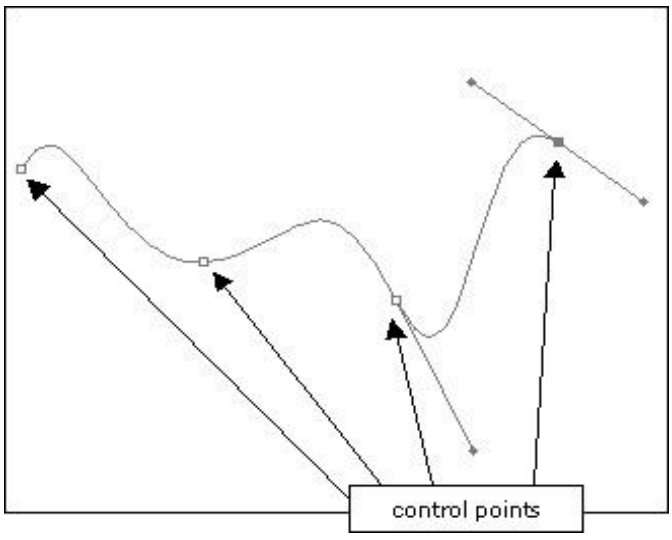


Figure 4-1: Programs like Flash draw using vectors. A very simple vector drawing might look like this.

In a vector drawing, you create control points. The lines in a vector drawing are created by the software and join up the control points that the user has drawn. There are 4 control points in the drawing above (3 are little white squares; the last one is dark to indicate that it is being worked on).

1. Advantages of vector graphics

- Pretty much resolution-independent. It is possible to rescale up a whole chunk of animation without the blockings you would get from doing this with bitmaps.
- For painting, you can specify that the bounding lines are automatically closed even when not visible, so avoiding problems of paint flooding out.
- Shapes easily edited.
- Smaller output files for Internet use.
- Shapes can be made to animate automatically from one to another, providing they have the same number of control points.

Painting and drawing programs continue to evolve; one common feature is that both type of program incorporate more and more elements of the other type; painting programs have more drawing features in them and drawing programs have more painting features. In some drawing software, it is possible to create graphics that look like typical bitmaps, (say,
with airbrush lines, for example), yet still remain vectors and so be editable. Some software can do a good job of transforming a given bitmap into a vector graphic, though there is always a loss of detail involved.

2. Vector graphics editors

A **vector graphics editor** is a computer program that allows users to compose and edit vector images interactively on a computer and save them in one of many popular vector graphics formats, such as EPS, PDF, WMF, SVG, orVML.

Features of Vector Graphics

- Some vector editors support animation, while others (e.g. ***Adobe Flash***) are specifically geared towards producing animated graphics. Generally, vector graphics are more suitable for animation, though there are raster-based animation tools as well.

- Vector editors are closely related to desktop publishing software such as ***Adobe InDesign or Scribus***, which also usually include some vector drawing tools. Modern vector editors are capable of, and often preferable for, designing unique documents of up to a few pages; it's only for longer or more standardized documents that the page layout programs are more suitable.
- Special vector editors are used for computer-assisted drafting. They are not suitable for artistic or decorative graphics, but are rich in tools and object libraries used to ensure precision and standards compliance of drawings and blueprints.
- Finally, 3D computer graphics software such as ***Maya, Blender or 3D Studio Max*** can also be thought of as an extension of the traditional 2D vector editors, as they share some common concepts and tools.

4.2 Bitmap Graphics

In Computer graphics, a **raster graphics** image, or **bitmap**, is a dot matrix data structures representing a generally rectangular grid of pixels, or points of color, viewable via a monitor, paper, or other display medium. Raster images are stored in image files with varying formats.

A bitmap corresponds bit-for-bit with an image displayed on a screen, generally in the same format used for storage in the display's video memory, or maybe as a device-independent bitmap. A bitmap is technically characterized by the width and height of the image in pixels and by the number of bits per pixel.

Raster graphics are resolution dependent. They cannot scale up to an arbitrary resolution without loss of apparent quality. This property contrasts with the capabilities of vector graphics, which easily scale up to the quality of the device rendering them. Raster graphics deal more practically than vector graphics with photographs and photo-realistic images, while vector graphics often serve better for typesetting or for graphic design. Modern computer-monitors typically display about 72 to 130 pixels per inch (PPI), and some modern consumer printers can resolve 2400 dots per inch (DPI) or more; determining the most appropriate image resolution for a given printerresolution can pose difficulties, since printed output may have a greater level of detail than a viewer can discern on a monitor.

1. Advantages of bitmaps

- In paint programs, you see what you are getting, usually in real time when wielding a ―paintbrush‖.
- When you use a scanner, the output will normally be a bitmap.
- Much easier to create the appearance of ―natural‖ media, such as areas of water colors bleeding into each other.
- More universally available interchange file formats; most bitmaps can be read by most bitmap-based software and certain file formats such as ***jpeg*** and ***png*** can be read and written by every paint program. This is not, unfortunately, the case with vector file formats where many programs can only deal with their own file formats and a very limited choice of others such as ***eps*** may be available.

1. Bitmap Editors

Raster-based image editors, such as painter, Photoshop, and MS Paint, , revolve around editing pixels , unlike vector-based image editors, such as Xfig ,CorelDraw , Adobe Illustrator, which revolve around editing lines and shapes (vectors). When an image is rendered in a raster- based image editor, the image is composed of millions of pixels. At its core, a raster image editor works by manipulating each individual pixel. Most pixel-based image editors work using the RGB color model, but some also allow the use of other color models such as the CMYK color model.

Features of Bitmap graphics

- Advanced raster editors (like Photoshop) use vector methods (mathematics) for general layout soften have special capabilities in doing so, such as brightness/contrast, and even adding "lighting" to a raster image or photograph.
- Raster editors are more suitable forretouching, photo processing, photo- realistic illustrations, collage, and hand drawn illustrations using a graphics tablet. Many contemporary illustrators use Corel Photo Paint, Photoshop,and other raster editors to make all kinds of illustrations.

3. Different types of Digital Media Technology

Uncompressed graphics data require substantial storage capacity, which is not possible in the case of uncompressed image data. Most compression methods address the same problems, one at a time or in combination. Most are already available as product. Others are currently under development or are only partially completed. While fractal image compression may be important in the future, the most important compression techniques in use today are JPEG for single image pictures.

1. JPEG (Joint Photographic Experts Group)

Since June 1982, Working Group 8(WG8) of ISO has been working on standards for the compression and decompression of still images. In June 1987, ten different techniques for coding color and gray-scaled still images were presented. An adaptive transformation coding technique based on the Discrete Cosine Transform (DCT) achieved the best subjective results. This technique was then further developed with consideration paid to the other two methods. The coding known as JPEG (Joint Photographic Experts Group) is a joint project of ISO/IECJTC/SC2/WG10 and comissionQ.16 of CCITT SGVIII. Hence, the ―J‖(from ―Joint‖) in JPEG-ISOtogether with CCITT. In 1992,JPEG became an ISO International Standard (IS).

JPEG applies to color and gray-scaled still images. Video sequences can also be handled through a fast coding and decoding of still images, a technique known as Motion JPEG. Today, implementations of parts of JPEG already available, either as software-only packages or using special-purpose hardware supports. It should be noted that as yet, most products support only the absolutely necessary algorithms. The only part of JPEG currently commercially available is the base mode with certain processing restrictions.

1. Requirements

In order to ensure the widespread distribution and application of JPEG, the following requirements were established and fulfilled:

- The standard should be the independent of image size.
- It should be applicable to any image aspect ratio and any pixel aspect ratio.
- The color space and the number of colors used should be independent of one another.
- Image content may be of any complexity, with any statistical characteristics.
- The JPEG standard should be start-of-the-art regarding the compression factor and image quality.

- The processing complexity should permit a software solution to run on a large number of available genera-purpose processors, and should be drastically reduced with the use of special-purpose hardware.
- Sequential (line by line) and progressive (successive refinement of the whole image) decoding should both be possible.

The user can select the parameters to trade off the quality of the reproduced image, the compression processing time, and the size of the compressed image.

2. JPEG overview

Applications do not have to include both an encoder and a decoder. In many applications only one of them needed. The encoded data stream has a fixed interchange format that includes encoded image data, as well as the chosen parameters and tables of the coding process, enabling decoding. If there is a common context between the coder and the decoder (e. g., if the coder and decoder are parts of the same application), then there can be abbreviated interchange format.

Figure 4-2 outlines the fundamental steps of JPEG compression. JPEG defines several image compression modes by selecting different combinations of these steps.

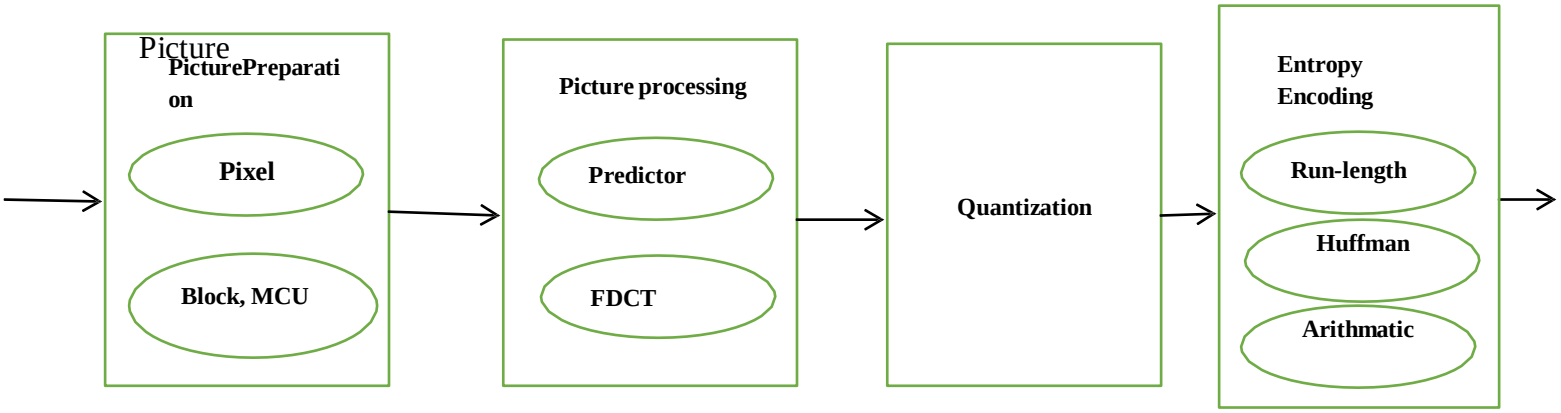


Figure 4-2 : Steps of the JPEG compression technique: Summary of the different modes.

3. JPEG Modes

JPEG defines four modes, which themselves include additional variations:

- The lossy, sequential DCT-based mode (baseline process, base mode) must be supported by every JPEG decoder.

- The expanded lossy, DCT-based mode provides a set of further enhancements to the base mode.
- The lossless mode has a low compression ratio and allows a perfect reconstruction of the original image.
- The lossless mode has a low compression ratio and allows a perfect reconstruction of the original image.
- The hierarchical mode accommodates images of different resolutions by using algorithms defined for the other three modes.

4.3.1.1 Image preparation

For image preparation, JPEG specifies a very general image model that can describe most commonly used still image representations. For instance, the model is not based on three image based on three image components with 9-bit YUV coding and a fixed numbers of lines and columns. The mapping between coded color values and the colors they represent is also not correct. This fulfills the JPEG requirement of independence from image parameters such as image size or the image and pixel aspect ratio.

An image consists of at least one and at most N=255 components or planes, as shown a left side of Figure 4-4. These planes can be assigned to individual RGB (red, green, blue) colors, or to the YIQ or YUV signals, for example

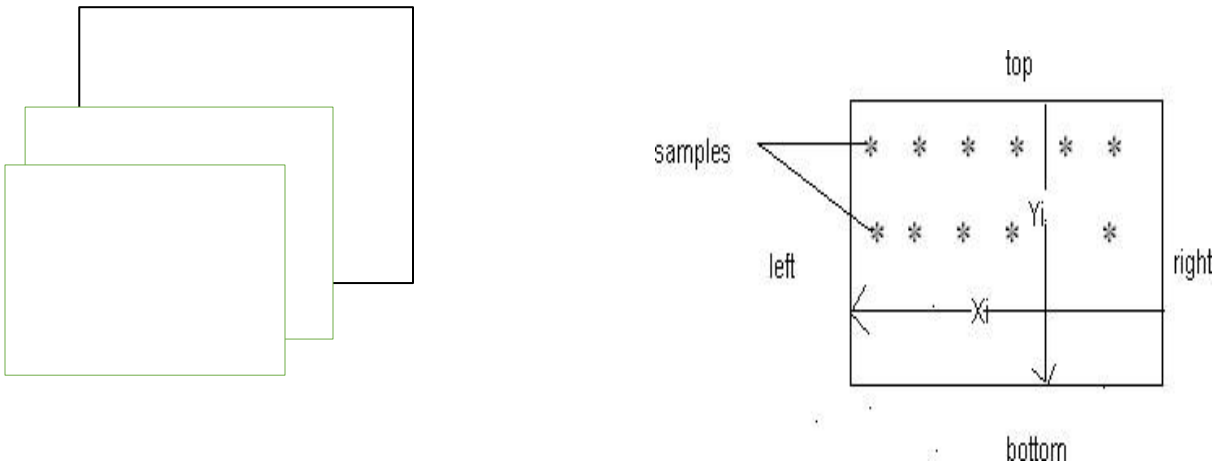


Figure 4-4 Uncompressed digital image.

Each component is a rectangular array $X_i * Y_i$ of pixels (the samples).Figure 4-5 shows an image with three planes, each with the same resolution.

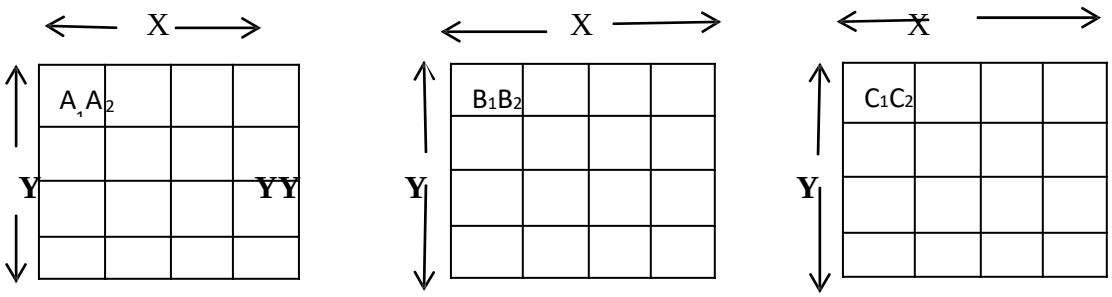


Figure 4-5 Example of JPEG image preparation with three components having the same resolution.

The resolution of the individual components may be different. Figure 4-6 shows an image with three planes where the second and third planes have half of the first plane. A gray-scale image will, in most cases, consists of a single component, while RGB color image will have three components with the same resolution (same number of lines $Y_1 = Y_2 = Y_3$ and the same number of columns $X_1 = X_2 = X_3$). In JPEG image preparation, YUV color images with subsampling of the chrominance components use three planes with $Y_1 = 4Y_2 = 4Y_3$ and $X_1 = 4X_2 = 4X_3$.

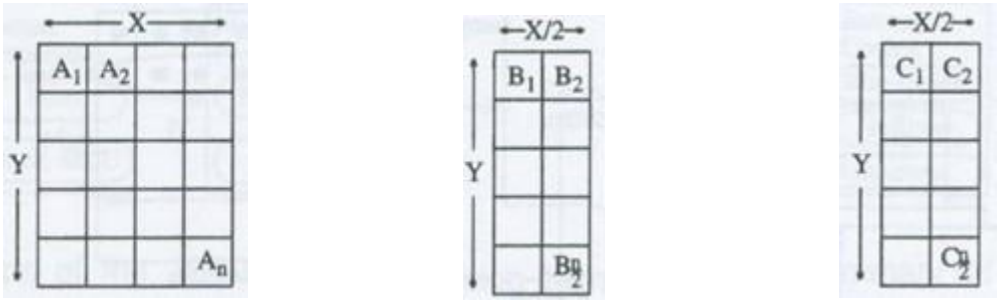


Figure 4-6 Example of JPEG image preparation with three components having different resolutions.

Each pixel is represented by *P* bits with the values in the range from 0 to $2^p - 1$. All pixels of all components of an image must have the same number of bits. The lossy modes of JPEG use a precision of either 8 or 12 bits per pixel. The lossless modes can use between 2 and 12 bits per pixel. If a JPEG application uses any other number of bits, the application itself must suitably transform the image to conform to the number of bits defined by the JPEG standard.

Instead of the values X_i and Y_i, the compressed data includes the values X (maximum of all X_i), as well as factors H_i and V_i for each plane. H_i and V_i represent the relative horizontal and vertical resolutions with respect to the minimal horizontal and vertical resolutions.

Let us consider the following example from. An image has a maximum resolution of 512 pixels in both the horizontal and vertical directions and consists of three planes. The following factors are given:

Plane 0: $H_0 = 4$, $V_0 = 1$

Plane 1: $H_1 = 2$, $V_1 = 2$

Plane 2: $H_2 = 1$, $V_2 = 1$

This leads to:

$X = 512$, $Y = 512$, $H_{max} = 4$ and $V_{max} = 2$

Plane 0: $X_0 = 512$, $Y_0 = 256$

Plane 1: $X_1 = 256$, $Y_1 = 512$

Plane 2: $X_2 = 128$, $Y_2 = 256$

H_i and V_i must be integers between 1 and 4. This awkward-looking definition is needed for the interleaving components.

In the image preparation stage of compression, the image is divided into data units. The lossless mode uses one pixel as one data unit. The lossy mode uses blocks of 8*8 pixels. This is a consequence of DCT, which always transforms connected blocks.

Up to now, the data units are usually prepared component by component and passed on in order to the following image processing. Within each component, the data units are processed from left to right, as shown in Figure 4-7. This is known as a non-interleaved data ordering.

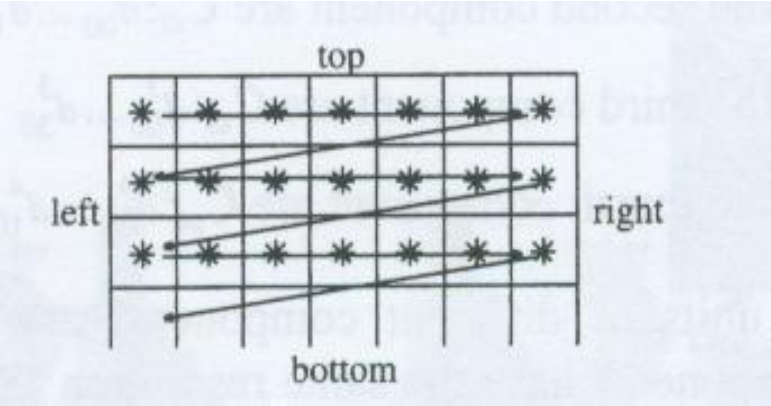


Figure 4-7 Non-interleaved processing order of data units when processing a single component.

Due to the finite processing speed of the decoder, processing of data units of different components may be interleaved. If the non-interleaved mode were used for a very highresolution.

RGB-encoded image, during rendering the display would first show only red, then green, and finally the correct colors.

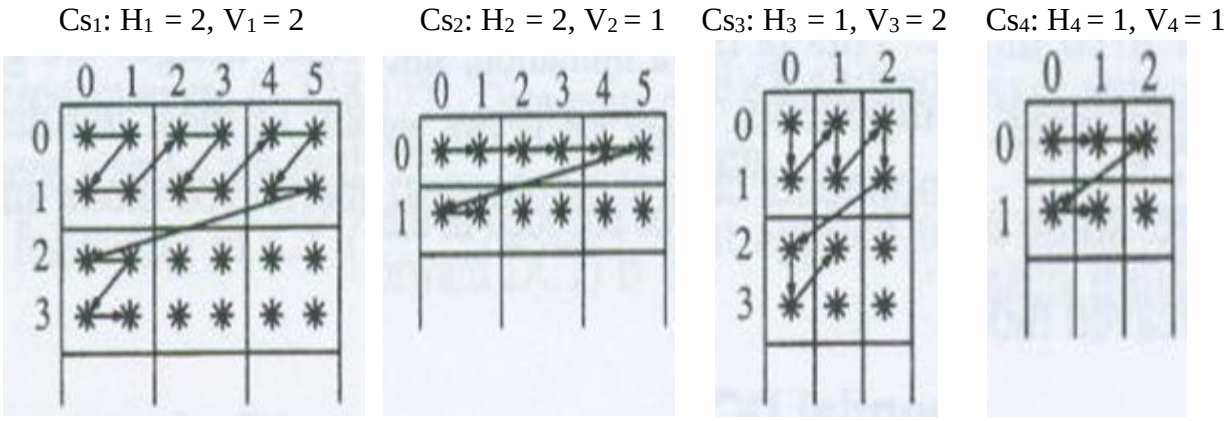


Figure 4-8 Inteleaved processing order of data units.

Figure 4-8 shows an example with four components. Above each component, the corresponding values for H and V are shown. The first component has the highest resolution in both dimensions and fourth component has the lowest resolution. The arrows within each component indicate the sampling order of individual data units.

Minimum Coded Units (MCUs) are built in the following order;

MCU1 = $d_{100}d_{101}d_1$ d_1 d_2 d_2 d_3 d_3 d_4
10 11 00 01 00 10 00

MCU2 = $d_{102}d_{103}d_1$ d_1 d_2 d_2 d_3 d_3 d_4
12 13 02 03 01 11 01

MCU3 = $d_{104}d_{105}d_1$ d_1 d_2 d_2 d_3 d_3 d_4
14 15 04 05 02 12 02

MCU3 = $d_{120}d_{121}d_1$ d_1 d_2 d_2 d_3 d_3 d_4
30 31 10 11 20 30 10

The data units of the first component are Cs_1: d^1_{00}…….d^1_{31}

The data units of the second component are Cs_2: d^2_{00}…….d^2_{11}

The data units of the third component are Cs_3: d^3_{00}…….d^3_{30}

The data units of the fourth component are Cs_1: d^4_{00} …….d^4_{10}.

Interleaved data units of different components are combined into Minimum Coded Units. If all components have the same resolution ($X_i * Y_i$), an MCU consists exactly one data unit from each component. The decoder displays the image MCU by MCU. This allows for correct color presentation, even for partly decoded images.

In the case of different component resolutions, the construction of MCUs becomes more complex (see Figure 4-8). For each component, regions of data units, potentially with different numbers of data units, are constructed. Each component consists of the same regions. For example, in Figure 4-8 each component has six regions. MCUs are comprised of exactly one region from each component. The data units within a region are ordered from left to right and from top to bottom.

According to the JPEG standard, a maximum of four components can be encoded using the interleaved mode. This is not a limitation, since color images are generally represented using three components. Each MCU can contain at most ten data units. Within an image, some components can be encoded in the interleaved mode and others in the non-interleaved mode.

4.3.1.2 Lossy Sequential DCT-Based Mode

After image preparation, the uncompressed image samples are grouped into data units of 8×8 pixels, as shown in Figure 4-9; the order of these data units is defined by the MCUs. In this baseline mode, each sample is encoded using p = 8bit. Each pixel is an integer in the range 0 to 255.

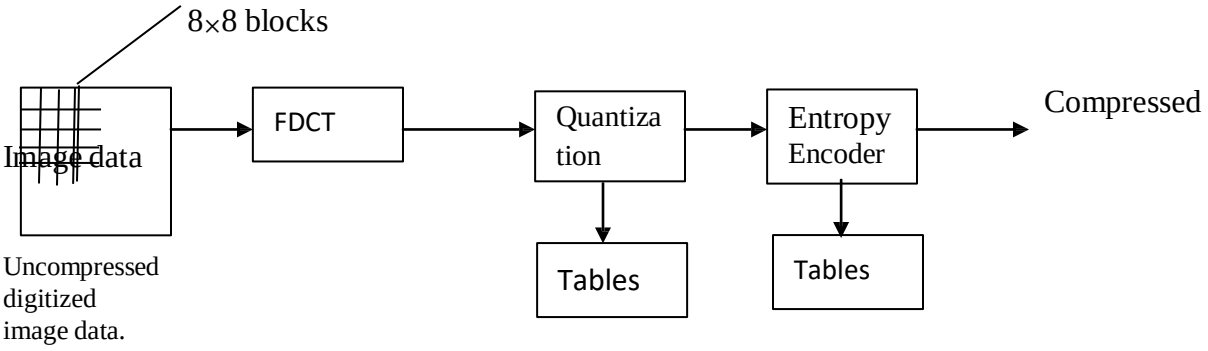


Figure 4-9 Steps of the lossy sequential DCT-based coding mode, starting with an uncompressed image after image preparation.

4.3.1.2.1 Image processing

~~The first step of image processing in the baseline mode, as shown in Figure 4-9, is a~~ transformation coding performed using the Discrete Cosine Transform (DCT).the pixel values are shifted into the zero-centered interval(-128, 127). Data units of 8×8 shifted pixel values are defined by S_{yx}, where x and y are in the range of zero to seven.

The following FDCT (Forward DCT) is then applied to each transformed pixel value:

$$Svu = \frac{1}{4} CuCv \sum_{x=0}^{7} \sum_{y=0}^{7} Syx \cos\frac{2x+1\,u\pi}{16} \cos\frac{2y+1\,v\pi}{16}$$

Where: C_u, $C_v = 1/\sqrt{2}$ for u,v = 0; otherwise C_u, $C_v = 1$

Altogether, this transformation must be carried out 64 times per data unit. The result is 64 coefficients S_{uv}. Due to the dependence of DCT on the Discrete Fourier Transform (DFT), which maps values from the time domain to the frequency domain, each coefficient can be regarded as a two-dimensional frequency.

The coefficient S_{00} corresponds to the portion where the frequency is zero in both dimension. It is also known an DC-coefficient and determines the fundamental color of all 64 pixels of the data unit. The other coefficients are called as AC-coefficients.

For later reconstruction of the image, the decoder uses the Inverse DCT (IDCT). The coefficients S_{uv}must be used for the calculation:

$$Sxy = \frac{1}{4} CuCv \sum_{u=0}^{7} \sum_{v=0}^{7} Suv \cos\frac{2x+1\,u\pi}{16} \cos\frac{2y+1\,v\pi}{16}$$

Where: C_u, $C_v = 1/\sqrt{2}$ for u,v = 0; otherwise C_u, $C_v = 1$

If the FDCT, as well as the IDCT, could be calculated with full precision, it would be possible to reproduce the original 64 pixels exactly. From a theoretical point of view, DCT would be lossless in this case. In practice, precision is limited and DCT is thus lossy. The JPEG standard does not

define any specific precision. It is thus possible that two different JPEG decoder implementations could generate different images as output of the same compressed data. JPEG merely defines the maximum allowable deviation.

Many images contain only a small portion of sharp edges; they consist mostly of areas of a single color. After applying DCT, such areas are represented by a small portion of high frequencies. Sharp edges, on the other hand, are represented as high frequencies. Images of average complexity thus consist of many AC-coefficients with a value of zero. This means that subsequent entropy encoding can be used to achieve considerable data reduction.

4.3.1.2.2 Quantization

Image processing is followed by the quantization of all DCT coefficients; this is a lossy process. For this step, the JPEG application provides a table with 64 entries. One for each of the 64 DCT coefficients. This allows each of the 64coefficients to be adjusted separately. The application can thus relative significance of the different coefficients. Specific frequencies can be given more importance than others depending on the characteristics of the image material to be compressed. The possible compression is influenced at the expense of achievable image quality.

The table entries Q_{vu} are integer values coded with 8 bits. Quantization is performed according to the formula:

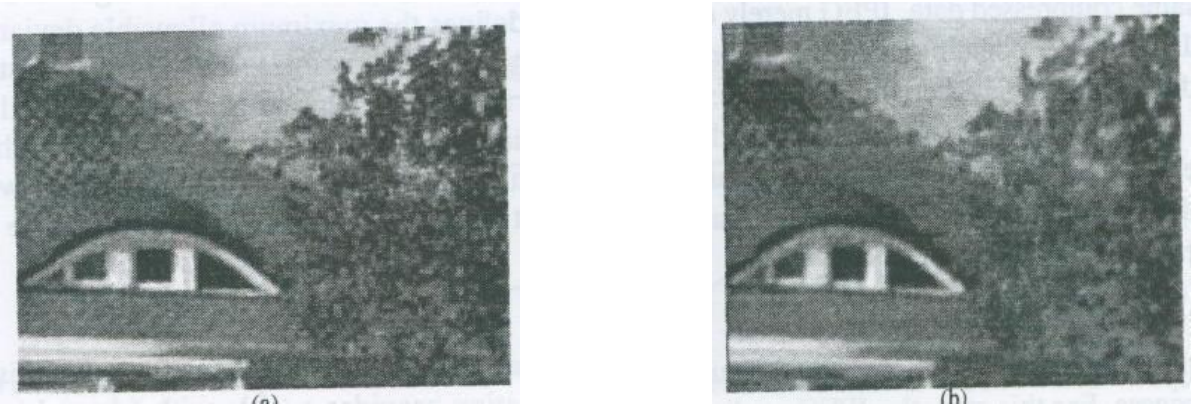
$$sq_{uv} = round\frac{S_{vu}}{Q_{vu}}$$

The greater the table entries, the coarser the quantization.Dequantization is performed prior to the IDCT according to the formula:

$$Rvu = Sq_{uv} \times Q_{uv}$$

Quantization and dequantization must use the same tables.

Figure 4-10 shows a greatly enlarged detail. The blocking and the effects of quantization are clearly visible. In Figure 4-10(b) a coarser quantization was performed to highlight the edges of the 8×8 blocks.



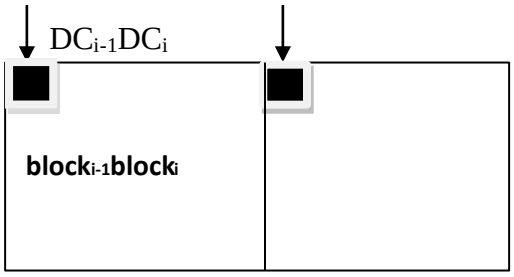
(a) (b)

Figure 4-10 Quantization effect

3. Entropy Encoding

During the next step, either the initial step of entropy encoding or preparations for the coding process, the quantized DC-coefficients are treated differently than the quantized AC-coefficients. The processing order of all coefficients is specified by the zig-zag sequence.

- The DC-coefficients determine the fundamental color of the data units. Since this changes little between neighboring data units, the differences between successive DCcoefficients are very small values. Thus each DC-coefficient of the previous data unit, in Figure 4-11, and subsequently using only the difference.



$$DIFF = DC_i - DC_{i-1}$$

Figure 4-11 Preparation of DCT DC-coefficients for entropy encoding. Calculation of the difference between neighboring values.

- The DCT processing order of the AC-coefficients using zig-zag sequence as shown in Figure 4-12 illustrates that coefficients with lower frequencies are encoded first, followed by the higher frequencies. The result is an extended sequence of similar data bytes, permitting efficient entropy encoding.

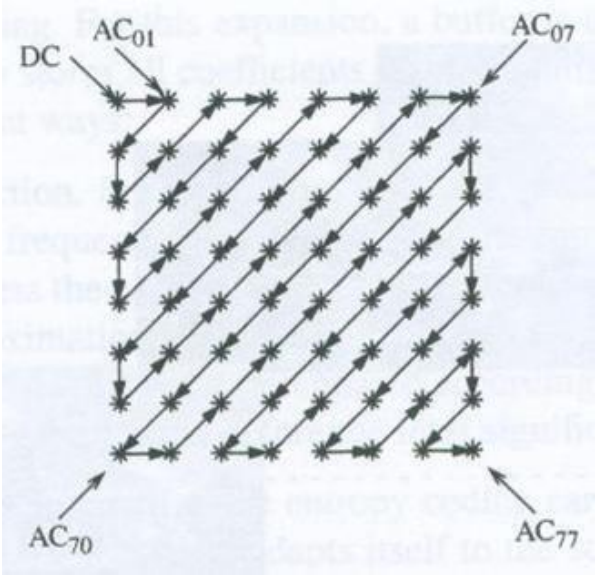


Figure 4-12 preparation of DCT AC-coefficients for entropy encoding, in order of increasing frequency.

JPEG uses Huffman coding and arithmetic coding as entropy encoding methods. For the lossy sequential DCT-based base mode, discussed in this section, only Huffmann encoding allowed. In both methods, a run-length encoding of zero values is first applied to the quantized ACcoefficients. Additionally, non-zero AC- coefficients as well as the DC-coefficients are transformed into a spectral representation to further compress data. The number of bits required depends on the value of each coefficient. Non-zero AC-coefficients are represented using between 1 and 10 bits. For the representation of DC-coefficients, a higher resolution of 1 bit to a maximum of 11 bits is used.

In sequential coding, the whole image is coded and decoded in a single run. Figure 4-13 shows an example of decoding with immediate presentation on the display. The picture is completed from top to bottom.

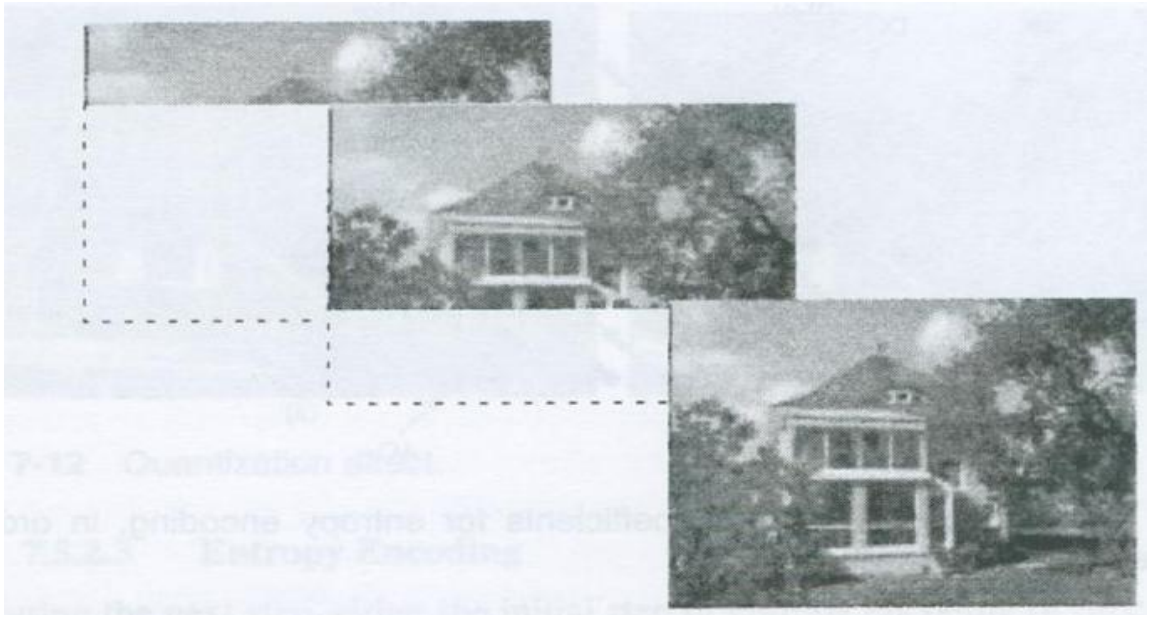


Figure 4-13 Example of sequential picture preparation, here using the lossy DCT-based mode.

4.3.1.3 Expanded Lossy DCT-Based mode

Image preparation in this mode differs from the previously described mode in terms of the number of bits per pixel. This mode supports 12 bits per sample value in addition to 8 bits. The image processing is DCT mode and is performed analogously to the baseline DCT mode. For the expanded lossy DCT-based mode, JPEG defines progressive coding in addition to sequential coding. In the first decoding run, a very rough, unsharp image appears. This is defined during successive runs. An example of a very unsharp image is shown in Figure 4-14 (a). it is substantially sharper in Figure 4-14 (b), and in its correct resolution in Figure 4-14 (c).

(a) (b) (c)

Figure 4-14Proggressive picture representation: (a) first phase, a very sharp image; (b) second phase, unsharp image; (c) third phase, sharp image.

Progressive image presentation is achieved by expanding quantization. This is equivalent to layered coding. For this expansion, a buffer is added at the output of the quantizer that temporarily stores all coefficients of the quantized DCT. Progressiveness is achieved in two different ways:

- Using spectral selection, in the first run only the quantized DCT-coefficients of each data unit‘s low frequencies are passed on to the entropy encoding. Successive runs gradually process the coefficients of higher frequencies.
- In successive approximation, all of the quantized coefficients are transferred in each run, but individual bits are differentiated to their significance. The most significant bits are encoded before the least significant bits.

Besides Huffman coding, arithmetic entropy coding can be used in the expanded mode. Arithmetic coding automatically adapts itself to the statistical characteristics of an image and thus requires no tables from the application. According several publications, arithmetic encoding achieves around five to ten percent better compression rate. Arithmetic coding is slightly more complex and its protection by patents must be considered.

4.3.1.4 Lossless Mode

The lossless mode shown in figure 4-15 uses single pixel as data units during image preparation. Between 2 and 16 bits can be used per pixel. Although all pixels of an image must use the same precision, one can also conceive of adaptive pixel precision.

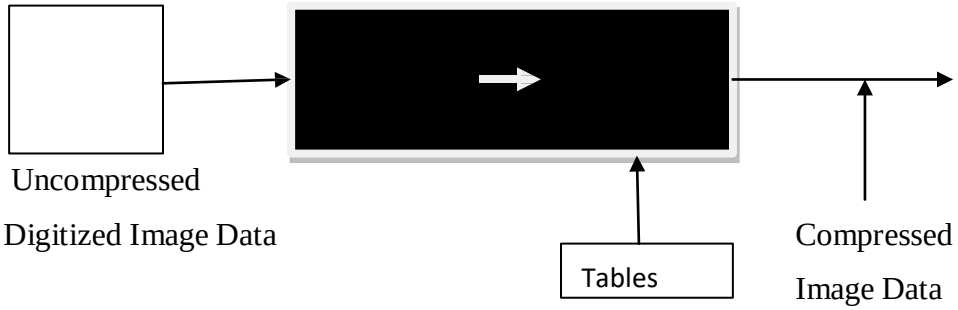


Figure 4-15 Lossless modes, based on prediction.

In this mode, image processing and quantization use a predictive technique instead of transformation coding. For each pixel X as shown in Figure 4-16, one of eight possible predictors

is selected. The selection criterion is the best possible prediction of the value X from the already known adjacent samples.A,B, and C. Table 4-1 lists the specified predictors.

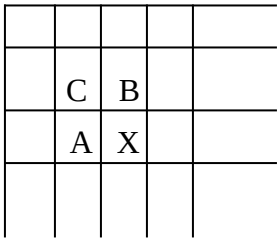


Figure 16-4Basis of prediction in lossless mode.

The number of the chosen predictor, as well as the difference between the prediction and the actual value, are passed to the subsequent entropy encoding, which can use either Huffman or Arithmetic coding.

In summary, this mode supports two types of processing, each with between 2 and 16 bits per pixel. Each variant can use either Huffman coding or Arithmetic coding.

Selection Value	**Prediction**
0	No Prediction
1	X = A
2	X =B
3	X = C
4	X = A + B + C
5	X = A + (B –C)/2
6	X = B + (A – C)/2
7	X = (A+B)/2

Table 4-1 Predictors in lossless mode.

4.3.2 Graphics Interchange Format (GIF)

GIF (Graphics Interchange Format) was developed by CompuServe and uses the LZW (LempelZiv-Welch) compression method, which is lossless. This method of compression builds a color table for the image where each color value is assigned to pixels. This compression

method makes this image format ideal for line art, logos or other simple images without gradients or varying color.

In fact GIF comes in two flavors. The original specification GIF87a. The later version, GIF89a, supports simple animation via a Graphics Control Extension block in the data.

It is worthwhile examining the file format for GIF87 in more detail, since many such formats bear a resemblance to it but have grown a good deal more complex than this ―simple‖ standard. For the standard specification, the general file format is as in FIG 4-17. The signature is 6 bytes: GIF87a; the Screen Descriptor is a 7-byte set of flags. A GIF87 file contains more than one image definition, usually to fit on several different parts of the screen. Therefore each image can contain its own color lookup table, a *Local Color Map,* for mapping 8 bits into 24-bit RGB values. The GIF file format is as shown below:

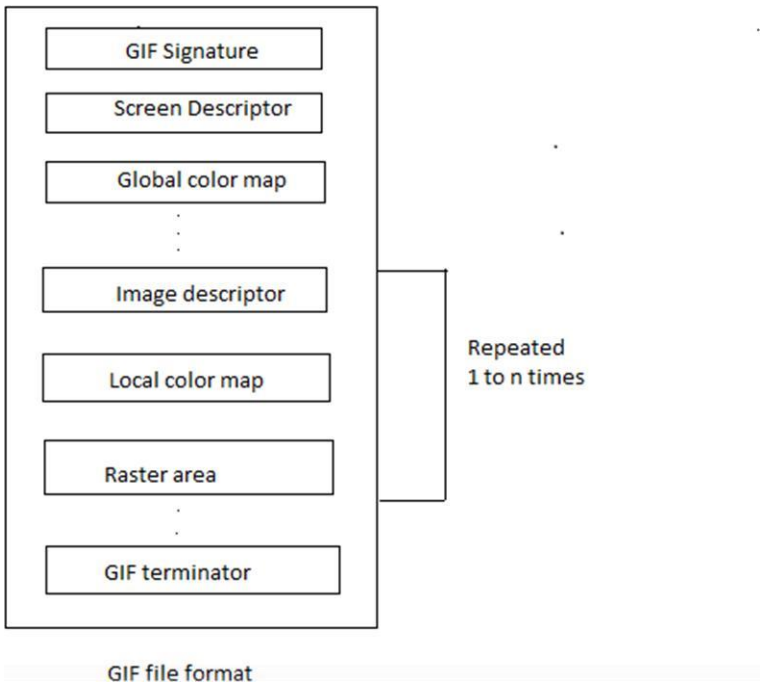


Figure 4-17 General format of GIF.

The screen Descriptor comprises a set of attributes that belong to every image inthe file. According to the GIF87 standard, it is defined as in FIG 4-18. *Screen Width* is given in the first 2 bytes. Since some machines invert the order MSB/LSB, this order is specified. *Screen Height* is the next 2 bytes. The _m‘in byte 5 is 0 if no global color map is given. Color resolution, ―Cr‖,is 3 bits in 0…..7. Since this is an old standard meant to operate on a variety of low-order hardware, ―Cr‘is requesting this much color resolution. The next bit, shown as ―0‖,is extra

and is not used in this standard. ―pixel‖is another 3 bits,, indicating the number of bits per pixel in the image, as stored in the file. Although ―Cr‖usually equals ―pixel‖,it need not. Byte 6 gives the color table index byte for the background color, and byte 7 is filled with zeros.

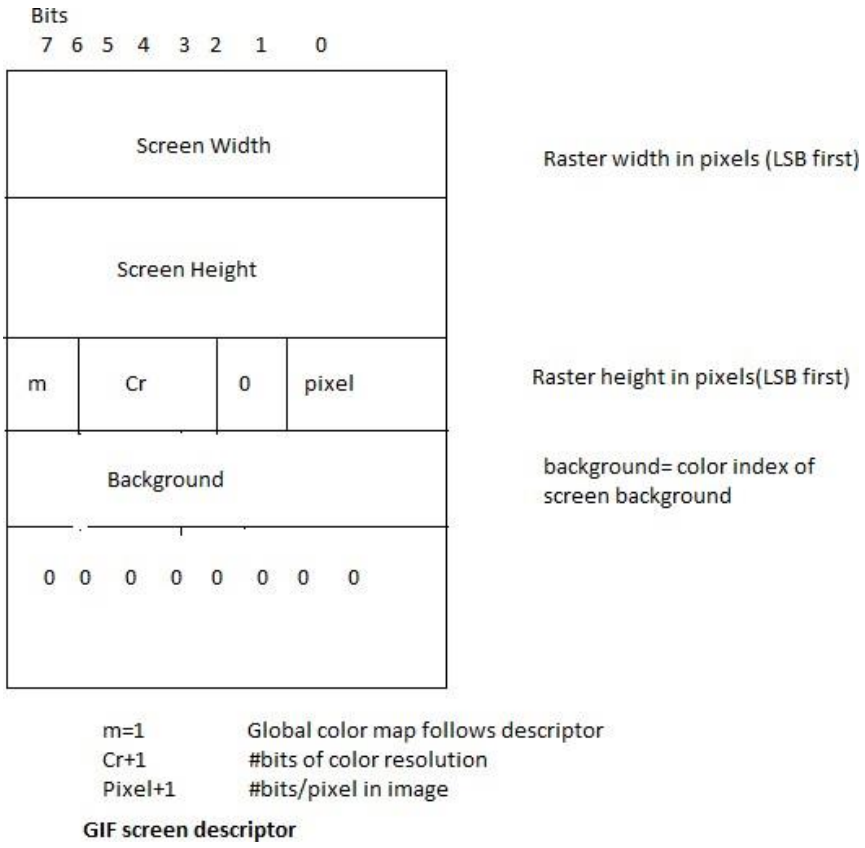


Figure 4-17 GIF Screen Descriptor

A color map is set up in a simple fashion, as in FIG 4-19. However, the actual length of the table is equals $2^{pixel+1}$ as given in the screen descriptor.

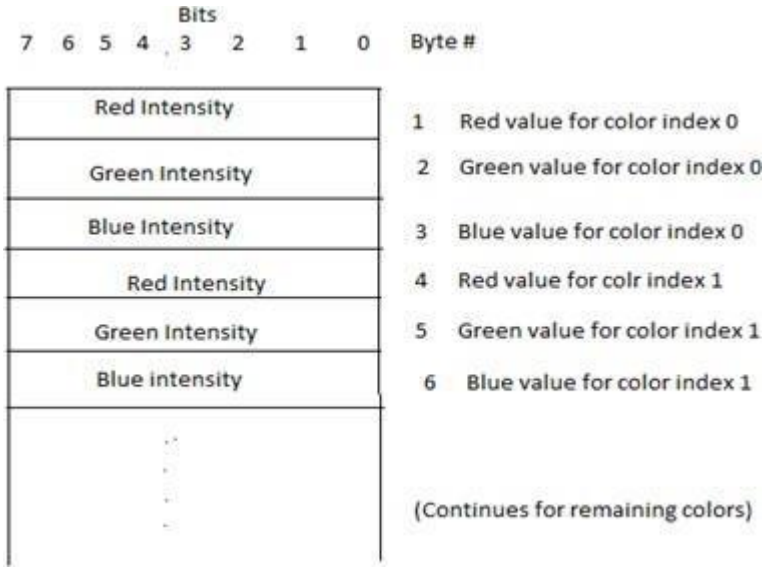


Figure 4-19 GIF color map

4.4 Input devices

Graphics workstations can make use of various devices for data input. Most systems have a keyboard and one more additional devices specifically designed for interactive input. These include a mouse, trackball, spaceball, and joystick. Some other input devices used in particular applications are digitizers, dials, button boxes, data gloves, touch panels, and voice systems.

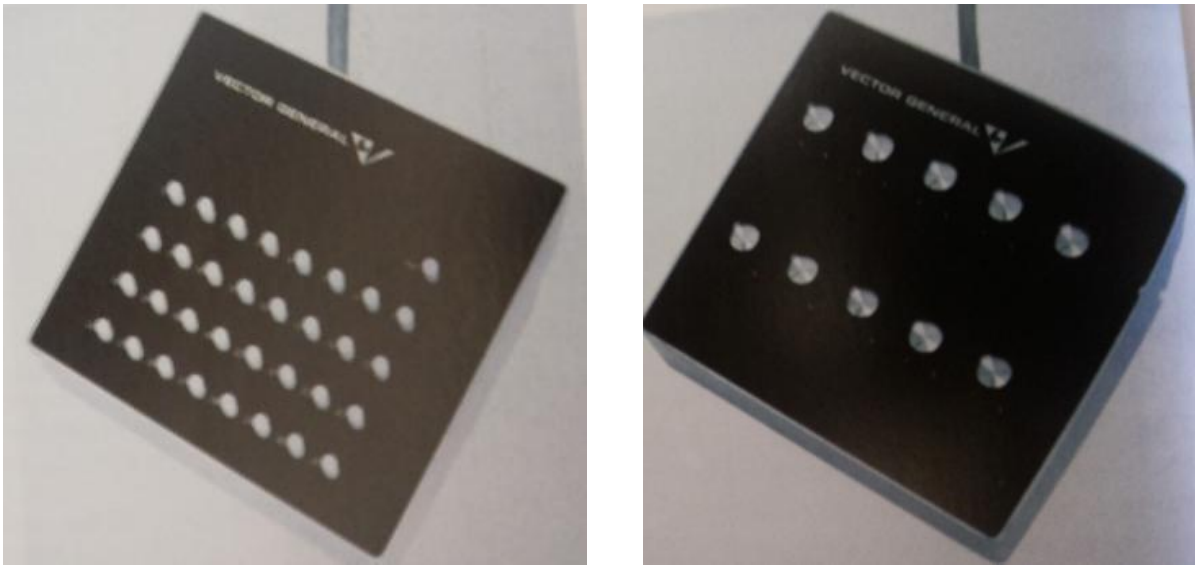
Keyboards, Button boxes, and Dials

An alphanumeric keyboard on a graphics system is used primarily as a device for entering text strings, issuing certain commands, and selecting menu options. The keyboard is an efficient device for inputting such nongraphic data as picture labels associated with a graphics display. Keyboards can also be provided with features to facilitate entry of screen coordinates, menu selections, or graphic functions.

Cursor-controlled keys and function keys are common features on general-purpose keyboards. Function keys allow users to select frequently accessed operations with a single keystroke, and cursor control keys are convenient for selecting a displayed object or a location by positioning the screen cursor. A keyboard can also contain other types of cursor-positioning devices, such as trackball o joystick, along with a numeric keyboard for fast entry of numeric data. In addition to these features, some keyboards have an ergonomic design (figure 4-20) that provides adjustments for relieving operator fatigue.

For specialized tasks, input to a graphics application may come from a set of buttons, dials, or switches that select data values or customized graphics operations. Figure 4-21 gives an example of a button box and a set of input dials**.** Buttons and switches are often used to input predefined functions, and dials are common devices for entering scalar values. Numerical values within some defined range are selected for input with dial rotations. A potentiometer is used to measure dial notation, which is then converted to the corresponding numerical value.

Figure 4-20 Ergonomically designed keyboard with removable palm resets.



(a) (b)

Figure 4-21 A button box (a) and a set of input details (b).

Joysticks

Another positioning device is **Joystick,** which consists of a small, vertical lever mounted on a base. We use the joystick to steer the screen cursor around. Most Joysticks, such as the unit figure 4-22, select screen positions with actual stick movement; others respond to pressure on the stick. Some joysticks are mounted on a keyboard, and some are designed as stand-alone units.

The distance that the stick is moved in any direction from its center position corresponds to the relative screen-cursor movement in that direction.

Potentiometers mounted at the base of the joystick measure the amount of movement, and springs return to the stick to the center position when it is released. One or more buttons can be executed once a screen position has been selected.

In another type of movable joystick, the stick is used to activate switches that cause the screen cursor to move at a constant rate in the direction selected. Eight switches, arranged in a circle, are sometimes are provided so that the stick can select any one of eight directions for cursor movement. Pressure-sensitive joysticks, also called *isometric joysticks*, have a non-movable stick. A push or pull on the stick is measured with strain gauges and converted to movement of the screen cursor in the direction of the applied pressure.

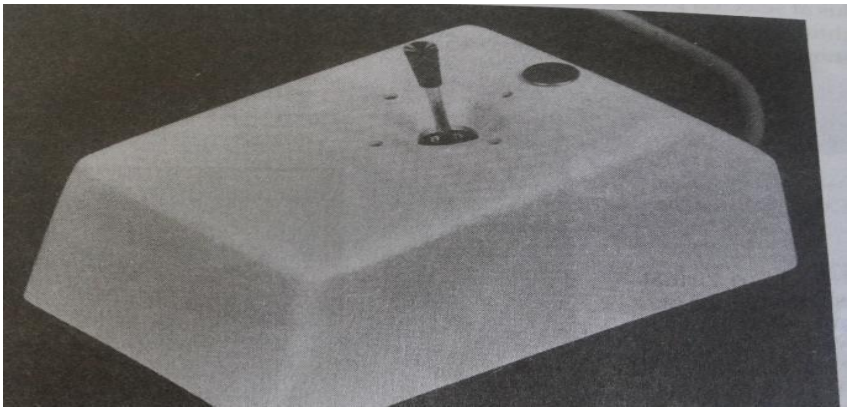


Figure 4-22 A movable Joystick.

Data Gloves

Figure 4-23 shows a **Data Glove** that can be used to grasp a ―virtual object‖. The glove is constructed with a series of sensors that detect hand and finger motions. Electromagnetic coupling between transmitting antennas and receiving antennas are used to provide information about the position and orientation of the hand. The transmitting and receiving antennas can each be structured as a set of three mutually perpendicular coils, forming a three-dimensional Cartesian reference system. Input from the gloves is used to position or manipulate objects in a virtual-scene. A two-dimensional projection of the scene can be viewed on a video monitor, or a three-dimensional projection can be viewed with a headset.

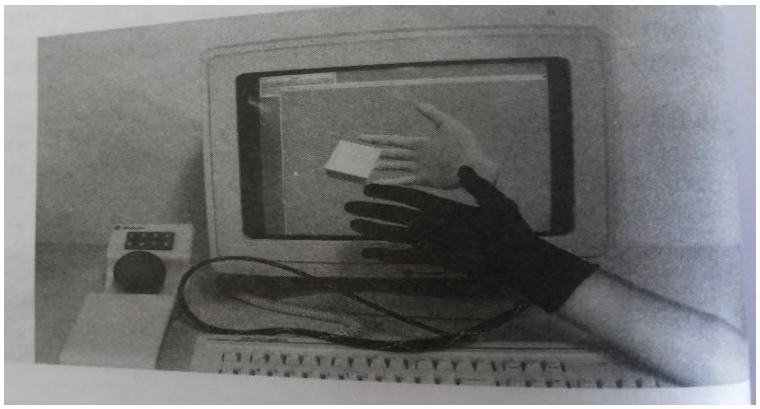


Figure 4-23 A virtual-reality scene, displayed on a two-dimensional video monitor, with input from a data glove and a spaceball.

Digitizers

A common device for drawing, painting, or interactively selecting position is a **Digitizer.** These devices can be designed to input coordinate values in either a two-dimensional or threedimensional space. In engineering or architectural applications, a digitizer is often usedto scan a drawing or object and input a set of discrete coordinate positions. The input positions are then joined with straight-line segments to generate an approximation of a curve or surface shape. One type of digitizer is the **Graphic Tablet (**also referred as data tablet), which is used to input two-dimensional coordinates by activating a hand cursor or stylus at selected positions on a flat surface. A hand cursor contains cross hairs for sighting positions, while a stylus is a pencilshaped device that is pointed at positions on the tablet. Figures 4-24 and 4-25 show example of desktop and floor model tablets**,** using hand cursors that are available with two, four, or sixteen buttons. Examples of stylus input with a tablet are shown in Figure 4-26 and 4-27 uses electromagnetic resonance to detect the three-dimensional position of the stylus. This allows an artist to produce different brush strokes by applying different pressures to the tablet surface. Tablet size varies from 12 by 12 inches for desktop models to 44 by 60 inches or larger for floor models. Graphics tablets provides a highly accurate method for selecting coordinate positions, with an accuracy that varies from about 0.2mm on desktop models to about 0.05mm or less on larger models.

Many graphics tablets are constructed with a rectangular grid of wires embedded in the tablet surface. Electromagnetic pulses are generated in sequence along the wires, and an electric signal is induced in a wire coil in an activated stylus or hand-cursor to record a tablet position. Depending on the technology, signal strength, coded pulses, or phase shifts can be used to determine the position on the tablet.

An *acoustic (sonic tablet)*used sound waves to detect a stylus position. Either strip microphones or point microphones can be employed to detect the sound emitted by timing the arrival of the generated sound at the different microphone positions. An advantage of two-dimensional acoustic tablets is that the microphonescan be placed on any surface to form the ―tablet‖work area. For example, the microphones could be placed on a book page while a figure on that page is digitized.

Three-dimensional digitizers use sonic or electromagnetic transmissions to record positions. One electromagnetic transmission method is similar to that employed in the data glove: a coupling between the transmitter and receiver is used to compute the location of a stylus as it moves over an object surface. Figure 4-28 shows a digitizer recording the locations of positions on the surface of a three-dimensional object. As the points are selected on a nonmetallic object, a wireframe outline of the surface displayed on the computer screen. Once the surface outline is constructed, it can be rendered using lighting effects to produce a realistic display of the object. Resolution for this system is from 0.8mm to 0.08mm, depending on the model.

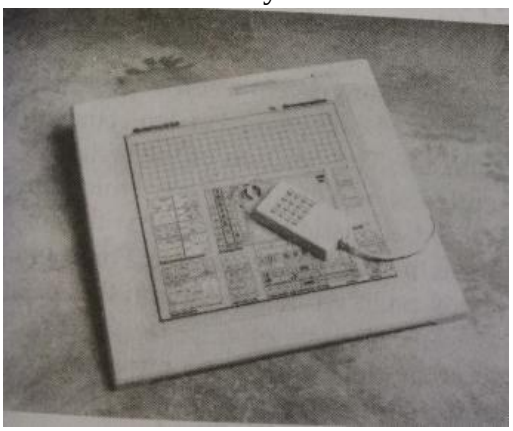


Figure 4-24 The SummenSketch III desktop with a sixteen-button hand cursor.

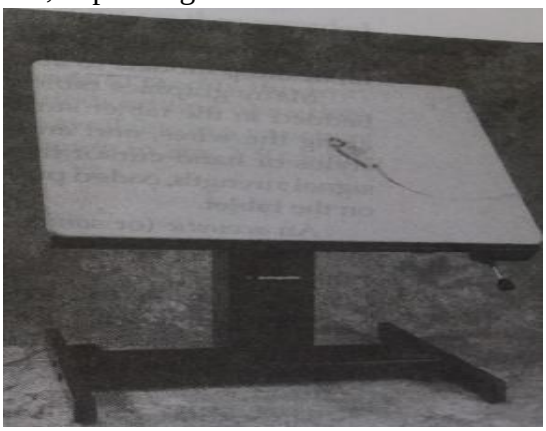


Figure 4-25 The Microgrid III tablet with a tablet sixteen-button hand cursor, designed for digitizing larger drwaings

Figure 4-26 The Notepad desktop Tablet with stylus.

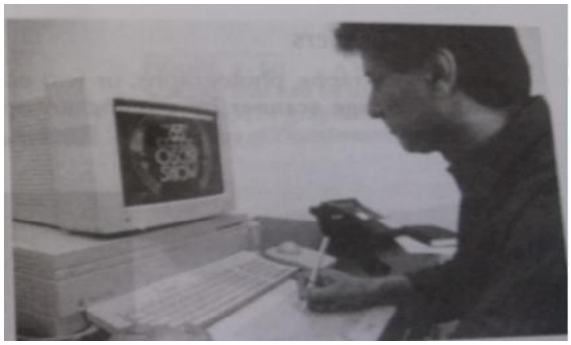
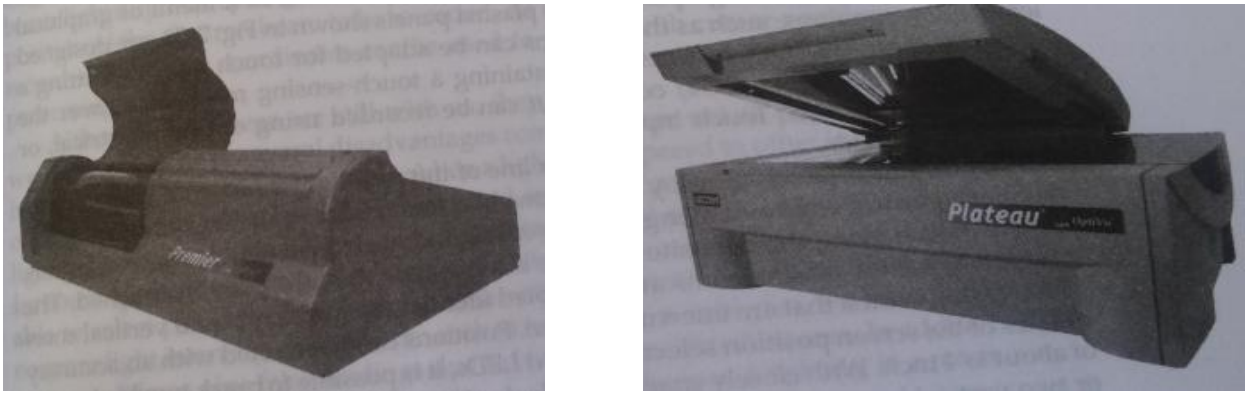


Figure 4-27 An artist's digitizer system, with a Pressure-sensitive, cordless stylus.

Image Scanners

Drawings, graphs, photographs, or text can be stored for computer processing with an **image scanner** by passing an optical scanning mechanism over the information to be stored. The gradations of gray scale or color are then recorded and stored in an array. Once we have the internal representation of a picture, we can apply transformations to rotate, scale, or crop the picture to a particular screen area. We can also apply various image-processing methods to modify the array representation of a picture. For scanned text input, various editing operations can be performed on the stored documents. Scanners are available in a variety of sizes and capabilities.A larger hand-model scanner is shown in **Figure 4-29 (a) and (b).**



(a) (b)

Figure 4-28 Desktop scanners: (a) drum scanner and (b) flatbed scanner.

Touch panels

As the name implies, **touch panels** allow displayed objects or screen positions to be selected with the touch of a finger. A typical application of touch panels is for the selection of processing options that are represented as a menu of graphical icons. Some monitors can be adapted for touch input by fitting a transparent device (**Fig 4-30)** containing a touch-sensing mechanism over the video monitor screen. Touch input can be recorded using optical, electrical, or acoustic methods.

(a) (b)

4-29 plasma panels with touch screens.

Optical touch panels employ a line of infrared light-emitting diodes (LEDs) along one vertical edge and along one horizontal edge of the frame. Light detectors are placed along the opposite vertical and horizontal edges. These detectors are used to record which beams are interrupted when the panel is touched. The two crossing beams that are interrupted identify the horizontal and vertical coordinates of the screen position selected. Positions can be selected with an accuracy of about ¼ inch. With closely spaced LEDs, it is possible to break two horizontal or two vertical beams simultaneously. In this case, an average position between the two interrupted beams is recorded. The LEDs operate at infrared frequencies so that the light is not visible is not visible to a user.

Light Pens

Figure 4-30 shows the design of one type of **light pens.** Such pencil-shaped devices are used to select screen positions by detecting the light coming from points on the CRT screen. They are sensitive to the short burst of light emitted from the phosphor coating at the instant the electron beam strikes a particular point. Other light sources, such as the background light in the room, are usually not detected by a light pen. An activated light pen, pointed at a spot on the screen as the electron beam lights up that spot, generates as electrical pulse that causes the coordinate position of the electron beam to be recorded. As with cursor-positioning devices, recorded light-pen coordinates can be used to position an object or to select a processing option.

Although light pens are still with us, they are not as popular as they once were since they have several disadvantages compared to other input devices that have been developed. For example, when a light pen is pointed as the screen, part of the screen image is obscured by the hand and pen. And prolonged use of the light pen cause arm fatigue. Also, light pens require special

implementations for some applications since they cannot detect positions within black areas. To be able to select positions in any screen area with a light pen, we must have some nonzero light intensity emitted from each pixel within that area. In addition, light pens sometimes give false readings due to background lightning in a room.

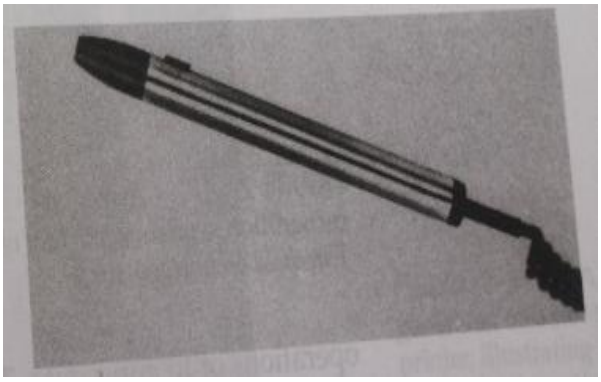


Figure 4-30 A light pen with a button activation.

Voice Systems

Speech recognizers are used with some graphics workstations as input devices for voice commands. The **voice system** input can be used to initiate graphics operations or to enter data. These systems operate by matching an input against a predefined dictionary of words and phrases.

A dictionary is set up by speaking the command words several times. The system then analyzes each word and establishes a dictionary of word frequency patterns, along with the corresponding functions that are to be performed. Later, when a voice command is given, the system searches the dictionary for a frequency-pattern match. A separate dictionary is needed for each operator using the system. Input for a voice system is typically spoken into a microphone mounted on a headset, as in Figure 4-31**,** and the microphone is designed to minimize input of background sounds. Voice systems have some advantage over other input devices, since the attention of the operator need not switch from one device to another to enter a command.

Figure 4-31 A speech recognition system.

4.5 Video display devices

Typically, the primary output device in a graphics system is a video monitor. The operation of most video monitors is based on the standard **cathode-ray tube (CRT)** design, but several other technologies exist and solid state monitors may eventually predominate.

Refresh Cathode-Ray Tubes

Figure 4-32 illustrates the basic operation of a CRT. The **cathode ray tube** (**CRT**) is a vacuum tube containing one or more electron guns (a source of electrons or electron emitter) and a fluorescent screen used to view images. It has a means to accelerate and deflect the electron beam(s) onto the screen to create the images. The images may represent electrical waveforms (oscilloscope), pictures (television, computer monitor), radar targets or others. CRTs have also been used as memory devices, in which case the visible light emitted from the fluorescent material is not intended to have significant meaning to a visual observer.

The CRT uses an evacuated glass envelope which is large, deep (i.e. long from front screen face to rear end), fairly heavy, and relatively fragile. As a matter of safety, the face is typically made of thick lead glass so as to be highly shatter-resistant and to block most X-ray emissions, particularly if the CRT is used in a consumer product.

CRTs have largely been superseded by newer display technologies such as LCD, plasma display, and OLED, which have lower manufacturing and distribution costs.

In television sets and computer monitors, the entire front area of the tube is scanned repetitively and systematically in a fixed pattern called a raster. An image is produced by controlling the

intensity of each of the three electron beams, one for each additive primary color (red, green, and blue) with a video signal as a reference. In all modern CRT monitors and televisions, the beams are bent by *magnetic deflection*, a varying magnetic field generated by coils and driven by electronic circuits around the neck of the tube, although electrostatic deflection is commonly used in oscilloscopes, a type of diagnostic instrument

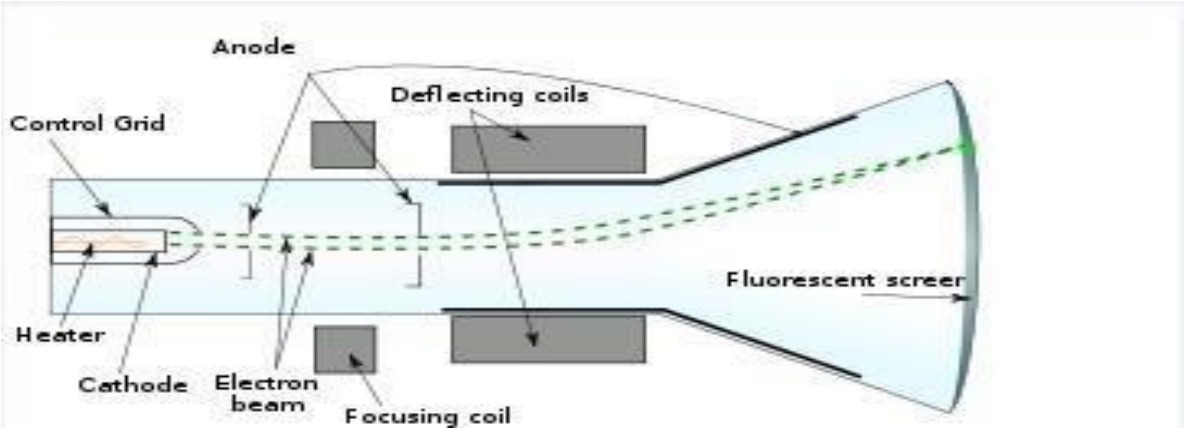


Figure 4-32 Electrostatic deflection of the electron beam in a CRT.

Raster-scan displays

The most common type of graphics monitor employing a CRT is the **raster-scan display,** based on television technology. In a raster-scan system, the electron beam is swept across the screen, one row at a time, from top to bottom. Each row is referred to as a scan-line. As the `electron beam moves across a scan line, the beam intensity is turned on and off to create a pattern of illuminated spots. Picture definition n is stored in a memory area called the **refresh buffer,** wherethe term **frame** refers to the total screen area. This memory area holds the set of color values for the screen points. These stored color values are then retrieved from the refresh buffer and used to control the intensity of the electron beam as it moves from spot to spot across the screen. In this way, the picture is ―pointed‖on the screen one scan line at a time, as demonstrated

in **fig 4-33.** Each screen spot that can be illuminated by the electron beam is referred to as a **pixel or pel.** Since the refresh buffer is used to store the set of screen color values, it is also sometimes called a **color buffer.** Also, other kinds of pixel information, besides color, are stored in buffer locations, so all the different buffer areas are sometimes referred to collectively as the ―frame buffer‖. The capability of a raster-scan system to store color information for each screen point

makes it well suited for the realistic display of scenes containing subtle shading and color patterns.

Raster systems are commonly characterized by their resolution, which is the number of pixel positions that can be plotted. Another property of video monitors is **aspect ratio,** which is now often defined as the number of pixel columns divided by the number of scanned lines that can be displayed by the system.

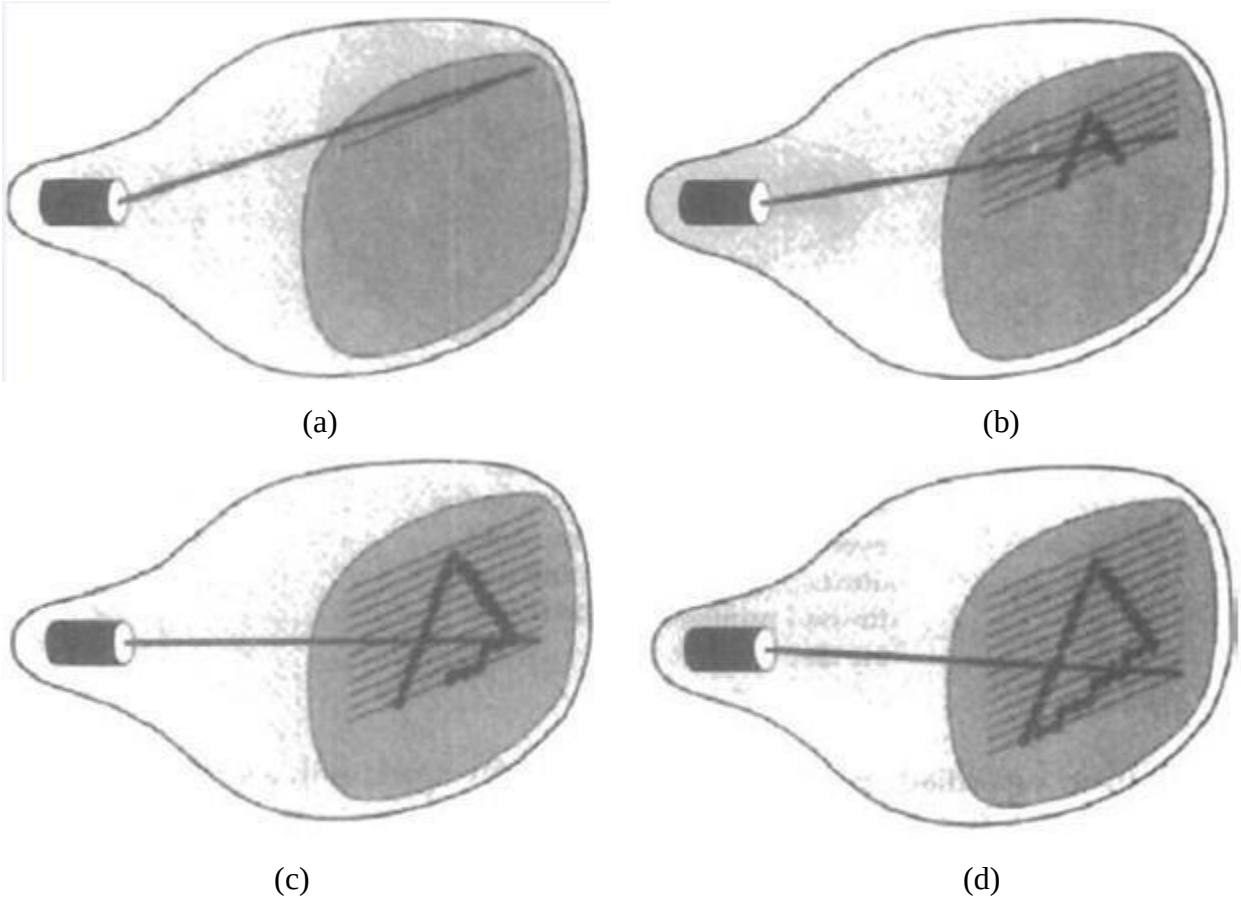


Figure 4-33 A raster-scan system displays an object as a set of discrete points across each scan line.

Random-Scan displays

When operated as a **Random-scan display unit,** a CRT has the electron beam directed only to those parts of the screen where a picture is to be displayed. Pictures are generated as line drawings, with the electron beam tracing out the component lines one after the other. For this reason, random-scan monitors are also referred to as **vector displays (**or **stroke-writing displays or calligraphic displays).** The component lines of a picture can be drawn and refreshed by a random-scan system in any specified order **(fig 4-34)**. A pen plotter operates in a similar way and is an example of a random-scan, hard-copy device.

Refresh rate on a random-scan systems depends on the number of lines to be displayed on that system. Picture definition is now stored as a set of line-drawing commands in an area of memory referred to as the **display list, refresh display file, vector file, or display program.** To display a specific picture, the system cycles through the set of commands in the display file, drawing each component line in turn. After all line-drawing commands have been processed, the system cycles back to the first line command in the list. Random-scan displays are designed to draw all the component lines in a picture 30 to 60 times each second, with up to 100,000 ―short‖ lines in the display list. When a small set of lines is to be displayed, each refresh cycle is delayed to avoid very high refresh rates, which could burn out the phosphor.

Random-scan systems were designed for line-drawing applications, such as architectural and engineering layouts, and they cannot display realistic shaded scenes. Since picture definition is stored as a set of line-drawing instructions rather than a set of intensity values for all screen points, vector displays generally have higher resolutions than raster system. Also, vector displays produce smooth line drawings because the CRT beam directly follows the line path. A raster system, by contrast produced jagged lines that are plotted as discrete point sets. However, the greater flexibility and improved line-drawing capabilities of raster systems have resultant in the abandonment of vector technology.

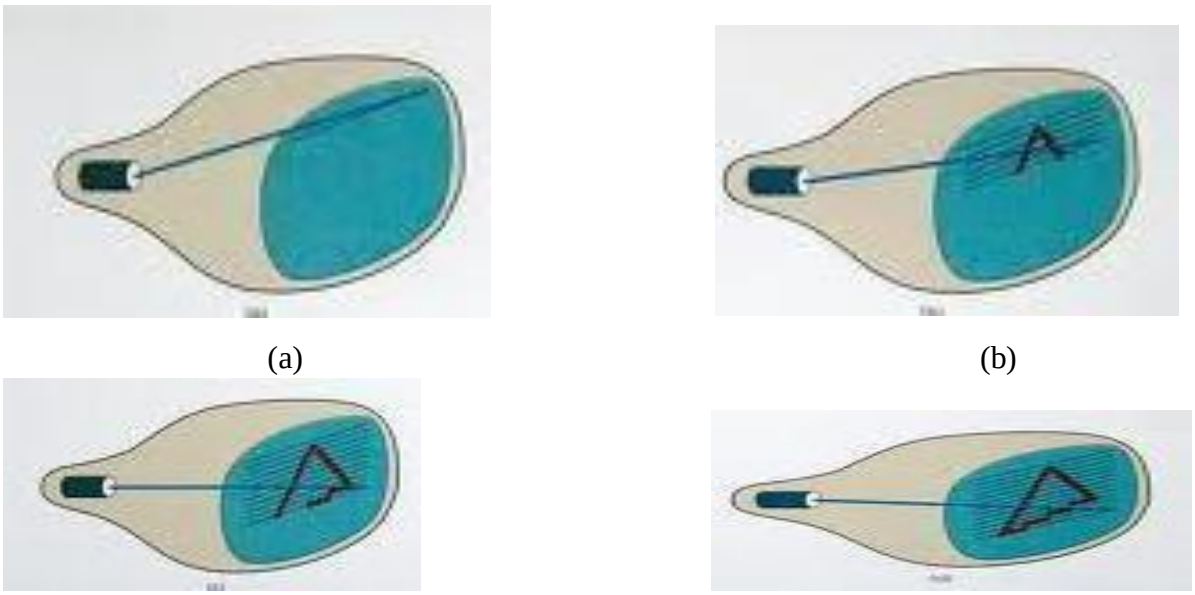


Figure 4-34 A random-scan system draws the component lines of an object in ant specified order.

Color CRT monitors

A CRT monitor displays color pictures by using a combination of phosphors that emit-colored light. The emitted light from the different phosphors merges to form a single perceived color, which depends on the particular set of phosphors that have been excited.

One way to display color pictures is to coat the screen with layers of different colored phosphors. The emitted color depends on how far the electron beam penetrates into the phosphors layers. This approach, called the **beam-penetration** method, typically used only two phosphor layers: red and green. A beam of sloe electrons excite only the outer layer, but a beam of very fast electrons penetrates through the red layer and excites the inner green layer. At intermediate beam speeds, combination of red and green light emitted to show two additional colors, orange and yellow. The speed of electrons, and hence the screen color at any point, is controlled by the beam acceleration voltage. Beam penetration has been inexpensive way to produce color, but only a limited number of colors are possible, and picture quality is not good as with other methods. **Shadow-mask** methods are commonly used in raster-scan systems since they produce much wider range of colors than the beam-penetration method. This approach is based on the way that we seem to perceive colors as combinations of red, green, and blue components, called the **RGB color model.** Thus, a shadow-mask CRT uses three phosphor color dots at each pixel position.

One phosphor dot emits a red light, another emits a green light, and the third emits a blue light. This type of CRT has three electron guns, one for each color dot, and a shadow-mask grid just behind the phosphor-coated screen. The light emitted from the three phosphors results in a small spot of color at each pixel position, since our eyes tend to merge the light emitted from the three dots into one composite color. Figure 4-35 illustrates the *delta-delta* shadow-mask method, commonly used in color CRT systems. The three electron beams are deflected and focused as a group onto the shadow mask, which contains a series of holes aligned with the phosphor-dot patterns. When the three beams pass through a hole in the shadow mask, they activate a dot triangle, which appears as a small color spot on the screen. The phosphor dots in the triangles are arranged so that each electron beam can activate only its corresponding color dot when is passes through the shadow mask. Another configuration for the three electron guns is an *in-line* arrangement in which the three electron guns, and the corresponding red-green-blue color dots on the screen, are aligned along one scan line instead of in a triangular pattern. This in-line arrangement of electron guns is easier to keep in alignment and is commonly used in high resolution color CRTs.

We obtain color variations in a shadow-mask CRT by varying the intensity levels of the three electron beams. By turning off two of the three guns, we get only the color coming from the single activated phosphor (red, green, or blue). When all three dots are activated with equal beam intensities, we see a white color. Yellow is produced with equal intensities from the green and red dots only, magenta is produced with equal blue and red intensities, and cyan shows up when blue and green are activated equally. In an inexpensive system, each of the three electron beams might be restricted to either on or off, limiting displays to eight colors. More sophisticated systems can allow intermediate intensity levels to be set for the electron beams, so that several million colors are possible.

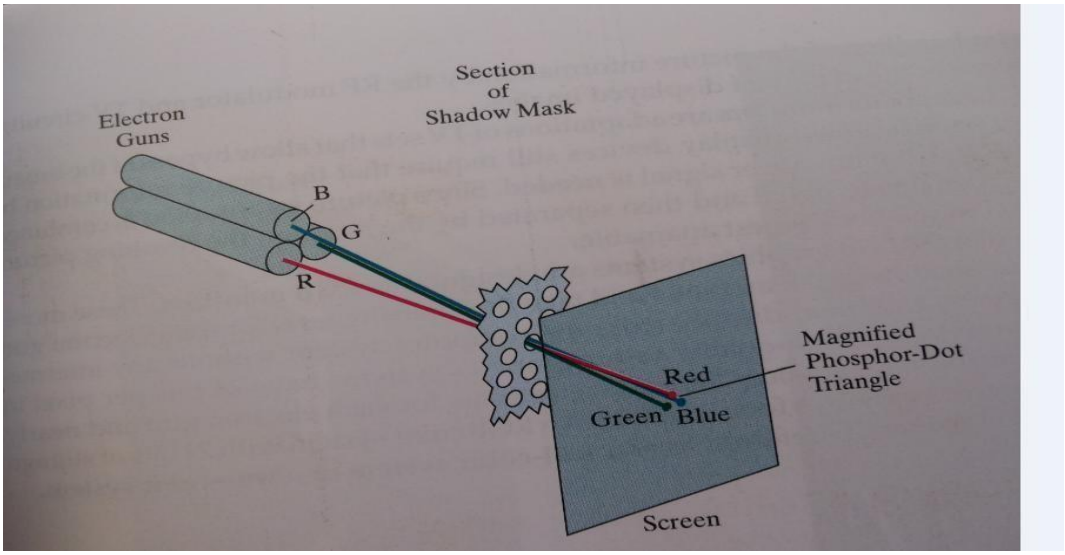


Figure 4-35 Operation of a delta-delta, shadow-mask CRT.

Flat-Panel displays

Although most graphics are still constructed with CRTs, other technologies are emerging that may soon replace CRT monitors. The term **flat-panel display** refers to a class of video devices that have reduced volume, weight, and power requirements compared to CRT. A significant feature of a flat-panel display is that they are thinner than CRTs, and we can hang them on walls or wear them on our wrists. Since we can even write on some flat-panel displays, they are also available as pocket notepads. Some additional uses for flat-panel displays are as small TV monitors, calculator screens, pocket video-game screens, laptop computer screens, armrest movie- viewing stations on airlines, advertisement boards in elevators, and graphics displays in applications requiring rugged portable monitors.

We can separate flat-panel displays into two categories: **emissive displays and non-emissive displays.** The emissive displays (or **emitters)** are devices that convert electrical energy into light. Plasma panels, thin-film electroluminescent displays, and light-emitting diodes are examples of emissive displays. Flat CRTs have also been devised, in which electron beams are accelerated parallel to the screen and then deflected 90^0 onto the screen. But flat CRTs have not proved to be as successful as other emissive devices. Non-emissive displays (or **non-emitter**) use optical effects to convert sunlight or light from some other source into graphics patterns. The most important example of a non-emissive flat-panel display is a liquid-crystal device.

Plasma panels, also called **gas-discharge displays,** are constructed by filling the region between two glass plates with a mixture of gases that usually includes neon. A series of vertical conducting ribbons is built into the other glass panel (figure 4-37). Firing voltages applied to an intersecting pair of horizontal and vertical conductors cause the gas at the intersection of the two conductors to break down into glowing plasma of electrons and ions. Picture definition is stored in a refresh buffer, and the firing voltages are applied to refresh the pixel positions 60 times per second. Alternating-current methods are used to provide faster application of the firing voltages and, thus, brighter displays. Separation between pixels is provided by the electric field of the conductors. Figure 4-38 shows a high definition plasma panel. One disadvantage of plasma panels has been that they were strictly monochromatic devices, but systems are now available with multicolor capabilities.

Thin-film electroluminescent displays are similar in construction to plasma panels. The difference is that the region between the glass plates is filled with a phosphor, such as zinc sulfide doped with manganese, instead of a gas (figure 4-39). When a sufficiently high voltage is applied to a pair of crossing electrodes, the phosphor becomes a conductor in the area of the intersection of the two electrodes. Electrical energy is absorbed by the manganese atoms, which then release the energy as a spot of light similar to the glowing plasma effect in a plasma panel. Electromagnetic displays require more power than plasma panels, and good color displays are harder to achieve.

A third type of emissive devices is the **light-emitted diode (LED).** A matrix of diodes is arranged to form the pixel positions in the display, and picture definition is stored in a refresh buffer. As in scan-line refreshing of a CRT, information is read from the refresh and converted to voltage levels that are applied to the diodes to produce the light patterns in the display.

Liquid-crystal displays (LCDs) are commonly used in small systems, such as laptop computers and calculators (figure 4-39)**.** These non-emissive devices produce a picture by passing polarized light from the surroundings or from an internal light source through a liquid-crystal material that can be aligned to either block or transmit the light.

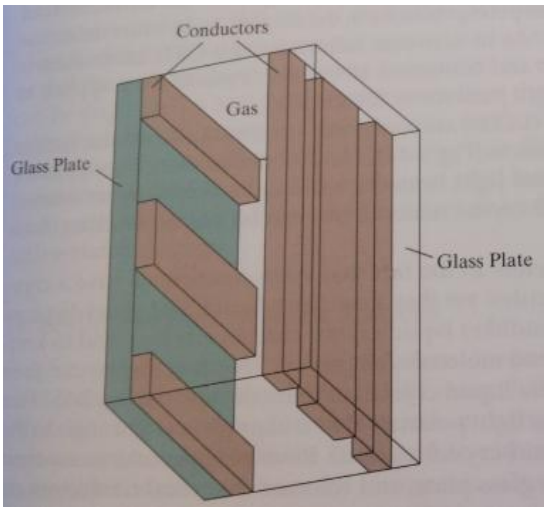


Fig 4-36 basic design of a Plasma-panel display

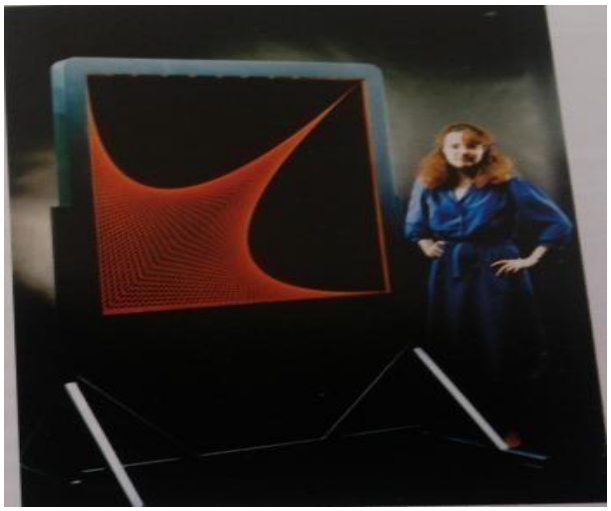
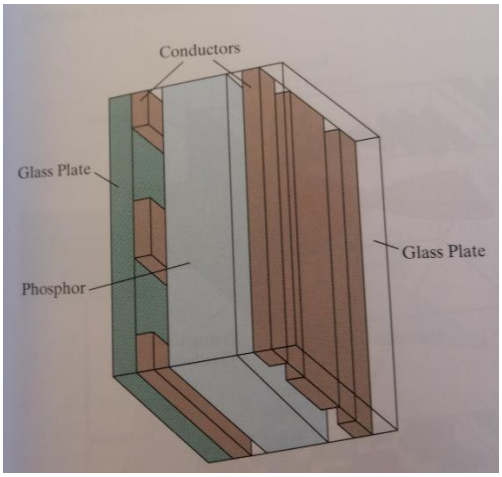
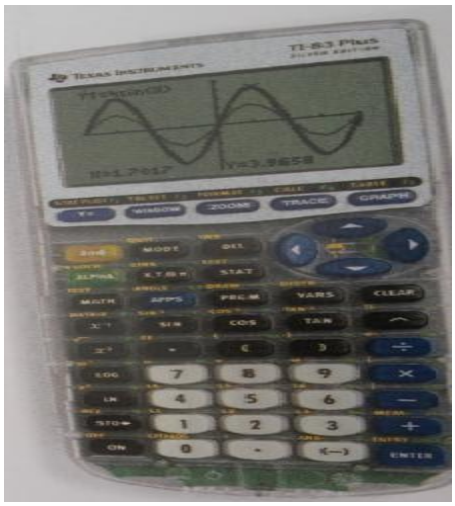


Figure [1]-37 A plasma-panel display with a resolution of 2048 by 2048 and a screen diagonal of 1.5 meters.





[1] .6 Summary

In this unit we provided basic information of Vector graphics and Bitmap graphics, their features. Vector graphics are based on vectors (also called paths, or strokes) which lead through locations called control points whereas, a **raster graphics** image, or **bitmap**, is a dot matrix data structures representing a generally rectangular grid of pixels.

This chapter gives brief discussion about the JPEG compression techniques. It briefs about the different types of JPEG modes (both lossy and lossless modes). We discussed about GIF file format description.

Figure 4-38 Basic design of a thin-film electroluminescent display device. **Figure 4-39 A handheld calculator with an LCD screen.**

The predominant graphics display device is the raster refresh monitor, based on television technology. A raster system uses a frame buffer to store the color value for each screen position. Pictures are then painted onto the screen by retrieving the information from the frame buffer as the electron beam in the CRT sweeps across each scan line, from top to bottom. Older vector displays construct pictures by drawing straight-line segments between specified endpoint positions. Picture information is then stored as a set of line-drawing instructions.

For graphical input, we have a range of devices to choose from. Keyboards, button boxes, and dials are used to input text, data values, or programming options. The most popular ―pointing‖ devices is the mouse, but trackballs, spaceballs, joysticks, cursor-controlled keys, and thumbwheels are also used to position the screen cursor. In virtual-reality environments, data gloves are commonly used. Other input devices are image scanners, digitizers, touch panels, light pens, and voice systems.

7. Keywords

Vector graphics, Bitmap graphics, JPEG, JPEG modes, Image preparation, Interleaved processing, Lossy sequential DCT-Based mode, Quantization, Entropy Encoding, Expanded Lossy DCT-Based mode, Lossless mode, GIF, Keyboards, Button boxes, Dials, joysticks, Data gloves, Digitizers, Image scanners, touch Panels, light pens, Voice systems, video display devices, CRT, Raster-scan displays, random-scan display, Color CRT monitors, Flat-panel monitors.

8. Exercises

1. Differentiate between Vector graphics and Bitmap graphics.
2. What are the different types of JPEG modes? 3. Explain briefly Sequential DCT-Based mode.
4. Discuss briefly about Expanded lossy DCT-Based mode and Lossless mode.
5. Explain the GIF file format.
6. Explain different types of multimedia input devices.
7. Explain Random and raster-scan displays, in detail.
8. Explain Flat-panel display in detail.

4.9 References

1. Ralf Steinmentz ,klaraNaestedt: Multimedia Fundamentals: Vol 1- Media Coding and Content processing, 2nd edition, PHI, Indian Reprint 2008.
2. Prabhat K. Andleigh, kiranThakrar,Multimedia Systems Design, PHI,2003.
3. Ze-Nian Li-Mark S Drew, Fundamentals of Multimedia, PHI, New Delhi .2011.
4. Donald Hearn and M. Pauline Baker, computer graphics, 3rd edition, Pearson.

MODULE-2

UNIT-5 Multimedia File Formats

Structure

1. Objectives

After studying this unit, you will be able to,

- Discuss different types of file formats such as, Bitmaps, Vector, Metafile, Animation, Scene, Multimedia, Hybrid, Hypertext file formats.

1. Introduction

There are a number of different types of graphics file formats. Each type stores graphics data in a different way. Bitmap, vector, and metafile formats are by far the most commonly used formats,

and we focus on these. However, there are other types of formats as well--scene, animation, multimedia, hybrid, hypertext, hypermedia, 3D, virtual modeling reality language (VRML), audio, font, and page description language (PDL). The increasing popularity of the World Wide

2. Bitmap Formats

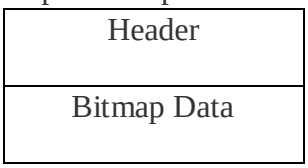
Bitmap formats are used to store bitmap data. Files of this type are particularly well-suited for the storage of real-world images such as photographs and video images. Bitmap files, sometimes called ***raster files***, essentially contain an exact pixel-by-pixel map of an image. A rendering application can subsequently reconstruct this image on the display surface of an output device.

Microsoft BMP, PCX, TIFF, and TGA are examples of commonly used bitmap formats. The following section is describes the construction of bitmap files in some detail.

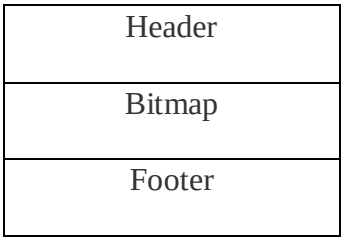
Bitmap files consist of a header, bitmap data, and other information, which may include a color palette and other data.

1. Organization of Bitmap file format

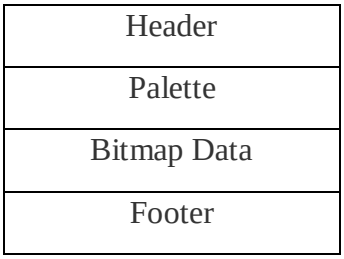
The basic components of a simple bitmap file are the following:



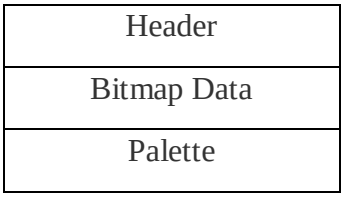
If a file contains no image data, only a header will be present. If additional information is required that does not fit in the header, footer will usually be present as well:



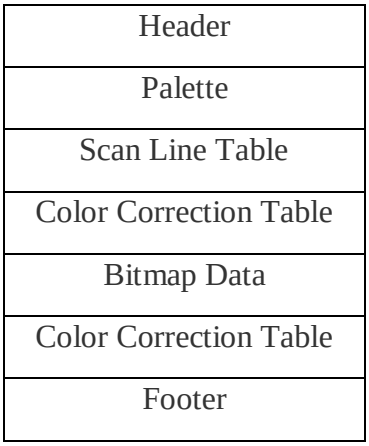
An image file may store a palette in the header, but it will more likely appear immediately after the header:



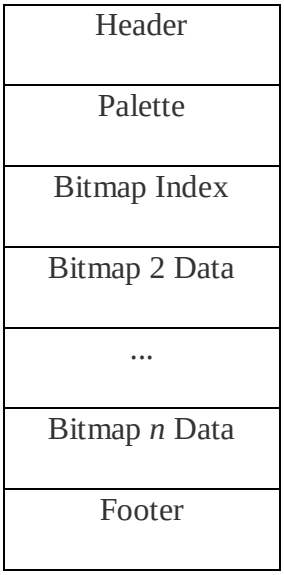
A palette can also appear immediately after the image data, like footer, or be stored in the footer itself:



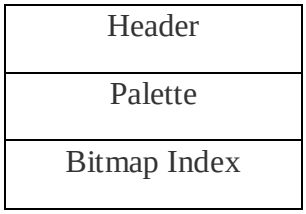
Scan-line tables and ***color correction tables*** may also appear after the header and before or after the image data:



If an image file format is capable of holding multiple images, then an ***image file index*** may appear after the header, holding the offset values of the starting positions of the images in the file:



If the format definition allows each image to have its own palette, the palette will most likely appear before the image data with which it is associated:



Palette 1
Bitmap Data
Palette 2
Bitmap 2 Data
...
Palette *n*
Bitmap *n* Data
Footer

5.2.2Header Structure

The header is a section of ***binary-*** **or** ***ASCII-format*** data normally found at the beginning of the file, containing information about the bitmap data found elsewhere in the file. All bitmap files have some sort of header, although the format of the header and the information stored in it varies considerably from format to format. Typically, a bitmap header is composed of fixed fields. None of these fields is absolutely necessary, nor are they found in all formats, but this list is typical of those formats in widespread use today. The following information is commonly found in a bitmap header:

Header
Palette
Bitmap Index
Palette 1
File Identifier
File Version
Number of Lines per Image
Number of Pixels per Line
Number of Bits per Pixel
Number of Color Planes
Compression Type
X Origin of Image

Y Origin of Image
Text Description
Unused Space

File Identifier

A header usually starts with some sort of unique identification value called a ***file identifier***, ***file ID,*** **or** ***ID value***. Its purpose is to allow a software application to determine the format of the particular graphics file being accessed.

ID values are often ***magic values*** in the sense that they are assigned arbitrarily by the creator of the file format. They can be a series of ASCII characters, such as BM or GIF, or a 2- or 4-byte word value, such as 4242h or 596aa695h, or any other pattern that made sense to the format creator. The pattern is usually assumed to be unique, even across platforms, but this is not always the case, as we describe in the next few paragraphs. Usually, if a value in the right place in a file matches the expected identification value, the application reading the file header can assume that the format of the image file is known.

File Version

Following the identification value in the header is usually a field containing the file version. Naturally enough, successive versions of bitmap formats may differ in characteristics such as header size, bitmap data supported, and color capability. Once having verified the file format through the ID value, an application will typically examine the version value to determine if it can handle the image data contained in the file.

Image Description Information

Next comes a series of fields that describe the image itself. As we will see, bitmaps are usually organized, either physically or logically, into lines of pixels. The field designated ***number of lines per image,*** also called the ***image length, image height,*** **or** ***number of scan lines,*** holds a value corresponding to the number of lines making up the actual bitmap data. The ***number of pixels per line***, also called the ***image width*** **or** ***scan-line width***, indicates the number of pixels stored in each line.

The number of bits per pixel indicates the size of the data needed to describe each pixel per ***color plane***. This may also be stored as the ***number of bytes per pixel***, and is more properly called *pixel depth*. Forgetting the exact interpretation of this field when coding format readers is a

common source of error. If the bitmap data is stored in a series of planes, the ***number of color planes*** indicates the number of planes used. Often the value defaults to one. There is an increasing tendency to store bitmaps in single-plane format, but multi-plane formats continue to be used in support of special hardware and alternate color models.

The number of bits in a line of the image can be calculated by multiplying the values of ***number of bits per pixel, number of pixels per line***, and ***number of color planes*** together. We can determine the number of bytes per scan line by then dividing the resulting product by eight. Note that there is nothing requiring ***number of bits per pixel*** to be an integral number of 8-bit bytes.

Compression Type

If the format supports some sort of ***encoding*** designed to reduce the size of the bitmap data, then a ***compression type*** field will be found in the header. Some formats support multiple compression types, including ***raw*** **or** ***uncompressed*** data. Some format revisions consist mainly of additions or changes to the compression scheme used. Data compression is an active field, and new types of compression accommodating advances in technology appear with some regularity.

TIFF is one of the common formats which have exhibited this pattern in the past.

x and y Origins

x origin of image and ***y origin of image*** specify a coordinate pair that indicates where the image starts on the output device. The most common origin pair is (0,0), which puts one corner of the image at the origin point of the device. Changing these values normally causes the image to be displayed at a different location when it is rendered.

Most bitmap formats were designed with certain assumptions about the output device in mind, and thus can be said to model either an actual or virtual device having a feature called the ***drawing surface*****.** The drawing surface has an implied origin, which defines the starting point of the image, and an implied orientation, which defines the direction in which successive lines are drawn as the output image is rendered. Various formats and display devices vary in the positioning of the origin point and orientation direction. Many place the origin in the upper-left corner of the display surface, although it can also appear in the center, or in the lower-left corner. Others, although this is far less common, put it in the upper- or lower-right corner.

Orientation models with the origin in the upper-left corner are often said to have been created in support of hardware, and there may be some historical and real-world justification for this. People with backgrounds in mathematics and the natural sciences, however, are used to having the origin in the lower-left corner or in the center of the drawing surface. You might find yourself guessing at the background of the format creator based on the implied origin and orientation found in the format. Some formats include provisions for the specification of the origin and orientation.

An image displayed by an application incorporating an incorrect assumption about the origin point or orientation may appear upside down or backwards, or may be shifted horizontally some fraction of the width of the drawing surface on the output device.

Sometimes the header will contain a text description field, which is a comment section consisting of ASCII data describing the name of the image, the name of the image file, the name of the person who created the image, or the software application used to create it. This field may contain 7-bit ASCII data, for portability of the header information across platforms.

Unused Space

At the end of the header may be an unused field, sometimes referred to as ***padding, filler, reserved space,*** or ***reserved fields***. Reserved fields contain no data, are undocumented and unstructured and essentially act as placeholders. All we know about them are their sizes and positions in the header. Thus, if the format is altered at some future date to incorporate new data, the reserved space can be used to describe the format or location of this data while still maintaining backward compatibility with programs supporting older versions of the format. This is a common method used to minimize version problems--creating an initial version based on a fixed header substantially larger than necessary. New fields can then be added to reserved areas of the header in subsequent revisions of the format without altering the size of the header.

Often format headers are intentionally padded using this method to 128, 256 or 512 bytes. This has some implications for performance, particularly on older systems, and is designed to accommodate common read and write buffer sizes. Padding may appear after the documented fields at the end of the header, and this is sometimes an indication that the format creator had performance and caching issues in mind when the format was created.

Reserved fields are sometimes only features left over from early working versions of the format, unintentionally frozen into place when the format was released. A vendor will normally change or extend a file format only under duress, or as a rational response to market pressure typically caused by an unanticipated advance in technology. In any case, the upgrade is almost always unplanned. This usually means that a minimal amount of effort goes into shoehorning new data into old formats. Often the first element sacrificed in the process is complete backward compatibility with prior format versions.

Bitmap Data

In many bitmap file formats the actual bitmap data is found immediately after the end of the file header. It may be found elsewhere in the file, however, to accommodate a palette or other data structure which also may be present. If this is the case, an offset value will appear in the header or in the documentation indicating where to find the beginning of the image data in the file.

How Bitmap Data Is Written to Files?

Before an application writes an image to a file, the image data is usually first assembled in one or more blocks of memory. These blocks can be located in the computer's main memory space or in part of an auxiliary data collection device. Exactly how the data is arranged then depends on a number of factors, including the amount of memory installed, the amount available to the application, and the specifics of the data acquisition or file write operation in use. When bitmap data is finally written to a file, however, only one of two methods of organization is normally used: **scan-line data or planar data**.

Scan-line data

The first, and simplest, method is the organization of pixel values into rows or scan lines, briefly mentioned above. If we consider every image to be made up of one or more scan lines, the pixel data in the file describing that image will be a series of sets of values, each set corresponding to a row of the image. Multiple rows are represented by multiple sets written from start to end in the file. This is the most common method for storing image data organized into rows.

If we know the size of each pixel in the image, and the number of pixels per row, we can calculate the offset of the start of each row in the file. For example, in an 8-bit image every pixel value is one byte long. If the image is 21 pixels wide, rows in the file are represented by sets of

pixel values 21 bytes wide. In this case, the rows in the file start at offsets of 0, 21, 42, 63, etc. bytes from the start of the bitmap data.

On some machines and in some formats, however, rows of image data must be certain even-byte multiples in length. An example is the common rule requiring bitmap row data to end on longword boundaries, where a long word is four bytes long. In the example mentioned in the preceding paragraph, an image 21 pixels wide would then be stored in the file as sets of pixel values 24 bytes in length, and the rows would start at file offsets 0, 24, 48, 64. The extra three bytes per row are padding. In this particular case, three bytes of storage in the file are wasted for every row, and in fact, images that are 21 pixels wide take up the same amount of space as images 24 pixels wide. In practice, this storage inefficiency is usually (but not always) compensated for by an increase of speed gained by catering to the peculiarities of the host machine in regard to its ability to quickly manipulate two or four bytes at a time. The actual

width of the image is always available to the rendering application, usually from information in the file header.

In a 24-bit image, each image pixel corresponds to a 3-byte long pixel value in the file. In the example we have been discussing, an image 21 pixels wide would require a minimum of 21 * 3 = 63 bytes of storage. If the format requires that the row starts be long-word aligned, 64 bytes would be required to hold the pixel values for each row. Occasionally, as mentioned above, 24bit image data is stored as a series of 4-byte long pixel values, and each image row would then require 21 * 4 = 84 bytes. Storing 24-bit image data as 4-byte values have the advantage of always being long-word aligned, and again may make sense on certain machines.

In a 4-bit image, each pixel corresponds to one-half byte, and the data is usually stored two pixels per byte, although storing the data as 1-byte pixel values would make the data easier to read and, in fact, is not unheard of.

Figure 1-1 illustrates the organization of pixel data into scan lines.

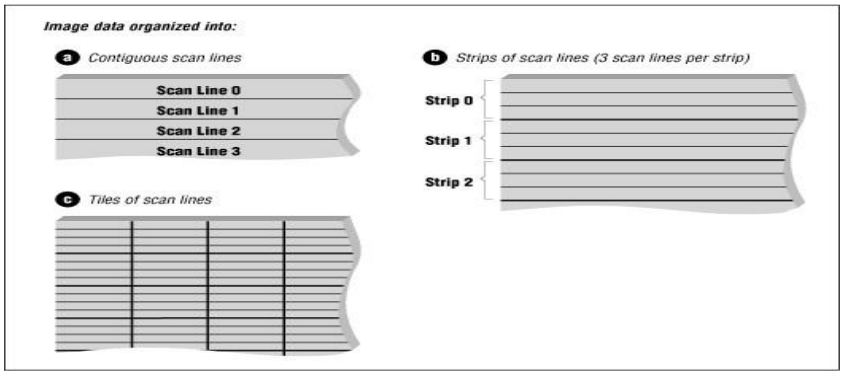


Figure 1-1: Organization of pixel data into scan lines (24-bit image)

Planar data

The second method of pixel value organization involves the separation of image data into two or more planes. Files in which the bitmap data is organized in this way are called planar files. We will use the term ***composite image*** to refer to an image with many colors (i.e., not monochrome, not gray-scale, and not one single color). Under this definition, most normal colored images that you are familiar with are composite images.

A composite image, then, can be represented by three blocks of bitmap data, each block containing just one of the component colors making up the image. Constructing each block is akin to the photographic process of making a separation--using filters to break up a color photograph into a set of component colors, usually three in number. The original photograph can be reconstructed by combining the three separations. Each block is composed of rows laid end to end, as in the simpler storage method explained above; in this case, more than one block is now needed to reconstruct the image. The blocks may be stored consecutively or may be physically separated from one another in the file.

Planar format data is usually a sign that the format designer had some particular display device in mind, one that constructed composite color pixels from components routed through hardware designed to handle one color at a time. For reasons of efficiency, planar format data is usually read one plane at a time in blocks, although an application may choose to laboriously assemble composite pixels by reading data from the appropriate spot in each plane sequentially.

As an example, a 24-bit image two rows by three columns wide might be represented in RGB format as six RGB pixel values:

```
(00, 01, 02) (03, 04, 05) (06, 07, 08)
```

(09, 10, 11) (12, 13, 14) (15, 16, 17)

But be written to the file in planar format as:

(00) (03)(06)

(09) (12)(15)

Red plane

(01) (04)(07)

(10) (13)(16)

Green plane

(02) (05)(08)

(11) (14)(17)

Blue plane

Notice that the exact same data is being written; it's just arranged differently. In the first case, an image consisting of six 24-bit pixels is stored as six 3-byte pixel values arranged in a single plane. In the second, planar, method, the same image is stored as 18 1-byte pixel values arranged in three planes, each plane corresponding to red, green, and blue information, respectively. Each method takes up exactly the same amount of space, 18 bytes, at least in this example.

It's pretty safe to say that most bitmap files are stored in non-planar format. Supporting planar hardware, then, usually means disassembling the pixel data and creating multiple color planes in memory, which are then presented to the planar rendering subroutine or the planar hardware.

Planar files may need to be assembled in a third buffer or, as mentioned above, laboriously set (by the routine servicing the output device) one pixel at a time.

Figure 1-2 illustrates the organization of pixel data into color planes.

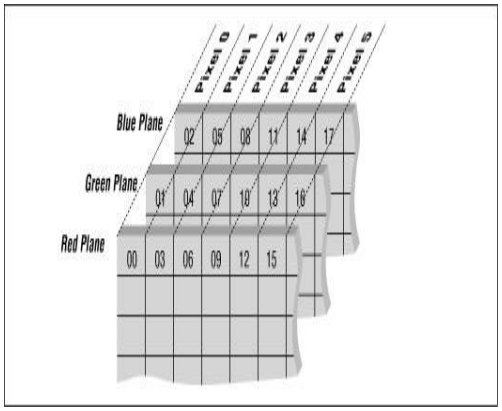


Figure 1-2: Organization of pixel data into color planes

DifferentApproaches to Bitmap Data Organization

Normally, we consider an image to be made up of a number of rows, each row a certain number of pixels wide. Pixel data representing the image can be stored in the file in three ways: as ***contiguous data***, as ***strips*** or as ***tiles.*** Figure 1-3 illustrates these three representations.

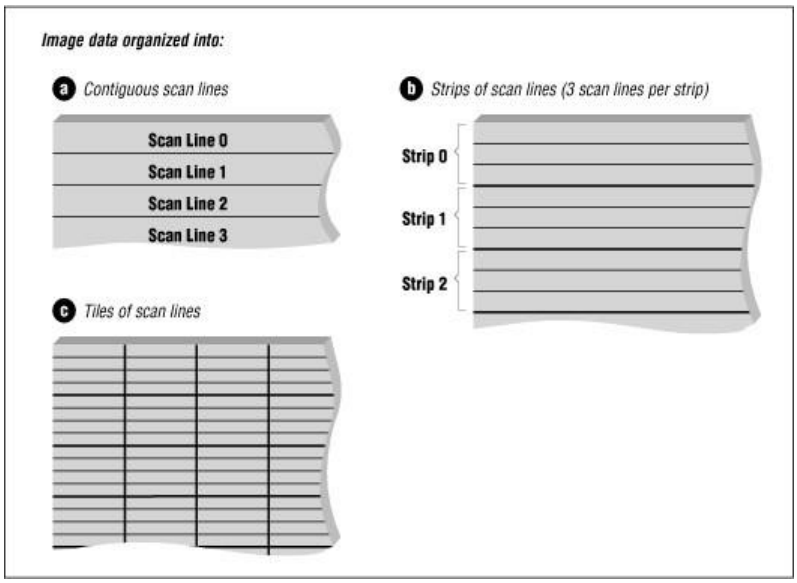


Figure 1-3: Examples of bitmap data organization (contiguous scan lines, strips, and tiles)

Contiguous data

The simplest method of row organization is where all of the image data is stored contiguously in the file, one row following the last. To retrieve the data you read the rows in file order, which delivers the rows in the order in which they were written. The data in this organizational scheme is stored in the file equivalent of a 2D array. You can index into the data in the file knowing the width of the row in pixels and the storage format and size of the pixel values. Data stored

contiguously in this manner can be read quickly, in large chunks, and assembled in memory quite easily.

Strips

In the second method of file organization, images are stored in strips, which also consist of rows stored contiguously. The total image, however, is represented by more than one strip, and the individual strips may be widely separated in the file. Strips divide the image into a number of segments, which are always just as wide as the original image.

Tiles

A third method of bitmap data organization is tiling. Tiles are similar to strips in that each is a delineation of a rectangular area of an image. However, unlike strips, which are always the width of the image, tiles can have any width at all, from a single pixel to the entire image. Thus, in one sense, a contiguous image is actually one large tile. In practice, however, tiles are arranged so that the pixel data corresponding to each is between 4Kb and 64Kb in size and is usually of a height and width divisible by 16. These limits help increase the efficiency with which the data can be buffered and decoded.

Dividing an image into tiles also allows different compression schemes to be applied to different parts of an image to achieve an optimal compression ratio. For example, one portion of an image (a very busy portion) could be divided into tiles that are compressed using JPEG, while another portion of the same image (a portion containing only one or two colors) could be stored as tiles that are run-length encoded. In this case, the tiles in the image would not all be the same uniform size; the smallest would be only a few pixels, and the largest would be hundreds or thousands of pixels on a side.

Tiling sometimes allows faster decoding and decompression of larger images than would be possible if the pixel data were organized as lines or strips. Because tiles can be individually encoded, file formats allowing the use of tiles will contain tile quantity, size, and offset information in the header specifications. Using this information, a reader that needs to display only the bottom-right corner of a very large image would have to read only the tiles for that area of the image; it would not have to read all of the image data that was stored before it.

Certain newer tile-oriented compression schemes, such as JPEG, naturally work better with file formats capable of supporting tiling. A good example of this is the incorporation of JPEG in later

versions of the TIFF file format. For more information about the use of tiles, see the article on the TIFF file format.

Palette

Many bitmap file formats contain a color palette.

Footer

The ***footer***, sometimes called the ***trailer***, is a data structure similar to a header and is often an addition to the original header, but appended to the end of a file. Footer is usually added when the file format is upgraded to accommodate new types of data and it is no longer convenient to add or change information in the header. It is mainly a result of a desire to maintain backward compatibility with previous versions of the format. An example of this is the TGA format, later revisions of which contain a footer that enables applications to identify the different versions of its format and to access special features available only in the later version of the format.

Because by definition it appears after the image data, which is usually of variable length, footer is never found at a fixed offset from the beginning of an image file unless the image data is always the same size. It is, however, usually located at a specified offset from the end of an image file. Like headers, footers are usually a fixed size. The offset value of the footer may also be present in the header information, provided there was reserved space or padding available in the header. Also like a header, footer may contain an identification field or magic number which can be used by a rendering application to differentiate it from other data structures in the file.

5.3 Vector Formats

Vector format files are particularly useful for storing line-based elements, such as lines and polygons, or those that can be decomposed into simple geometric objects, such as text. Vector files contain mathematical descriptions of image elements, rather than pixel values. A rendering application uses these mathematical descriptions of graphical shapes (e.g., lines, curves, and splines) to construct a final image.

In general, vector files are structurally simpler than most bitmap files and are typically organized as data streams.AutoCAD DXF and Microsoft SYLK are examples of commonly used vector formats.

Vector file formats have been around since computers were first used to display lines on an output device. CRTs, for example, were first used as computer-driven output devices in the 1950s. The first CRT displays were random scan devices similar to oscilloscopes, capable of producing images of mathematical and geometrical shapes. Vector display devices provided output sufficient for the needs of computer users for many years after their introduction, due to the limited range of tasks computers were called upon to perform.

At some point the need to store vector data arose, and portable storage media such as punch cards or paper tape were pressed into use. Prior to rendering time, an image was logically subdivided into its simplest elements. At rendering time, the image was produced and maintained by drawing each of its elements repeatedly in a specified order. At storage time, data was readily exported as a list of drawing operations, and mathematical descriptions of the image elements their size, shape, and position on the display screen--were written to the storage device in the order in which they were displayed.

Vector images are collections of device-independent mathematical descriptions of graphical shapes.

More so than their bitmap counterparts, various vector formats differ primarily because each was designed for a different purpose. While the conceptual differences between the designs of formats supporting 1-bit and 24-bit bitmap data may be small, the differences between vector formats used with CAD applications and those used for general data interchange can be formidable. Thus, it is difficult to generalize about vector formats in the same way we did when discussing bitmap formats.

On the other hand, most output devices are point-addressable, providing a grid of pixels which can be addressed individually, as if the surface of the device were graph paper made of discrete elements. This means that an application can always find a way to draw vector-format image elements on the device.

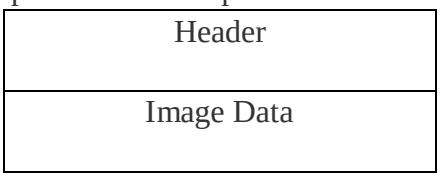
Organization of Vector files

Although vector files, like bitmap files, vary considerably in design, most contain the same basic structure: a header, a data section, and an end-of-file marker. Some structure is needed in the file to contain information global to the file and to correctly interpret the vector data at render time.

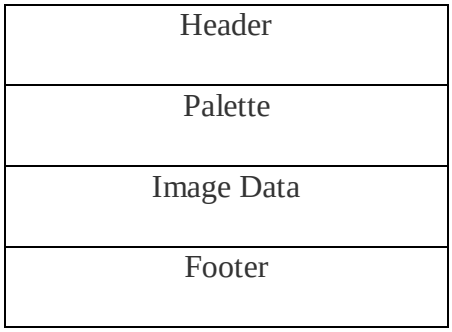
Although most vector files place this information in a header, some rely solely on a footer to perform the same task.

Vector files on the whole are structurally simpler than most bitmap files and tend to be organized as data streams. Most of the information content of the file is found in the image data.

The basic components of a simple vector file are the following:



If a file contains no image data, only a header will be present. If additional information is required that does not fit in the header, you may find footer appended to the file, and a palette may be included as well:



Header

The header contains information that is global to the vector file and must be read before the remaining information in the file can be interpreted. Such information can include a file format identification number, a version number, and color information.

Headers may also contain default attributes, which will apply to any vector data elements in the file lacking their own attributes. While this may afford some reduction in file size, it does so at the cost of introducing the need to cache the header information throughout the rendering operation.

Headers and footers found in vector-format files may not necessarily be a fixed size. For historical reasons mentioned above, it is not uncommon to find vector formats which use streams of variable-length records to store all data. If this is the case, then the file must be read sequentially and will normally fail to provide offset information that is necessary to allow the rendering application to subsample the image.

The type of information stored in the header is governed by the types of data stored in the file. Basic header information contains the height and width of the image, the position of the image on

the output device, and possibly the number of layers in the image. Thus, the size of the header may vary from file to file within the same format.

Vector Data

Vectors are line segments minimally defined as a starting point, a direction, and a length. They can, however, be much more complex and can include various sorts of lines, curves, and splines. Straight and curved lines can be used to define geometrical shapes, such as circles, rectangles, and polygons, which then can be used to create more complex shapes, such as spheres, cubes, and polyhedrons.

The bulk of all but tiny files consist of vector element data that contain information on the individual objects making up the image. The size of the data used to represent each object will depend upon the complexity of the object and how much thought went into reducing the file size when the format was designed.

Following the header is usually the image data. The data is composed of *elements*, which are smaller parts that comprise the overall image. Each element either inherits information or is explicitly associated with default information that specifies its size, shape, position relative to the overall image, color, and possibly other attribute information. An example of vector data in ASCII format containing three elements (a circle, a line, and a rectangle), might appear as:

```
;CIRCLE,40,100,100,BLUE;LINE,200,50,136,227,BLACK;RECT,80,65,25,78,RED;
```

Although this example is a simple one, it illustrates the basic problem of deciphering vector data, which is the existence of multiple levels of complexity. When deciphering a vector format, you not only must find the data, but you also must understand the formatting conventions and the definitions of the individual elements. This is hardly ever the case in bitmap formats; bitmap pixel data is all pretty much the same.

In this example, elements are separated by semicolons, and each is named, followed by numerical parameters and color information. Note, however, that consistency of syntax across image elements is never guaranteed. We could have just as easily defined the format in such a way as to make blocks of unnamed numbers signify lines by default:

```
;CIRCLE,40,100,100,BLUE;200,50,136,227,BLACK;RECT,80,65,25,78,RED;
```

and the default color black if unspecified: ;CIRCLE,40,100,100,BLUE;200,50,136,227;RECT,80,65,25,78,RED;

Many formats allow abbreviations:

```
;C,40,100,100,BL;200,50,136,227;R,80,65,25,78,R;
```

Notice that the R for RECT and R for RED are distinguished by context. You will find that many formats have opted to reduce data size at the expense of conceptual simplicity. You are free to consider this as evidence of flawed reasoning on the part of the format designer. The original reason for choosing ASCII was for ease of reading and parsing. Unfortunately, using ASCII may make the data too bulky. Solution: reduce the data size through implied rules and conventions and allow abbreviation (in the process making the format unreadable). The format designer would have been better off using a binary format in the first place.

Palettes and Color Information

Like bitmap files, vector files can contain palettes. Because the smallest objects defined in vector format files are the data elements, these are the smallest features for which color can be specified. Naturally, then, a rendering application must look up color definitions in the file palette before rendering the image. Our example above, to be correct, would thus need to include the color definitions, which take the form of a palette with associatedASCII names:

```
RED,255,0,0,
BLACK,0,0,0,
BLUE,0,0,255

;C,40,100,100,BL;200,50,136,227;R,80,65,25,78,R;
```

Some vector files allow the definition of enclosed areas, which are considered outlines of the actual vector data elements. Outlines may be drawn with variations in thickness or by using what are known as different ***pen styles***, which are typically combinations of dots and dashes and which may be familiar from technical and CAD drawings. Non-color items of information necessary for the reproduction of the image by the rendering application are called ***element attributes***.

Fills and color attributes

Enclosed elements may be designed to be filled with color by the rendering application. The filling is usually allowed to be colored independently from the element outline. Thus, each

element may have two or more colors associated with it, one for the element outline and one for the filling. ***Fill colors*** may be transparent, for instance, and some formats define what are called ***color attributes***. In addition to being filled with solid colors, enclosed vector elements may contain hatching or shading, which are in turn called ***fill attributes***. In some cases, fill and color attributes are lumped together, either conceptually in the format design, or physically in the file.

Formats that do not support fill patterns must simulate them by drawing parts of the pattern (lines, circles, dots, etc.) as separate elements. This not only introduces an uneven quality to the fill, but also dramatically increases the number of objects in the file and consequently the file size.

Gradient fills

An enclosed vector element may also be filled with more than one color. The easiest way is with what is called a ***gradient fill***, which appears as a smooth transition between two colors located in different parts of the element fill area. Gradient fills are typically stored as a starting color, an ending color, and the direction and type of the fill. A rendering application is then expected to construct the filled object, usually at the highest resolution possible. CGM is an example of a format that supports horizontal, vertical, and circular gradient fills. 1-4 illustrates a gradient fill.

Figure 1-4: Gradient fill

Footer

The footer may contain information that can be written to the file only after all the object data is written, such as the number of objects in the image. The footer in most vector formats, however, is simply used to mark the end of the object data.

Vector file size issues

Not counting palette and attribute information, the size of a vector file is directly proportional to the number of objects it contains. Contrast this with a complex bitmap file, which stays the same size no matter how complex the image described within. The only impact complexity has on bitmap files is on the degree of compression available to the file creator.

Vector files thus can vary greatly in size. A format can store images efficiently by using some form of shorthand notation to allow the compact definition of complex elements. A vector format rich in objects might be able to represent a single complex element using a Bezier curve, for instance. Another format not supporting Bezier curves would need to represent the same curve inefficiently, perhaps using a series of lines. Each line, in this case, would be a separate element, producing a file much larger than one supporting Bezier curves directly.

A format creator was probably addressing the problem of file size when he or she decided to support the creation and naming of complex elements. Great size savings come about when elements are repeated in the image; all that needs to be stored after the original element definition is a pointer to that definition, as well as attribute and position information specific to each individual repeated element.

Size savings may also come about from the way in which a format stores information. Different formats may support identical information in widely varying ways. For example, in the CGM format a hatch pattern is represented as a single object. In the PIC and Autodesk DXF formats, however, each line in the hatch pattern is stored as a separate element.

Because vector data is stored as numbers, it can be scaled, rotated, and otherwise manipulated easily and quickly, at least compared to bitmap data. Also, because scaling is so easy, vector files are not subject to image size limitations in the same way as bitmap files.

Vector formats normally do not support data compression as most bitmap formats do. Some formats, however, support an alternate encoding method that produces smaller data files, but contains the same information. CGM, for instance, normally stores vector information in a cleartext ASCII format that is human-readable, as does the example we presented earlier in this chapter. It also allows the storage of information in a binary format, however, which results in smaller files at the cost of readability and cross-platform portability. The DXF format also has a binary analog called DXB (Data eXchange Binary) which is not only smaller, but faster to load into its parent application (AutoCAD). It is, however, not portable to other applications.

Text in vector files

Vector formats that allow the storage of text strings do so in one of two ways. The simplest approach is to store the text as a literal ASCII string along with font, position, color, and attribute information. Although the text is provided in a compact form, this scheme requires the rendering application to have knowledge of the font to be used, which is always problematic. Because font names are for the most part vendor-controlled, it is sometimes difficult to even specify the font to be drawn. The CGM format addresses this problem through the use of an international registry of font names and associated descriptive data. Any rendering application supporting CGM must have access to this data, or it must use the font metric data supplied in the CGM file's header. Text, however, because it is stored in human-readable format, can be edited.

The second approach, and by far the most flexible, is to store the characters making up the text string as outlines constructed from a series of primitive vector data elements. Under this scheme each creator application must have access to font outline data; because it is stored like any other vector data, font outline data can be scaled at will, rotated, and otherwise manipulated. Until recently, access to outline data has been a problem, but vendors have realized the importance of support for outline fonts and are now routinely supplying this capability at the operating system level.

Because the graphics industry at large and the font industry have grown up in parallel, and only lately have begun to merge, there are naturally some incompatibilities between data storage models. Most fonts, for instance, are stored as a series of splines joined end-to-end, and a particular spline type may not be supported by the file format in use. In this case, the creator application may choose to convert the splines to arcs or lines and store these instead. This may or may not have an effect on the appearance of the text.

Creator applications may even incorporate vector or stroke fonts, which are usually primitive sets of character outlines with an angular or mechanical look, designed to be drawn with a minimum of fuss. Although many vendors have chosen to make their own, one widely available source are the Hershey fonts. Hershey data is available commercially, but is no longer considered adequate for general use.

The use of vector, stroke, or outline fonts usually increases the size of a file dramatically, but this may be offset by an increase in visual quality in the case of spline-based outline fonts. Although there are a number of older font formats still in use, spline-based outline font data in the

TrueType and Adobe Type 1 formats is easily available on all the major platforms. There is seldom any need to use stroke fonts now. Unfortunately, the reconstruction of characters from spline outline data is no trivial task, and the higher quality afforded by the ready availability of TrueType and Adobe Type 1 fonts comes at a price in rendering time and program development costs.

5.4 Metafile Formats

Metafiles can contain both bitmap and vector data in a single file. The simplest metafiles resemble vector format files; they provide a language or grammar that may be used to define vector data elements, but they may also store a bitmap representation of an image. Metafiles are frequently used to transport bitmap or vector data between hardware platforms, or to move image data between software platforms.WPG, Macintosh PICT, and CGM are examples of commonly used metafile formats. The following sectiondescribes the construction of metafiles in some detail.

When the term ***metafile*** first appeared, it was used in discussions of device- and machineindependent interchange formats. In the mid-1970s, the National Center for Atmospheric Research (NCAR), along with several other research institutions, reportedly used a format called metacode, which was device- and platform-independent to a certain degree. What is known for certain is that in 1979, the SIGGRAPH Graphics Standards and Planning Committee used the term, referring to a part of their published standards recommendations. These early attempts at defining device- and platform-independent formats mainly concerned themselves with vector data. Although work has continued along this line, we will refer to formats that can accommodate both bitmap and vector data as metafiles, because for all practical purposes the interchange formats in widespread use in the marketplace handle both types of data.

Although metafile formats may be used to store only bitmap or only vector information, it is more likely that they will contain both types of data. From a programmer's point of view, bitmap and vector data are two very different problems. Because of this, supporting both bitmap and vector data types adds to the complexity of a format. Thus, programmers find themselves avoiding the use of metafile formats unless the added complexity is warranted--either because they need to support multiple data types or for external reasons.

The simplest metafiles resemble vector format files. Historically, limitations of vector formats were exceeded when the data that needed to be stored became complex and diverse. Vector

formats were extended conceptually, allowing the definition of vector data elements in terms of a language or grammar, and also by allowing the storage of bitmap data. In a certain sense, the resulting formats went beyond the capabilities of both bitmap and vector formats--hence the term *metafile*.

Metafiles are widely used to transport bitmap or vector data between hardware platforms. The character-oriented nature of ASCII metafiles, in particular, eliminates problems due to byte ordering. It also eliminates problems encountered when transferring files across networks where the eighth bit of each byte is stripped off, which can leave binary files damaged beyond repair. Also, because a metafile supports bitmap and vector data, an application designer can kill two birds with one stone by providing support for one metafile rather than two separate bitmap and vector formats.

Metafiles are also used to transfer image data between software platforms. A creator application, for instance, can save an image in both bitmap and vector form in a metafile. This file may then be read by any bitmap-capable or vector-capable application supporting the particular metafile format. Many desktop publishing programs, for instance, can manipulate and print vector data, but are unable to display that same data on the screen. To accommodate this limitation, a bitmap representation of the image is often included along with the vector data in a metafile. The application can read the bitmap representation of the image from the metafile, which serves as a reduced-quality visual representation of the image that will eventually appear on the printed page. When the page is actually printed, however, the vector data from the metafile isused to produce the image on the printer. Display PostScript files are an example of this type of arrangement.

Organization of metafiles

Metafiles vary so widely in format that it is pointless to attempt to construct a hierarchical explanation of their general construction. Most metafiles contain some sort of header, followed by one or more sections of image data. Some metafiles contain nothing but bitmap data, and still others contain no data at all, opting instead for cryptic drawing instructions, or numerical data similar to that found in vector files.

---

5. Scene Formats

Scene format files (sometimes called ***scene description*** files) are designed to store a condensed representation of an image or scene, which is used by a program to reconstruct the actual image.

Difference between a vector format file and a scene format file,

Just that vector files contain descriptions of portions of the image, and scene files contain instructions that the rendering program uses to construct the image. In practice it's sometimes hard to decide whether a particular format is scene or vector; it's more a matter of degree than anything absolute.

6. Animation Formats

Animation formats have been around for some time. The basic idea is that of the flip-books you played with as a kid; with those books, you rapidly displayed one image superimposed over another to make it appear as if the objects in the image are moving. Very primitive animation formats store entire images that are displayed in sequence, usually in a loop. Slightly more advanced formats store only a single image but multiple color maps for the image. By loading in a new color map, the colors in the image change and the objects appear to move. Advanced animation formats store only the differences between two adjacent images (called ***frames***) and update only the pixels that have actually changed as each frame is displayed. A display rate of 10-15 frames per second is typical for cartoon-like animations. Video animations usually require a display rate of 20 frames per second or better to produce a smoother motion.

TDDD and TTDDD are examples of animation formats.

5.7 Multimedia Formats

~~Multimedia formats are relatively new but are becoming more and more important. They are~~ designed to allow the storage of data of different types in the same file. Multimedia formats usually allow the inclusion of graphics, audio, and video information. Microsoft's RIFF, Apple's QuickTime, MPEG, and Autodesk's FLI are well-known examples, and others are likely to emerge in the near future. The following section, describes various issues concerning multimedia formats.

Multimedia data and information must be stored in a disk file using formats similar to image file formats. Multimedia formats, however, are much more complex than most other file formats because of the wide variety of data they must store. Such data includes text, image data, audio

and video data, computer animations, and other forms of binary data, such as Musical Instrument Digital Interface (MIDI), control information, and graphical fonts. Typical multimedia formats do not define new methods for storing these types of data. Instead, they offer the ability to store data in one or more existing data formats that are already in general use.

For example, a multimedia format may allow text to be stored as PostScript or Rich Text Format (RTF) data rather than in conventional ASCII plain-text format. Still-image bitmap data may be stored as BMP or TIFF files rather than as raw bitmaps. Similarly, audio, video, and animation data can be stored using industry-recognized formats specified as being supported by that multimedia file format.

Multimedia formats are also optimized for the types of data they store and the format of the medium on which they are stored. Multimedia information is commonly stored on CD-ROM. Unlike conventional disk files, CD-ROMs are limited in the amount of information they can store. A multimedia format must therefore make the best use of available data storage techniques to efficiently store data on the CD-ROM medium.

There are many types of CD-ROM devices and standards that may be used by multimedia applications. If you are interested in multimedia, you should become familiar with them.

The original Compact Disc first introduced in early 1980s was used for storing only audio information using the CD-DA (Compact Disc-Digital Audio) standard produced by Phillips and Sony. CD-DA (also called the Red Book) is an optical data storage format that allows the storage of up to 74 minutes of audio (764 megabytes of data) on a conventional CD-ROM.

The CD-DA standard evolved into the CD-XA (Compact Disc-Extended Architecture) standard, or what we call the CD-ROM (Compact Disc-Read Only Memory). CD-XA (also called the Yellow Book) allows the storage of both digital audio and data on a CD-ROM. Audio may be combined with data, such as text, graphics, and video, so that it may all be read at the same time. An ISO 9660 file system may also be encoded on a CD-ROM, allowing its files to be read by a wide variety of different computer system platforms.

The CD-I (Compact Disc-Interactive) standard defines the storage of interactive multimedia data. CD-I (also called the Green Book) describes a computer system with audio and video playback capabilities designed specifically for the consumer market. CD-I units allow the integration of fully interactive multimedia applications into home computer systems.

A still-evolving standard is CD-R (Compact Disc-Recordable or Compact Disc-Write Once), which specifies a CD-ROM that may be written to by a personal desktop computer and read by any CD-ROM player.

The following sections describe various types of data that you might find, in addition to static graphics data, in multimedia files.

Animation

Somewhere between the motionless world of still images and the real-time world of video images lies the flip-book world of computer animation. All of the animated sequences seen in educational programs, motion CAD renderings, and computer games are computer-animated (and in many cases, computer-generated) animation sequences.

Traditional cartoon animation is little more than a series of artwork cells, each containing a slight positional variation of the animated subjects. When a large number of these cells is displayed in sequence and at a fast rate, the animated figures appear to the human eye to move.

A computer-animated sequence works in exactly the same manner. A series of images is created of a subject; each image contains a slightly different perspective on the animated subject. When these images are displayed (played back) in the proper sequence and at the proper speed (frame rate), the subject appears to move.

Computerized animation is actually a combination of both still and motion imaging. Each frame, or cell, of an animation is a still image that requires compression and storage. An animation file, however, must store the data for hundreds or thousands of animation frames and must also provide the information necessary to play back the frames using the proper display mode and frame rate.

Animation file formats are only capable of storing still images and not actual video information. It is possible, however, for most multimedia formats to contain animation information, because animation is actually a much easier type of data than video to store.

The image-compression schemes used in animation files are also usually much simpler than most of those used in video compression. Most animation files use a delta compression scheme, which is a form of Run-Length Encoding that stores and compresses only the information that is different between two images (rather than compressing each image frame entirely). RLE is relatively easy to decompress on the fly. Storing animations using a multimedia format also

produces the benefit of adding sound to the animation. Most animation formats cannot store sound directly in their files and must rely on storing the sound in a separate disk file which is read by the application that is playing back the animation.

Animations are not only for entertaining kids and adults. Animated sequences are used by CAD programmers to rotate 3D objects so they can be observed from different perspectives; mathematical data collected by an aircraft or satellite may be rendered into an animated fly-by sequence. Movie special effects benefit greatly by computer animation.

Digital Video

Video data normally occurs as continuous, analog signals. In order for a computer to process this video data, we must convert the analog signals to a non-continuous, digital format. In a digital format, the video data can be stored as a series of bits on a hard disk or in computer memory.

The process of converting a video signal to a digital bit-stream is called analog-to-digital conversion (A/D conversion), or digitizing.A/D conversion occurs in two steps:

1. Sampling captures data from the video stream.
2. Quantizing converts each captured sample into a digital format.

Each sample captured from the video stream is typically stored as a 16-bit integer. The rate at which samples are collected is called the ***sampling rate***. The sampling rate is measured in the number of samples captured per second (samples/second). For digital video, it is necessary to capture millions of samples per second.

Quantizing converts the level of a video signal sample into a discrete, binary value. This value approximates the level of the original video signal sample. The value is selected by comparing the video sample to a series of predefined threshold values. The value of the threshold closest to the amplitude of the sampled signal is used as the digital value.

A video signal contains several different components which are mixed together in the same signal. This type of signal is called a composite video signal and is not really useful in highquality computer video. Therefore, a standard composite video signal is usually separated into its basic components before it is digitized.

The composite video signal format defined by the NTSC (National Television Standards Committee) color television system is used in the United States. The PAL (Phase Alternation

Line) and SECAM (Sequential Coleur Avec Memoire) color television systems are used in Europe and are not compatible with NTSC. Most computer video equipment supports one or more of these system standards.

The components of a composite video signal are normally decoded into three separate signals representing the three channels of a color space model, such as RGB, YUV, or YIQ. Although the RGB model is quite commonly used in still imaging, the YUV, YIQ, or YCbCr models are more often used in motion-video imaging. TV practice uses YUV or similar color models because the U and V channels can be down sampled to reduce data volume without materially degrading image quality.

Once the video signal is converted to a digital format, the resulting values can be represented on a display device as pixels. Each pixel is a spot of color on the video display, and the pixels are arranged in rows and columns just as in a bitmap. Unlike a static bitmap, however, the pixels in a video image are constantly being updated for changes in intensity and color. This updating is called scanning, and it occurs 60 times per second in NTSC video signals (50 times per second for PAL and SECAM).

A video sequence is displayed as a series of frames. Each frame is a snapshot of a moment in time of the motion-video data, and is very similar to a still image. When the frames are played back in sequence on a display device, a rendering of the original video data is created. In realtime video the playback rate is 30 frames per second. This is the minimum rate necessary for the human eye to successfully blend each video frame together into a continuous, smoothly moving image.

A single frame of video data can be quite large in size. A video frame with a resolution of 512 x 482 will contain 246,784 pixels. If each pixel contains 24 bits of color information, the frame will require 740,352 bytes of memory or disk space to store. Assuming there are 30 frames per second for real-time video, a 10-second video sequence would be more than 222 megabytes in size! It is clear there can be no computer video without at least one efficient method of video data compression.

There are many encoding methods available that will compress video data. The majority of these methods involve the use of a transform coding scheme, usually employing a Fourier or Discrete Cosine Transform (DCT). These transforms physically reduce the size of the video data by selectively throwing away unneeded parts of the digitized information. Transform compression

schemes usually discard 10 percent to 25 percent or more of the original video data, depending largely on the content of the video data and upon what image quality are considered acceptable.

Usually a transform is performed on an individual video frame. The transform itself does not produce compressed data. It discards only data not used by the human eye. The transformed data, called coefficients, must have compression applied to reduce the size of the data even further. Each frame of data may be compressed using a Huffman or arithmetic encoding algorithm, or even a more complex compression scheme such as JPEG. This type of intra-frame encoding usually results in compression ratios between 20:1 to 40:1 depending on the data in the frame. However, even higher compression ratios may result if, rather than looking at single frames as if they were still images, we look at multiple frames as temporal images.

DigitalAudio

All multimedia file formats are capable, by definition, of storing sound information. Sound data, like graphics and video data, has its own special requirements when it is being read, written, interpreted, and compressed. Before looking at how sound is stored in a multimedia format we must look at how sound itself is stored as digital data.

All of the sounds that we hear occur in the form of analog signals. An analog audio recording system, such as a conventional tape recorder, captures the entire sound wave form and stores it in analog format on a medium such as magnetic tape.

Because computers are now digital devices it is necessary to store sound information in a digitized format that computers can readily use. A digital audio recording system does not record the entire wave form as analog systems do (the exception being Digital Audio Tape [DAT] systems). Instead, a digital recorder captures a wave form at specific intervals, called the sampling rate. Each captured wave-form snapshot is converted to a binary integer value and is then stored on magnetic tape or disk.

Storing audio as digital samples is known as Pulse Code Modulation (PCM). PCM is a simple quantizing or digitizing (audio to digital conversion) algorithm, which linearly converts all analog signals to digital samples. This process is commonly used on all audio CD-ROMs.

Differential Pulse Code Modulation (DPCM) is an audio encoding scheme that quantizes the difference between samples rather than the samples themselves. Because the differences are easily represented by values smaller than those of the samples themselves, fewer bits may be

used to encode the same sound (for example, the difference between two 16-bit samples may only be four bits in size). For this reason, DPCM is also considered an audio compression scheme.

One other audio compression scheme, which uses difference quantization, is Adaptive Differential Pulse Code Modulation (ADPCM). DPCM is a non-adaptive algorithm. That is, it does not change the way it encodes data based on the content of the data. DPCM uses the sample number of bits to represent every signal level. ADPCM, however, is an adaptive algorithm and changes its encoding scheme based on the data it is encoding. ADPCM specifically adapts by using fewer bits to represent lower-level signals than it does to represent higher-level signals. Many of the most commonly used audio compression schemes are based onADPCM.

Digital audio data is simply a binary representation of a sound. This data can be written to a binary file using an audio file format for permanent storage much in the same way bitmap data is preserved in an image file format. The data can be read by a software application, can be sent as data to a hardware device, and can even be stored as a CD-ROM.

The quality of an audio sample is determined by comparing it to the original sound from which it was sampled. The more identical the sample is to the original sound, the higher the quality of the sample. This is similar to comparing an image to the original document or photograph from which it was scanned.

The quality of audio data is determined by three parameters:

- Sample resolution
- Sampling rate
- Number of audio channels sampled

The ***sample resolution*** is determined by the number of bits per sample. The larger the sampling size, the higher the quality of the sample. Just as the apparent quality (resolution) of an image is reduced by storing fewer bits of data per pixel, so is the quality of a digital audio recording reduced by storing fewer bits per sample. Typical sampling sizes are eight bits and 16 bits.

The ***sampling rate*** is the number of times per second the analog wave form was read to collect data. The higher the sampling rate, the greater the quality of the audio. A high sampling rate collects more data per second than a lower sampling rate, therefore requiring more memory and

disk space to store. Common sampling rates are 44.100 kHz (higher quality), 22.254 kHz (medium quality), and 11.025 kHz (lower quality). Sampling rates are usually measured in the signal processing terms hertz (Hz) or kilohertz (kHz), but the term samples per second (samples/second) is more appropriate for this type of measurement.

A sound source may be sampled using one channel (monaural sampling) or two channels (stereo sampling). Two-channel sampling provides greater quality than mono sampling and, as you might have guessed, produces twice as much data by doubling the number of samples captured.

Sampling one channel for one second at 11,000 samples/second produces 11,000 samples.
Sampling two channels at the same rate, however, produces 22,000 samples/second.

One solution to the massive storage requirements of high-quality audio data is data compression. For example, the CD-DA (Compact Disc-Digital Audio) standard performs mono or stereo sampling using a sample resolution of 16 bits and a sampling rate of 44.1 samples/second, making it a very high-quality format for both music and language applications. Storing five minutes of CD-DA information requires approximately 25 megabytes of disk space--only half the amount of space that would be required if the audio data were uncompressed.

Audio compression algorithms, like image compression algorithms, can be categorized as lossy and lossless. Lossless compression methods do not discard any data. The decompression step produces exactly the same data as was read by the compression step. A simple form of lossless audio compression is to Huffman-encode the differences between each successive 8-bit sample. Huffman encoding is a lossless compression algorithm and, therefore the audio data is preserved in its entirety.

Lossy compression schemes discard data based on the perceptions of the psychoacoustic system of the human brain. Parts of sounds that the ear cannot hear, or the brain does not care about, can be discarded as useless data.

An algorithm must be careful when discarding audio data. The ear is very sensitive to changes in sound. The eye is very forgiving about dropping a video frame here or reducing the number of colors there. The ear, however, notices even slight changes in sounds, especially when specifically trained to recognize audial infidelities and discrepancies. However, the higher the quality of an audio sample, the more data will be required to store it. As with lossy image

compression schemes, at times you'll need to make a subjective decision between quality and data size.

Audio

There is currently no "audio file interchange format" that is widely used in the computer-audio industry. Such a format would allow a wide variety of audio data to be easily written, read, and transported between different hardware platforms and operating systems.

Most existing audio file formats, however, are very machine-specific and do not lend themselves to interchange very well. Several multimedia formats are capable of encapsulating a wide variety of audio formats, but do not describe any new audio data format in themselves.

Many audio file formats have headers just as image files do. Their header information includes parameters particular to audio data, including sample rate, number of channels, sample resolution, type of compression, and so on. An identification field ("magic" number) is also included in several audio file format headers.

Several formats contain only raw audio data and no file header. Any parameters these formats use are fixed in value and therefore would be redundant to store in a file header. Stream-oriented formats contain packets (chunks) of information embedded at strategic points within the raw audio data itself. Such formats are very platform-dependent and would require an audio file format reader or converter to have prior knowledge of just what these parameter values are. Most audio file formats may be identified by their file types or extensions.

MIDI Standard

Musical Instrument Digital Interface (MIDI) is an industry standard for representing sound in a binary format. MIDI is not an audio format, however. It does not store actual digitally sampled sounds. Instead, MIDI stores a description of sounds, in much the same way that a vector image format stores a description of an image and not image data itself.

Sound in MIDI data is stored as a series of control messages. Each message describes a sound event using terms such as pitch, duration, and volume. When these control messages are sent to a MIDI-compatible device (the MIDI standard also defines the interconnecting hardware used by MIDI devices and the communications protocol used to interchange the control information) the information in the message is interpreted and reproduced by the device.

MIDI data may be compressed, just like any other binary data, and does not require special compression algorithms in the way that audio data does.

5.8 Hybrid Formats

Currently, there is a good deal of research being conducted on the integration of unstructured text and bitmap data ("hybrid text") and the integration of record-based information and bitmap data ("hybrid database"). As this work bears fruit, we expect that hybrid formats capable of efficiently storing graphics data will emerge and will steadily become more important.

Hypertext and Hypermedia Formats

Hypertext is a strategy for allowing nonlinear access to information. In contrast, most books are linear, having a beginning, an end, and a definite pattern of progression through the text. Hypertext, however, enables documents to be constructed with one or more beginnings, with one, none, or multiple ends, and with many hypertext links that allow users to jump to any available place in the document they wish to go.

Hypertext languages are not graphics file formats, like the GIF or DXF formats. Instead, they are programming languages, like PostScript or C. As such, they are specifically designed for serial data stream transmission. That is, you can start decoding a stream of hypertext information as you receive the data. You need not wait for the entire hypertext document to be downloaded before viewing it.

The term ***hypermedia*** refers to the marriage of hypertext and multimedia. Modern hypertext languages and network protocols support a wide variety of media, including text and fonts, still and animated graphics, audio, video, and 3D data. Hypertext allows the creation of a structure that enables multimedia data to be organized, displayed, and interactively navigated through by a computer user.

Hypertext and hypermedia systems, such as the World Wide Web, contain millions of information resources stored in the form of GIF, JPEG, PostScript, MPEG, and AVI files. Many other formats are used as well.

5.9 Summary

Bitmap formats are used to store bitmap data. Files of this type are particularly well-suited for the storage of real-world images such as photographs and video images. Bitmap files, sometimes called ***raster files***, essentially contain an exact pixel-by-pixel map of an image.

Vector format files are particularly useful for storing line-based elements, such as lines and polygons, or those that can be decomposed into simple geometric objects, such as text.

Metafiles can contain both bitmap and vector data in a single file. The simplest metafiles resemble vector format files; they provide a language or grammar that may be used to define vector data elements, but they may also store a bitmap representation of an image. Metafiles are frequently used to transport bitmap or vector data between hardware platforms, or to move image data between software platforms.

Multimedia formats are relatively new but are becoming more and more important. They are designed to allow the storage of data of different types in the same file. Multimedia formats usually allow the inclusion of graphics, audio, and video information.

Currently, there is a good deal of research being conducted on the integration of unstructured text and bitmap data ("hybrid text") and the integration of record-based information and bitmap data ("hybrid database"). As this work bears fruit, we expect that hybrid formats capable of efficiently storing graphics data will emerge and will steadily become more important.

10. Keywords

Bitmap files, Metafile formats, Scene formats, Vector formats, Multimedia formats, Animation formats.

11. Exercises

1. Explain briefly the structure of Bitmap file format.
2. Discuss briefly about Vector file format.
3. Discuss different types of multimedia formats.
4. Explain briefly about scene formats.
5. Explain briefly Hypermedia file formats.]

12. References

1. https://W3.vanderdilt.edu

2. https://library.rice.edu

UNIT-6 Advanced File Formats : 3D File formats

Structure

1. Objectives
2. 3D Formats
3. VRML
4. Audio formats
5. Font formats
6. Different types of Multimedia tools for editing
7. Motion JPEG
8. Summary
9. Keywords
10. Questions
11. References

1. Learning Objectives

- Describe different types of file formats such as,3D, virtual modeling reality language (VRML), audio, font, and page description language (PDL).
- Discuss different type of video editing technologies such as, MPEG, MJPEG.

2. 3D Formats

Three-dimensional data files store descriptions of the shape and color of 3D models of imaginary and real-world objects. 3D models are typically constructed of polygons and smooth surfaces, combined with descriptions of related elements, such as color, texture, reflections, and so on, that a rendering application can use to reconstruct the object. Models are placed in scenes with lights and cameras, so objects in 3D files are often called ***scene element****s*.

Rendering applications that can use 3D data are generally modeling and animation programs, such as NewTek'sLightwave and Autodesk's 3D Studio. They provide the ability to adjust the

appearance of the rendered image through changes and additions to the lighting, textures applied to scene elements, and the relative positions of scene elements. In addition, they allow the user to animate, or assign motions to, scene elements. The application then creates a series of bitmap files, or frames that taken in sequence can be assembled into a movie.

It's important to understand that vector data historically has been 2D in nature. That is, the creator application with which the data originated made no attempt to simulate 3D display through the application of perspective. Examples of vector data include CAD drawings and most clip art designed to be used in desktop publishing applications. There is a certain amount of confusion in the market about what constitutes 3D rendering. This is complicated by the fact that 3D data is now supported by a number of formats that previously stored only 2D vector data. An example of this is Autodesk's DXF format. Formats like DXF are sometimes referred to as *extended vector formats*.

6.2 VRML (Virtual Reality modeling Language)

VRML, which stands for *Virtual Reality modeling Language,* was conceived at the first international conference of the World Wide Web. Mark Parisi, and David Ragget outlined the structure of VRML at the conference and specified that it would be a platform-independent language that would be viewed on the Internet. The objective of VRML was to have the capability to put colored objects into a 3D environment.

VRML is an interpreted today, which can be seen as disadvantage, because it runs slowly on many computers today. However, it has been influential, because it was the first method available for displaying a 3D world on the World Wide web.

Strictly speaking, VRML is not a ―tool‖, like premiere or Director. In fact, the only piece of software needed to create VRML content in a text editor. Nonetheless, VRML is a tool used to create 3D environments on the web, much like Flash is a tool used to create interactive movies.

History

VRML 1.0 was created in May 1995, with a revision for clarification called VRML 1.0C in January 1996. VRML is based on a subset of the file invertor format created by Silicon Graphics user-defined polygons. Materials and textures can be specified for objects to make the objects more realistic.

The last major revision of VRML was VRML 2.0. This revision added the ability to create an interactive world. VRML 2.0, also called ―Moving Worlds‖, allows for animation and sound in an interactive virtual world. New objects were added to make the creation of virtual worlds easier. Java and javascript have been included in VRML, to allow for interactive objects and userdefined actions. VRML 2.0 was a major change from VRML 1.0, and the two versions are not compatible. However, utilities are available to convert VRML 1.0 to VRML 2.0.

VRML 2.0 was submitted for standardization to the International Organization for Standardization (ISO), and as a result, VRML97 was specified. VRML97 is virtually identical to VRML 2.0 - only minor documentation changes and clarifications were added. VRML97 is an ISO/IEC standard.

VRML Shapes

VRML is made up of nodes put into a hierarchy that describe a scene of one or more objects. VRML contains basic geometric shapes that can be combined to create more complex objects. The **shape** node is a generic node for all objects in VRML. The **Box, Cylinder, Cones, and Sphere** are geometry nodes that place objects in the virtual world.

VRML allows for the definition of complex shapes that include **IndexdFaceSet** and **Extrusion.** An **IndexdFaceSet** is a set of faces make up an object. This allows for the creation of complex shapes, since an arbitrary number of faces is allowed. An **Extrusion** is a 2D cross-section extruded along a spine and is useful in creating a simple curved surface, such as flower petal.

An object‘s shape, size, color, and reflective properties can be specified in VRML. The **Appearance** node controls the way a shape looks and can contain a **Material** node and texture nodes.

A **Material** node specifies an object‘s surface properties. It can control what color object is by specifying the red, green, and blue values of the object. The specular and emissive colors can be specified similarly. Other attributes, such as how much the object reflects direct and indirect light, can also be controlled. Objects in VRML can be transparent or partially transparent. This is also included in the **Material** node.

Three kinds of texture nodes can be used to map textures onto any object. The most common one is the **ImageTexture,** which can an external JPEG or PNG image file and map it onto thr shape.

The way the image is textured can be specified – that is, the way the image should be tiled onto the object is editable.

A **MovieTexture**node allows mapping an MPEG movie onto an object; the starting and stopping time can also be specified.

The final texture-mapping node is called a **PixelTexture,** which simply means creating an image to use with **ImageTexture** VRML. Although it is more inefficient than a **ImageTexture**node, it is still useful for simple textures.

Text can be put into a VRML world using the **Text** node. You can specify the text to be included, as well as the font, alignment, and size. By default, the text faces in the positive Y direction, or
—up‖.

All shapes and text start in the middle of the VRML world. To arrange the shapes, **Transform** nodes must be wrapped around the shape nodes. The **Transform** node can contain *Translation, Scale an*d *Rotation* nodes. Translation simply moves the object a specific distance from its current location, which is by default the center of the world. Scale increases or decreases the size of the object, while Rotation rotates the object around the center.

VRMLWorld

A virtual world needs more than just shapes to be realistic; it needs cameras to view the objects, as well as backgrounds and lighting. The default camera is aligned with the negative z-axis, a few meters from the center of the scene. Using **Viewpoint** nodes, the default camera position can be changed and other cameras added. The viewpoint can be specified with the **position** node can be rotated from the default view with the **Orientation** node. The camera‘s angle for its field of view can be changed from its default 0.78radians with the **filedView** node. Changing the field of view can create a telephone effect.

Three types of lighting can be used in a VRML world. A **DirectionalLight** node shines a light across the whole world in a certain direction, similar to the light from the sun it is from one direction and affects all objects in the scene. A **PointLight** shines a light in all directions from a certain point in space. A **SpotLight** shines a light in a certain direction from a point. Proper

lightning important in adding realism to a world. Many parameters, such as the color and strength of the light, can be specified for every type of light.

The background of the VRML world can also be specified using the **Background** node. The background color, black by default, as well as the sky color can be changed. A **Panorama** node can map a texture to the sides of the world. A panorama is mapped onto a large cube surrounding the VRML world. If a panorama is used, the user can never approach the texture, because the panorama is centered on the user. It is also possible to add fog in VRML using the **Fog** node, where the color and density of the fog can be specified. Fog can increase the frame rate of a world, since objects hidden by the fog are not rendered.

Animation and interactions

An advantage of VRML97 over the original VRML 1.0 is that the VRML world can be interactive. The only method of animation in VRML is tweening, which can be done by slowly changing an object specified in an interpolator node. This node will modify an object over time, based on the type of interpolator.

There are six interpolators: *color, coordinate, normal, orientation, position, and scalar.* All interpolators have two nodes that must be specified: the *key* and *KeyValue.* The *key* consists of a list of a two or more numbers, starting with 0 and ending with 1. Each key element must be complemented with a *KeyValue* element. The key defines how far along the animation is, and the *keyValue* defines what values should change. For example, a key element of 0.5 and its matching keyValue define what the object should look at the middle of the animation.

A TimeSensor node times an animation, so that the interpolator knows what stage the object should be in. A TimeSensor has no physical form in the VRML world and just keeps time. To notify an interpolator of a time change, a ROUTE is needed to connect two nodes. One is needed between the TimeSensor and the interpolator and another between the interpolator and the object to be animated. Most animation can be accomplished this way. Chaining ROUTE commands so that one event triggers many others can accomplish complex animation.
Two categories of sensors can be used in VRML to obtain from input from a user. The first is *environment* sensors. There are three kinds of environment sensor nodes.

3. Audio Formats

An **audio file format** is a file format for storing digital audio data on a computer system. This data can be stored uncompressed, or compressed to reduce the file size. It can be a raw bitstream, but it is usually a container format or an audio data format with defined storage layer.

Format Types

It is important to distinguish between a file format and an audio codec. A codec performs the encoding and decoding of the raw audio data while the data itself is stored in a file with a specific audio file format. Although most audio file formats support only one type of audio data (created with an audio coder), a multimedia container format (as Matroska or AVI) may support multiple types of audio and video data.

There are three major groups of audio file formats:

- Uncompressed audio formats, Such as WAV,AIFF, AU.
- Formats with lossless compression, such as FLAC, WavPack (filename extension WV), TTA, ATRAC Advanced lossless, Apple Lossless (filename extension m4a), MPEG-4 ALS, MPEG-4 DST, Windows Media Audio Lossless (WMA Lossless), and Shorten (SHN).
- Formats with lossy compression, such as MP3, VorbisMusepack, AAC, ATRAC, and Windows Audio Llossy (WMAlossy).

Uncompressed audio format

Here is one major uncompressed audio format, PCM, which is usually stored in a .wav file on Windows or in a .aiff file on Mac OS. The AIFF format is based on the Interchange File Format (IFF). The WAV format is based on the Resource Interchange File Format (RIFF), which is similar to IFF. WAV and AIFF are flexible file formats designed to store more or less any combination of sampling rates or bitrates. This makes them suitable file formats for storing and archiving an original recording.

BWF (Broadcast Wave Format) is a standard audio format created by the European Broadcasting Union as a successor to WAV. BWF allows metadata to be stored in the file.

Lossless compressed audio format

A lossless compressed format stores data in less space by eliminating unnecessary data.

Uncompressed audio formats encode both sound and silence with the same number of bits per unit of time. Encoding an uncompressed minute of absolute silence produces a file of the same size as encoding an uncompressed minute of music. In a lossless compressed format, however, the music would occupy a smaller file than an uncompressed format and the silence would take up almost no space at all.

Lossless compression formats enable the original uncompressed data to be recreated exactly. They include the commonFLAC, WavPack, Monkey's Audio, ALAC (Apple Lossless). They provide a compression ratio of about 2:1 (i.e. their files take up half the space of the originals). Development in lossless compression formats aims to reduce processing time while maintaining a good compression ratio.

Lossy compressed audio format

Lossy compression enables even greater reductions in file size by removing some of the data. Lossy compression typically achieves far greater compression than lossless compression by simplifying the complexities of the data.This of course results in a reduction in audio quality, but a variety of techniques are used, mainly by exploiting psychoacoustics, to remove the data that has least effect on perceived quality. The loss in data (and thus quality) may not always be easily perceived through casual listening. The popular MP3 format is probably the best-known example, but the AAC format found on the iTunes Music Store is also common. Most formats offer a range of degrees of compression, generally measured in bit rate. The lower the rate, the smaller the file and the more significant the quality loss.

6.4 Font Formats

Another class of formats not covered in this book are font files. Font files contain the descriptions of sets of alphanumeric characters and symbols in a compact, easy-to-access format. They are generally designed to facilitate random access of the data associated with individual characters. In this sense, they are databases of character or symbol information, and for this reason font files are sometimes used to store graphics data that is not alphanumeric or symbolic in nature. Font files may or may not have a global header, and some files support sub-headers for each character. In any case, it is necessary to know the start of the actual character data, the size of each character's data, and the order in which the characters are stored in order to retrieve individual characters without having to read and analyze the entire file. Character data in the file

may be indexed alphanumerically, by ASCII code, or by some other scheme. Some font files support arbitrary additions and editing, and thus have an index somewhere in the file to help you find the character data.

Some font files support compression, and many support encryption of the character data. The creation of character sets by hand has always been a difficult and time-consuming process, and typically a font designer spent a year or more on a single character set. Consequently, companies that market fonts (called foundries for reasons dating back to the origins of printing using mechanical type) often seek to protect their investments through legal means or through encryption. In the United States, for instance, the names of fonts are considered proprietary, but the outlines described by the character data are not. It is not uncommon to see pirated data embedded in font files under names different from the original.

Historically there have been three main types of font files: bitmap, stroke, and spline-based outlines, described in the following sections.

Bitmap fonts

Bitmap fonts consist of a series of character images rendered to small rectangular bitmaps and stored sequentially in a single file. The file may or may not have a header. Most bitmap font files are monochrome, and most store font in uniformly sized rectangles to facilitate speed of access. Characters stored in bitmap format may be quite elaborate, but the size of the file increases, and, consequently, speed and ease of use decline with increasingly complex images.

The advantages of bitmap files are speed of access and ease of use--reading and displaying a character from a bitmap file usually involve little more than reading the rectangle containing the data into memory and displaying it on the display surface of the output device. Sometimes, however, the data is analyzed and used as a template for display of the character by the rendering application. The chief disadvantages of bitmap fonts are that they are not easily scaled, and that rotated bitmap fonts look good only on screens with square pixels.

Most character-based systems, such as MS-DOS, character-mode UNIX, and character terminalbased systems use bitmap fonts stored in ROM or on disk. However, bitmap fonts are seldom used today when sufficient processing power is available to enable the use of other types of font data.

Stroke fonts

Stroke fonts are databases of characters stored in vector form. Characters can consist of single strokes or may be hollow outlines. Stroke character data usually consists of a list of line endpoints meant to be drawn sequentially, reflecting the origin of many stroke fonts in applications supporting pen plotters. Some stroke fonts may be more elaborate, however, and may include instructions for arcs and other curves. Perhaps the best-known and most widely used stroke fonts were the Hershey character sets, which are still available online.

The advantages of stroke fonts are that they can be scaled and rotated easily, and that they are composed of primitives, such as lines and arcs, which are well-supported by most GUI operating environments and rendering applications. The main disadvantage of stroke fonts is that they generally have a mechanical look at variance with what we've come to expect from reading highquality printed text all our lives.

Stroke fonts are seldom used today. Most pen plotters support them, however. You also may need to know more about them if you have a specialized industrial application using a vector display or something similar.

Spline-based outline fonts

Character descriptions in spline-based fonts are composed of control points allowing the reconstruction of geometric primitives known as splines. There are a number of types of splines, but they all enable the drawing of the subtle, eye-pleasing curves we've come to associate with high-quality characters that make up printed text. The actual outline data is usually accompanied by information used in the reconstruction of the characters, which can include information about kerning, and information useful when scaling characters that are very large or very small ("hints").

The advantages of spline-based fonts are that they can be used to create high-quality character representations, in some cases indistinguishable from text made with metal type. Most traditional fonts, in fact, have been converted to spline-based outlines. In addition, characters can be scaled, rotated, and otherwise manipulated in ways only dreamed about even a generation ago.

Unfortunately, the reconstruction of characters from spline outline data is no trivial task, and the higher quality afforded by spline outlines comes at a price in rendering time and program development costs.

Page Description Language (PDL) Formats

Page description languages (PDLs) are actual computer languages used for describing the layout, font information, and graphics of printed and displayed pages. PDLs are used as the interpreted languages used to communicate information to printing devices, such as hardcopy printers, or to display devices, such as graphical user interface (GUI) displays. The greatest difference is that PDL code is very device-dependent. A typical PostScript file contains detailed information on the output device, font metrics, color palettes, and so on. A PostScript file containing code for a 4-color, A4-sized document can only be printed or displayed on a device that can handle these metrics.

Markup languages, on the other hand, contain no information specific to the output device. Instead, they rely on the fact that the device that is rendering the markup language code can adapt to the formatting instructions that are sent to it. The rendering program chooses the fonts, colors, and method of displaying the graphical data. The markup language provides only the information and how it is structured.

Although PDL files can contain graphical information, we do not consider PDLs to be graphics file formats any more than we would consider a module of C code that contains an array of graphical information to be a graphics file format. PDLs are complete programming languages, requiring the use of sophisticated interpreters to read their data; they are quite different from the much simpler parsers used to read graphics file formats.

5. Different types of multimedia tool for editing

1. MPEG

MPEG was developed and defined by ISO/IEC JTC/SC 29/WG 11 to cover motion video as well as audio coding. In light of the state of the art in CD technology, the goal was a compressed stream data rate of about 1.2Mbit/s. MPEG specifies a maximum data rate of 1,856,000bit/s, which should not be exceeded [ISO93b]. The data rate for each audio channel can be chosen between 32 and 448 Kbit/s in increments of 16Kbit/s. This data rate enables video and audio compression of acceptable quality. Since1993, MPEG has been an International Standard (IS) [ISO93b]. MPEG explicitly takes into account developments in other standardization activities:

- JPEG: Since a video sequence can be regarded as a sequence of still images and the JPEG standard development was always ahead of the MPEG standardization, the MPEG standard makes use of the results of the JPEG work.

- H.261: Since the H.261 standard already existed during the MPEG work, the MPEG group strived to achieve at least certain compatibility between the two standards in some areas. This should simplify implementations of MPEG that also support H.261. In any case, technically MPEG is the more advanced technique. Conversely, H.263 borrowed techniques from MPEG.

Although mainly designed for asymmetric compression, a suitable MPEG implementation can also meet symmetric compression requirements. Asymmetric coding requires considerably more effort for encoding than for decoding. Compression is carried out once, whereas decompression is performed many times. A typical application area is retrieval systems. Symmetric compression is characterized by a comparable effort for the compression and decompression processing. This is a requirement for interactive dialogue applications, as is a bounded processing delay for the processing.

Besides the specification of video coding and audio coding, the MPEG standard provides a system definition, which describes the combination of individual data streams into a common stream.

6.5.2 Video Encoding

In the image preparation phase (according to the reference scheme shown in Figure2-1) MPEG, unlike JPEG but similar to H.263, defines the format of an image very precisely.

Image Preparation:

An image must consist of three components. In addition to the luminance Y there are two color difference signals C_r and C_b(similar to the YUV format). The luminance component has twice as many samples in the horizontal and vertical axes as the other components, that is, there is color subsampling. The resolution of the luminance component should not exceed 768 x 576 pixels. The pixel depth is eight bits in each component.

An MPEG data stream also contains information that is not part of a data stream compressed according to the JPEG standard, for example the pixel aspect ratio. MPEG supports 14 different pixel aspect ratios. The most important are:

- A square pixel (1:1) is suitable for most computer graphics systems.
- For a 625- line image, a ratio of 16:9 is defined (European HDTV).
- For a 525- line image, a ratio of 16:9 is defined (U.S. HDTV).

- For a 702 x 575 pixel images, an aspect ratio of 4:3 is defined
- For a 711 x 487 pixel images, an aspect ratio of 4:3 is also defined

The image refresh frequency is also encoded in the data stream. So far, eight frequencies have been defined (23.976 Hz, 24Hz, 25Hz, 29.97Hz, 30Hz, 50Hz, 59.94Hz, and 60Hz), so no low image refresh frequencies are permitted.

Temporal prediction of still images usually yields a considerable data reduction. Areas within an image with strong, irregular motions can only be reduced by a ratio similar to that of intraframe coding. The use of temporal predictors requires the storage of a huge amount of previously determined by balancing the required storage capacity against the achievable compression rate.

In most cases, predictive coding only makes sense for parts of an image and not for the whole image. The image is thus divided into areas called macro blocks. An MPEG macro block is partitioned into 16 x 16 pixels for the luminance component and 8 x 8 pixels for each of the two chrominance components. These sizes are well suited for compression based on motion estimation. This is a compromise between the computational effort required for estimation and the resulting data reduction. A macro block is formed of six blocks of 8 x 8 pixels, ordered as follows: first four blocks for the luminance component then the two chrominance blocks. There are no user-defined MCU‘s as in JPEG since, given the defined frame rates, the maximum time to present an image is 41.7ms. The three components are compressed and decompressed together. From the MPEG user‘s perspective, there is no fundamental advantage to progressive image display over sequential display.

---

3. Image Processing

~~For the image processing stage, MPEG supports four types of image coding. The reason for this~~ is the contradictory demands of efficient coding and random access. In order to achieve a high compression ratio, temporal redundancies of successive images need to be exploited. Fast random access requires that images be coded individually. Hence the following image types are distinguished:

- I frames (intra coded pictures) are coded without using information about other frames (intraframe coding). An I frame is treated as a still image. Here MPEG falls back on the results of JPEG. Unlike JPEG, real-time compression must be possible. The compression rate is thus the lowest within MPEG. I frames form the anchors for random access.

- P frames (predictive coded pictures) require information about previous I and/or P frames for encoding and decoding. Decoding a P frame requires decompression of the last I frame and any intervening P frames. In return, the compression ratio is considerably higher than for I frames. A P frame allows the following P frame to be accessed if there are no intervening I frames.
- B frames (bidirectionally predictive coded pictures) require information from previous and following I and/or P frames. B frames yield the highest compression ratio attainable in MPEG. A B frame is defined as the difference from a prediction based on a previous and a following I or P frame. It cannot, however, ever serve as a reference for prediction coding of other pictures.
- D frames (DC coded pictures) are intraframe coded and can be used for efficient fast forward. During the DCT, only the DC- coefficients are coded; the AC- coefficients are ignored.

  Figure shows a sequence of I, P and B frames. This example illustrates the prediction for the first P frame and the bidirectional prediction for a B frame. Note that the order in which the images are presented differs from the actual decoding order if B frames are present in an MPEG-coded video stream.

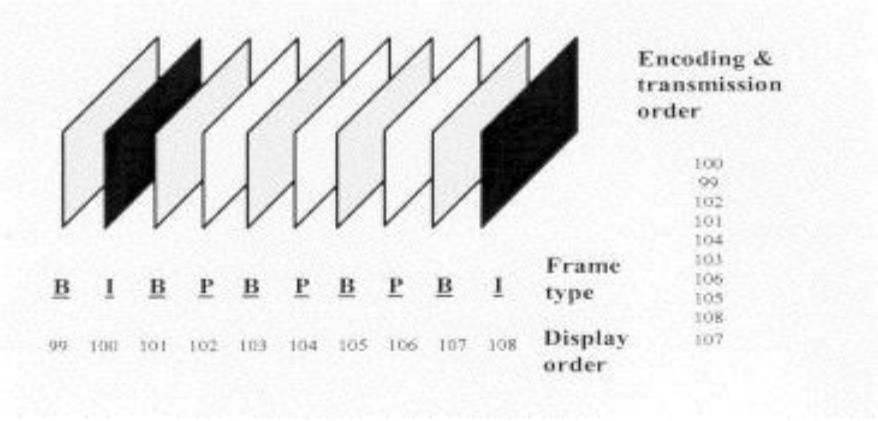


Figure 2-1 types of individual in MPEG: I, B, and p frames.

The pattern of I, P and B frames in a sequence is determined by the MPEG application. For random access, the ultimate resolution would be attained by encoding the entire stream using I frames. The highest compression rate can be achieved by using as many B frames as possible. For practical applications, the sequence IBBPBBPBBIBBPBBPBB..has proven to be useful. This permits random access with a resolution of nine still images (i.e., about 330ms) and still provides a very good compression ratio. Every 15 images include one I frame.

The following detailed description of image processing, quantization and entropy encoding distinguishes the four image types.

I Frames

I frames are encoded by performing a DCT on the 8×8 locks defined within the macro blocks, as in JPEG. The DC-coefficients are then DPCM coded, and differences between consecutive blocks of each component are calculated and transformed into variable-length code words. ACcoefficients are run-length encoded and then transformed into variable length code words. MPEG distinguishes two types of macro blocks: those that contain only coded data and those that additionally contain a parameter used for the characteristic curve used for subsequent quantization.

P Frames

The coding of P frames exploits the fact that in consecutive images, some areas of the image often shift, but do not change. To encode a block, the most similar macro block in the preceding image must be determined. This is done by calculating the differences between the absolute values of the luminance component for each macro block in the preceding image. The macro block with the lowest sum differences is the most similar. MPEG does not specify an algorithm for motion estimation, rather specifies the coding of the result. Only the motion vector and the small difference between the macro blocks need to be encoded. The search range, that is, the maximum length of the motion vector, is not defined by the standard. As the search range is increased, the motion estimation becomes better, although the computation becomes slower.

P frames can consist of macro blocks as in I frames, as well as six different predictive macro blocks. The coder essentially decides whether to code a macro block predictively or like an I frame and whether it should be coded with a motion vector. A P frame can thus also include macro blocks that are coded in the same way as I frames.

In coding P-frame-specific macro blocks, differences between macro blocks as well as the motion vector need to be considered. The difference values between all six 8×8 pixel blocks of a macro block being coded and the best matching macro block are transformed using a twodimensional DCT. Further data reduction is achieved by not further processing blocks where all DCT coefficients are zero. This is coded by inserting a six-bit value into the encoded data stream. Otherwise, the DC- and AC- coefficients are then encoded using the same technique. This differs from JPEG and from the coding of I frame macro blocks. Next, run-length encoding is applied and a variable length encoding is determined according to an algorithm similar to

Huffman. Since motion vectors of adjacent macro blocks often differ only slightly, they are DPCM coded. The result is again transformed using a table; a subsequent calculation performs a transformation into variable-length coded words.

B Frames

In addition to the previous P or I frame, B frame prediction takes into account the following P or I frame. The following example illustrates the advantages of bidirectional prediction:
In a video sequence, a ball moves from left to right in front of a static background. In the left area of the scene, parts of the image appear that in previous images were covered by the moving ball. A prediction of these areas would ideally be derived from the following, not from the previous, image. A macro block can be derived from macro blocks of previous and following P and/or I frames. Motion vectors can also point in orthogonal directions (i.e., in x direction and in y direction). Moreover, a prediction can interpolated two similar macro blocks. In this case, two motion vectors are encoded and one difference block is determined between the macro block to be encoded and the interpolated macro block. Subsequent quantization and entropy encoding are performed as for P-frame specific macro blocks. Since B frames cannot serve as reference frames for subsequent decoding, they need not be stored in the decoder.

D Frames

D frames contain only the low- frequency components of an image. A D-frame always consists of one type of macro block and only the DC- codfficients of the DCT are coded. D Frames are used for fast-forward display. This could also be realized by a suitable placement of I frames. For fast- forward, I frames must appear periodically in the data stream. In MPEG, slow-rewind playback requires considerable storage. All images in a so-called group of pictures must decoded forwards and stored before they can be displayed in reverse.

4. Quantization

Concerning quantization, it should be noted that AC-coefficients of B and P frames are usually very large values, whereas those of I frames are very small. Thus, MPEG quantization adjusts itself accordingly. If the data rate increases too much, quantization becomes more coarse. If the data rate falls, then quantization is performed with finer granularity.

5. Audio Coding

MPEG audio coding is compatible with the coding of audio data used for Compact Disc Digital Audio (CD-DA) and Digital Audio Tape (DAT). The most important criterion is the choice of sample rate of 44.1kHz or 48kHz (additionally 32kHz) at 16bits per sample value. Each audio signal is compressed to 64, 96, 128, or 192Kbit/s.

Three quality levels (layers) are defined with different encoding and decoding complexity. An implementation of a higher layer must be able to decode the MPEG audio signals of lower layers. Similar to two-dimensional DCT for video, a transformation into the frequency domain is applied for audio. The Fast Fourier Transform (FFT) is a suitable technique. As shown in Fig, the relevant portion of the spectrum is divided into 32 non overlapping subbands. The audio signal is thus split into 32 subbands. Different components of the spectrum can then be quantized differently. In parallel with the actual FFT, the noise level in each subband is determined using a psychoacoustic model. At a higher noise level, a coarser quantization is performed. A lower noise level results in finer quantization. In the first and second layers, the appropriately quantized spectral components are simply PCM-encoded. The third layer additionally performs Huffman Coding.

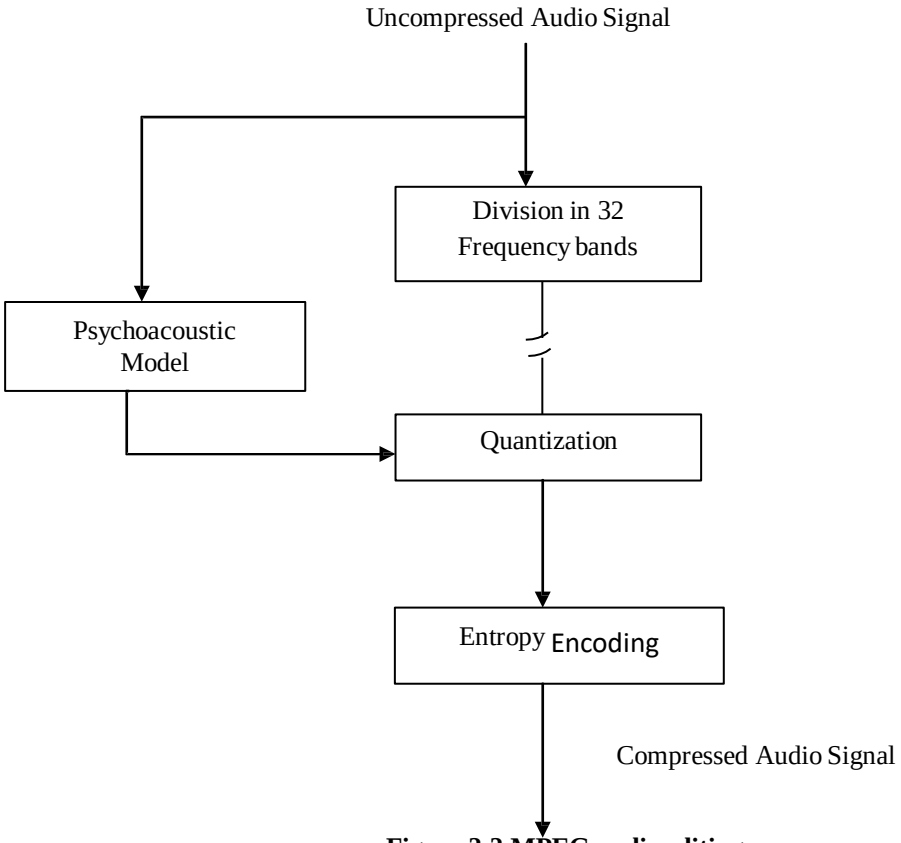


Figure 2-2 MPEG audio editing.

Audio coding can be performed on stereo sound, a single channel, or two independent channels. MPEG provides for two types of stereo sound. In the first case, two channels are processed completely independently. In the joint stereo mode, MPEG achieves a higher compression ratio y exploiting redundancies between the two channels.

Each layer defines 14 fixed bit rates for the coded audio data stream, which in MPEG are addressed by a bit rate index. The minimal value is always 32Kbit/s. The layers support different maximal bit rates: layer 1 allows for a maximum of 448Kbit/s, layer 2 for 384Kbit/s, and layer 3 for 320Kbit/s. For layers 1 and 2, a decoder is not required to support a variable bit rate. In layer 3, a variable bit rate is specified by allowing the bit rate index to be switched. In layer 2, not all combinations of bit rate and mode are allowed.

- 32 Kbit/s, 48 Kbit/s, 56 Kbit/s, and 80 Kbit/s are only allowed for a single channel.
- 64 Kbit/s, 96 Kbit/s, 112 Kbit/s, 128 Kbit/s, 160 Kbit/s, and 192 Kbit/s are allowed for all modes.
- 224 Kbit/s, 256 Kbit/s, 320 Kbit/s, and 384 Kbit/s are allowed for the stereo, joint stereo and dual channel modes.

6. Data Stream

Like JPEG, MPEG specifies a precise syntax for the compressed audio and video data streams.

Audio Stream

An audio stream is comprised of frames, which are made up of audio access units, which in turn are divided into slots. At the lowest coding complexity (layer 1), a slot consists of 4 bytes; otherwise it is one byte. Frames consist of a fixed number of samples. An audio access unit is the smallest compressed audio sequence that can be completely decoded independently of all other data. At 48kHz, the audio access units contained in a frame have a play time of 8ms; at 44.1 kHz, the play time is 8.7 ms; and at 32 kHz, the play time is 12ms. In the case of stereo, data from both channels are included in one frame.

7. Video Stream

A video stream is comprised of 6 layers:

1. At the highest level, the sequence layer, data buffering is handled. For example, it does not make sense to generate a data stream that places excessive demands on storage space in the decoder. Among other things, the beginning of the sequence layer thus includes two entries: the constant bit rate of the sequence and the minimum storage capacity required during decoding.

   A video buffer verifier is added after the quantizer that uses the bit rate resulting from decoding in order to monitor the decoding delay. The video buffer verifier influences the quantizer and forms a type of control loop. Successive sequences can have varying bit rates. During decoding of several consecutive sequences, often no data output between the end of one sequence and the beginning of the next sequence because the underlying decoder parameters need to be updated and an initialization performed.

2. The group of pictures layer is the next layer. This layer contains at least an I frame, which must be one of the first images. Random access to this image is always possible. At this layer, it is possible to distinguish between the order of images in the data stream and their display order. The data stream must always begin with an I frame so that the decoder can first decode and store the reference image. In order of presentation, however , B frames can precede the I frame. The following example illustrates the difference between decoding order and display order:

3. The picture layer contains a whole still image. The image‘s temporal reference is defined using an image number. This number is shown belwo each image in the example above. There are also data fields defined in this layer that are not yet used in MPEG. These fields are reserved for future extensions and may not be used by the decoder.

4. The next layer is the slice layer. Each slice consists of macro blocks, the number of which can vary from image to image. A slice also includes the scaling used for DCT quantization of all its macro blocks.

5. The next layer is the macro block layer. This contains a macro block with the properties described above.

6. The lowest layer is the block layer, also described above.

Decoding order													
Type of frame	I	P	B	B	P	B	B	P	B	B	I	B	B

Frame number	1	4	2	3	7	5	6	10	8	9	13	11	12
Display order													
Type of frame	I	P	B	B	P	B	B	P	B	B	I	B	B
Frame number	1	2	3	4	5	6	7	8	9	10	11	12	13

6.5.8 MPEG-2

MPEG-2 is a standard for "the generic coding of moving pictures and associated audio information". It describes a combination of lossy video compression and lossy audio data compression methods, which permit storage and transmission of movies using currently available storage media and transmission bandwidth.

Systems

MPEG-2 includes a Systems section, part-1, that defines two distinct, but related, container formats. One is the *transport stream*, a data packet format designed to transmit one data packet in four ATM data packets for streaming digital video and audio over fixed or mobile transmission mediums, where the beginning and the end of the stream may not be identified, such as radio frequency, cable and linear recording mediums, examples of which include ATSC/DVB/ISDB/SBTVD broadcasting, and HDV recording on tape. The other is the *program stream*, an extended version of the MPEG-1container format without the extra overhead of previously mentioned *transport stream* designed for random access storage mediums such as hard disk drives, optical discs and flash memory.

four network ordered bytes to the end of every transport stream packet, which is used as a random access timing reference for faster read times over using the *Program Clock Reference* contained in the adaption section of the primary stream.

MPEG-2 Systems is formally known as ISO/IEC 13818-1 and as ITU-T Rec. H.222.0. ISO authorized the "SMPTE Registration Authority, LLC" as the registration authority for MPEG-2 format identifiers. The registration descriptor of MPEG-2 transport is provided by ISO 13818-1 in order to enable users of the standard to unambiguously carry data when its format is not necessarily a recognized international standard. This provision will permit the MPEG-2 transport

standard to carry all types of data while providing for a method of unambiguous identification of the characteristics of the underlying private data.

Video

The Video section, part-2 of MPEG-2, is similar to the previous MPEG-1 standard, but also provides support for interlaced video, the format used by analog broadcast TV systems. MPEG-2 video is not optimized for low bit-rates, especially less than 1 Mbit/s at standard definition resolutions. All standards-compliant MPEG-2 Video decoders are fully capable of playing back MPEG-1 Video streams conforming to the Constrained Parameters Bitstream syntax. MPEG-2/Video is formally known as ISO/IEC 13818-2 and as ITU-T Rec. H.262.

Audio

MPEG-2 introduces new audio encoding methods compared to MPEG-1.

MPEG-2 Part 3

The MPEG-2 Audio section, defined in Part 3 (ISO/IEC 13818-3) of the standard, enhances MPEG-1's audio by allowing the coding of audio programs with more than two channels, up to 5.1 multi channel. This method is backwards-compatible (also known as MPEG-2 BC), allowing MPEG-1 audio decoders to decode the two main stereo components of the presentation. MPEG2 part 3 also defined additional bit rates and sample rates for MPEG-1 Audio Layer I, II and III.

MPEG-2 BC (backward compatible with MPEG-1 audio formats).

- Low bitrate encoding with halved sampling rate (MPEG-1 Layer 1/2/3 LSF - a.k.a. MPEG-2 LSF - "Low Sampling Frequencies")
- Multichannel encoding with up to 5.1 channels, a.k.a. MPEG Multichannel.

MPEG-2 Part 7

Part 7 (ISO/IEC 13818-7) of the MPEG-2 standard specifies a rather different, non-backwardscompatible audio format(also known as MPEG-2 NBC). Part 7 is referred to as MPEG- 2 AAC. AAC is more efficient than the previous MPEG audio standards, and is in some ways less complicated than its predecessor, MPEG-1 Audio, Layer 3, in that it does not have the hybrid filter bank. It supports from 1 to 48 channels at sampling rates of 8 to 96 kHz, with

multichannel, multilingual, and multiprogramming capabilities.Advanced Audio is also defined in Part 3 of the MPEG-4 standard.

6.5.9 MPEG-4

MPEG-4 is a method of defining compression of audio and visual (AV) digital data. It was introduced in late 1998 and designated a standard for a group of audio and video codingformats and related technology agreed upon by the ISO/IEC Moving Picture Experts Group(MPEG) (ISO/IEC JTC1/SC29/WG11) under the formal standard ISO/IEC 14496 – *Coding of audiovisual objects*. Uses of MPEG-4 include compression of AV data for web (streaming media) and CD distribution, voice (telephone, videophone) and broadcast televisionapplications.

Background

MPEG-4 absorbs many of the features of MPEG-1 and MPEG-2 and other related standards contributing to this, adding new features such as (extended) VRML support for 3D rendering, object-oriented composite files (including audio, video and VRML objects), support for externally specified Digital Rights Management and various types of interactivity.AAC (Advanced Audio Coding) was standardized as an adjunct to MPEG-2 (as Part 7) before MPEG4 was issued.

MPEG-4 is still an evolving standard and is divided into a number of parts. Companies promoting MPEG-4 compatibility do not always clearly state which "part" level compatibility they are referring to. The key parts to be aware of are MPEG-4 part 2 (including Advanced Simple Profile, used by codecssuch as DivX, Xvid, Nero Digital and 3ivx and byQuickTime 6) and MPEG-4 part 10 (MPEG-4 AVC/H.264 or Advanced Video Coding, used by the x264 encoder, by Nero Digital AVC, by QuickTime 7, and by high-definition videomedia like Blu-ray Disc).

Most of the features included in MPEG-4 are left to individual developers to decide whether to implement them. This means that there are probably no complete implementations of the entire MPEG-4 set of standards. To deal with this, the standard includes the concept of "profiles" and "levels", allowing a specific set of capabilities to be defined in a manner appropriate for a subset of applications.

Initially, MPEG-4 was aimed primarily at low bit-rate video communications; however, its scope as a multimedia coding standard was later expanded. MPEG-4 is efficient across a variety of bitrates ranging from a few kilobits per second to tens of megabits per second. MPEG-4 provides the following functions:

- Improved coding efficiency over MPEG-2.
- Ability to encode mixed media data (video, audio, speech).
- Error resilience to enable robust transmission.
- Ability to interact with the audio-visual scene generated at the receiver.

Overview

MPEG-4 provides a series of technologies for developers, for various service-providers and for end users:

- MPEG-4 enables different software and hardware developers to create multimedia objects possessing better abilities of adaptability and flexibility to improve the quality of such services and technologies as digital television, animation graphics, the World Wide Web and their extensions.
- Data network providers can use MPEG-4 for data transparency. With the help of standard procedures, MPEG-4 data can be interpreted and transformed into other signal types compatible with any available network.
- The MPEG-4 format provides end users with a wide range of interaction with various animated objects.
- Standardized Digital right signaling, otherwise known in the MPEG community as Intellectual Property Management and Protection (IPMP).

The MPEG-4 format can perform various functions, among which might be the following:

- Multiplexes and synchronizes data, associated with media objects, in such a way that they can be efficiently transported further via network channels.
- Interaction with the audio-visual scene, which is formed on the side of the receiver.

6.5.10 MPEG-7

After MPEG-3 did not materialize, it was decided not to allow MPEG-5 or MPEG-6 either, in order to preserve the logic of the sequence MPEG-1 (+1) to MPEG-2 (+2) to MPEG-4 (+#) to MPEG-7 (+4)…….

MPEG-7 is not meant as another compression format, rather the intention of the MPEG-7 ISO Working Group was to establish a metadata to supplement content coded in other formats (such as MPEG-4). The use of metadata aims at a new type of extension to the coding of multimedia data: mainly improved searching techniques and display strategies, but also consistency checking and, for example, scaling that incorporates consistency and priority. MPEG-7 calls for the integration and extension of other media content formats (especially MPEG-4) in this way.

In addition to the coding of metadata, MPEG-7 will define interfaces for working in tandem with tools for automatic content analysis and search engines, but not these services themselves.

6.6 Motion JPEG

In multimedia, **Motion JPEG** (**M-JPEG** or **MJPEG**) is a video format in which each video frame or interlaced field of a digital video sequence is compressed separately as a JPEGimage. Originally developed for multimedia PC applications, M-JPEG is now used by video-capture devices such as digital cameras, IP cameras, and webcams; and by non-linear video editing systems.

Video capture and editing

M-JPEG is frequently used in non-linear video editing systems. Modern desktop CPUs are powerful enough to work with high-definition video so no special hardware is required and they in turn offer native random-access to a frame, M-JPEG support is also widespread in videocapture and editing equipment.

Encoding

M-JPEG is an intraframe only compression scheme (compared with the more computationally intensive technique of inter frame prediction). Whereas modern inter frame video formats, such as MPEG1, MPEG2 and H.264/MPEG-4 AVC, achieve real-world compression-ratios of 1:50 or better, M-JPEG's lack of inter frame prediction limits its efficiency to 1:20 or lower, depending on the tolerance to spatial artifacting in the compressed output. Because frames are compressed

independently of one another, M-JPEG imposes lower processing and memory requirements on hardware devices.

As a purely intraframe compression scheme, the image-quality of M-JPEG is directly a function of each video frame's static (spatial) complexity. Frames with large smooth-transitions or monotone surfaces compress well, and are more likely to hold their original detail with few visible compression artifacts. Frames exhibiting complex textures, fine curves and lines (such as writing on a newspaper) are prone to exhibit DCT-artifacts such as ringing, smudging, and macro -blocking. M-JPEG compressed-video is also insensitive to motion-complexity, i.e. variation over time. It is neither hindered by highly random motion (such as the surface-water turbulence in a large waterfall), nor helped by the absence of motion (such as static landscape shot by tripod), which are two opposite extremes commonly used to test inter frame video-formats.

For QuickTime formats, Apple has defined two types of coding: MJPEG-A and MJPEG-B. MJPEG-B no longer retains valid JPEG Interchange Files within it; hence it is not possible to take a frame into a JPEG file without slightly modifying the headers.

Advantages

- **It** is simple to implement because it uses a mature compression standard (JPG) with welldeveloped libraries, and it's an intraframe method of compression.
- It tolerates rapidly changing motion in the video stream, whereas compression schemes using interframe compression can often experience unacceptable quality loss when the video content changes significantly between each frame.
- Because the M-JPEG standard emerged from a market-adoption process rather than a standards body, it enjoys broad client support most major web browsers and players provide native support and plug-ins are available for the rest.
- Minimal hardware is required because it is not computationally intensive.

Disadvantages

- Unlike the video formats specified in international standards such as MPEG-2 and the format specified in the JPEG still-picture coding standard, there is no document that defines a single exact format that is universally recognized as a complete specification of ―Motion JPEG‖ for use in all contexts. This raises compatibility concerns about file outputs from different manufacturers. However, each particular file format usually has some standard how M-JPEG is encoded. For example, Microsoft documents their standard format to store M-JPEG in AVI files, Apple documents how M-JPEG is stored in QuickTime files, RFC 2435 describes

how M-JPEG is implemented in an RTP stream, and an M-JPEG CodecID is planned for the Matroskafile format.

- JPEG is inefficient, using more bits to deliver similar quality, compared to more modern formats (such as JPEG 2000 and H.264/MPEG-4 AVC). Since the development of the original JPEG standard in the early 1990s, technology improvements have made not only to the JPEG format but to the inter frame compression schemas possible.
- Technology improvements can be found in the designs of H.263v2 Annex I and MPEG-4 Part 2, that use frequency-domain prediction of transform coefficient values, and inH.264/MPEG-4 AVC, that use spatial prediction and adaptive transform block size techniques. There is also more sophisticated entropy coding than what was practical when the first JPEG design was developed. All of these new developments make M-JPEG an inefficient recording mechanism.

6.7 Summary

Three-dimensional data files store descriptions of the shape and color of 3D models of imaginary and real-world objects. 3D models are typically constructed of polygons and smooth surfaces, combined with descriptions of related elements, such as color, texture, reflections, and so on, that a rendering application can use to reconstruct the object.

VRML is made up of nodes put into a hierarchy that describe a scene of one or more objects. VRML contains basic geometric shapes that can be combined to create more complex objects. The **shape** node is a generic node for all objects in VRML. The **Box, Cylinder, Cones, and Sphere** are geometry nodes that place objects in the virtual world.

MPEG was developed and defined by ISO/IEC JTC/SC 29/WG 11 to cover motion video as well as audio coding. In light of the state of the art in CD technology, the goal was a compressed stream data rate of about 1.2Mbit/s. MPEG specifies a maximum data rate of 1,856,000bit/s, which should not be exceeded [ISO93b]. The data rate for each audio channel can be chosen between 32 and 448 Kbit/s in increments of 16Kbit/s. This data rate enables video and audio compression of acceptable quality. Since1993, MPEG has been an International Standard (IS) [ISO93b].

In multimedia, **Motion JPEG** (**M-JPEG** or **MJPEG**) is a video format in which each video frame or interlaced field of a digital video sequence is compressed separately as a JPEGimage.

Originally developed for multimedia PC applications, M-JPEG is now used by video-capture devices such as digital cameras, IP cameras, and webcams; and by non-linear video editing systems.

8. Keywords

3D, VRML, Scene elements, PDL, MPEG, MPEG-2, MPEG-4, MPEG-7, MJPEG.

9. Exercises

1. Explain briefly 3D file formats.
2. Discuss in detail about VRML.
3. Explain briefly Page Description language (PDL).
4. Explain briefly the audio and video encoding techniques in MPEG.
5. Discuss in detail about I, P, B, and D frames.
6. Differentiate between MPEG-2, MPEG-4, and MPEG-7.
7. Differentiate between MPEG and Motion JPEG.

10. References

5. Ralf Steinmentz ,klaraNaestedt: Multimedia Fundamentals: Vol 1- Media Coding and Content processing, 2nd edition, PHI, Indian Reprint 2008.
6. Donald Hearn and M. Pauline Baker, computer graphics, 3rd edition, Pearson.

UNIT-7 Transformation Techniques

Structure

1. Objectives
2. Introduction
3. Basic two dimensional geometric transformations
4. Matrix representations and homogeneous coordinates
5. Two dimensional composite transformations
6. Summary
7. Keywords

8. Questions
9. References

1. Learning Objectives

After studying this unit, you will be able to,

- Discuss Basic Computer graphics Two-dimensional Geometric transformations,
- Explain Matrix representations and Homogeneous coordinates,
- Describe Two-Dimensional Composite Transformations and Other Two-Dimensional transformations.

1. Introduction

We know to describe a scene in terms of graphics primitives, such as line segments and the attributes associated with these primitives. And we have explored the scan line algorithms for displaying output primitives on a raster device. Now, we take a look at transformation operations that we can apply to objects to reposition or resize them. These operations are also used in the viewing routines that convert a world-coordinate scene description to a display for an output device. In addition, they are used in a variety of other applications, such as computer-aided design and computer animation. An architect, for example, creates a layout by arranging the orientation and size of the component parts of a design, and a computer animator develops a video sequence by moving the ―camera‖ position or the objects in a scene along specified paths. Operations that are applied to the geometric description of an object to change its position, orientation, or size are called **geometric transformations.**

Sometimes geometric transformation operations are also referred to as***modeling transformations***, but some graphics packages make a distinction between the two. In general, modeling transformations are used to construct a scene or to give the hierarchical description of a complex object that is composed of several parts, which in turn could be composed of simpler parts, and so forth. As an example, an aircraft consists of wings, tail, fuselage, engine, and other components, each of which can be specified in terms of second-level components, and so on, down the hierarchy of component parts. Thus, the aircraft can be described in terms of these components and as associated ―modeling‖ transformation for each one that describes how that component is to be fitted into the overall aircraft design. Geometric transformations, on the other

hand, can be used to describe how objects might move around in a scene during an animation sequencer simply to view them from another angle. Therefore, some graphics packages provide two sets of transformation routines, while other packages have a single set of functions that can be used for both geometric transformations and modeling transformations.

7.2 Basic Two-Dimensional Geometric Transformations

The geometric-transformations functions that are available in all graphics packages are those for translation, rotation, and scaling. Other useful transformation routines that are sometimes included in a package are reflection and shearing operations. To introduce the general concepts associated with geometric transformations, we first consider operations in two dimensions, and then we discuss how these basic ideas can be extended to three-dimensional scenes. Once we understand the basic concepts, we can easily write routines to perform geometric transformations on objects in a two-dimensional scene.

Two-Dimensional Translation

We perform a **translation** on a single coordinate point by adding offsets to its coordinates so as to generate a new coordinate position. In effect, we are moving the original point position along a straight-line path to its new location. Similarly, a translation is applied to an object that is defined with multiple coordinate positions, such as a quadrilateral, by relocating all the coordinate positions by the same displacement along parallel paths. Then the complete object is displayed at the new location.

To translate a two-dimensional position, we add **translation distances** t_x and t_y to the original coordinates (x, y) to obtain the new coordinate position (x_1, y_1) as shown in Fig. 3-1.

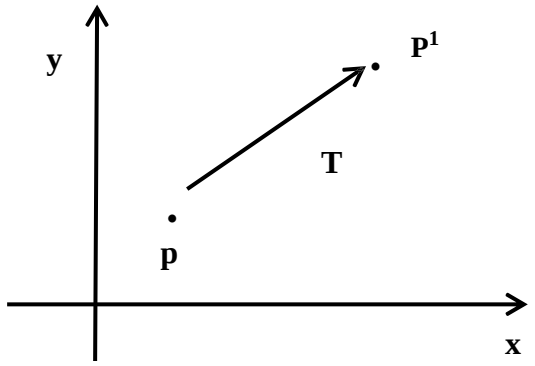


Figure 3-1 Translating a point from position p to position p^1 using translation vector T.

To translate a two-dimensional position, we add **translation distances** t_x and t_y to the original coordinates position ***(x, y)*** to obtain the new coordinate position (x^1, y^1) as shown in Fig. 3-1.

$$x^1 = x + t_x, \qquad y^1 = y + t_y \tag{3-1}$$

The translation distance pair (t_x, t_y) is called a **translation vector**or **shift vector.**

We can express the translation equations 3-1 as a single matrix equation by using the following column vectors to represent coordinate positions and the translation vector.

, **P = □□□x□□** , **P□ = □x□□□y□□ □ T =□□□ttxy□□□** (3-2) **y□**

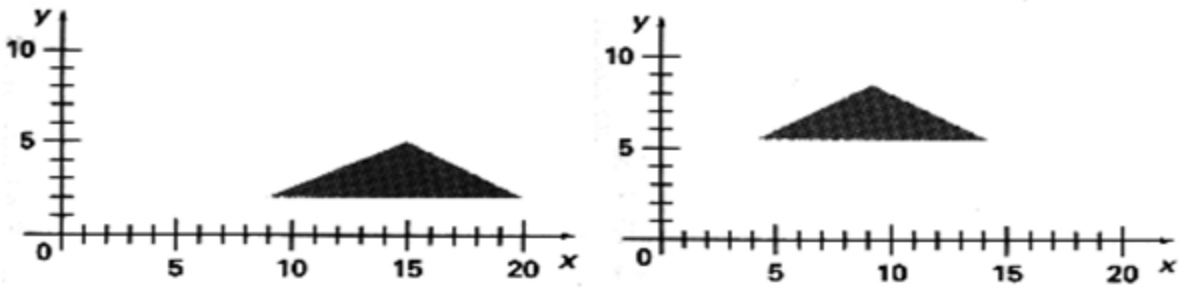
□

This allows us to write the two-dimensional translation equations in the matrix form,

$$P^1 = P + T \tag{3-3}$$

Translation is a ***rigid-body transformation*** that moves objects without deformation. That is, every point on the objects is translated by the same amount. A straight-line segment is translated by applying the transformation equation 3-3 to each of the two line endpoints and redrawing the line between the new endpoint positions. A polygon is translated similarly. We add a translation vector to the coordinate position of each vertex and then regenerate the polygon using the new set of vertex coordinates. Figure 3-2 illustrates the application of a specified translation vector to move an object from one position to another.

The following routine illustrates the translation operations. An input translation vector is used to move the n vertices of a polygon from one world-coordinate position to another, and OpenGL routines are used to regenerate the translated polygon.



(a) (b)

Figure 3-2 Moving a polygon from position (a) to position (b) with the translation vector (-5.50, 3.75)

If we want to delete the original polygon, we could display it in the background color before translating it. Other methods for deleting picture components are available in some graphics packages. Also, if we want to save the original polygon positions can, we can store the translated positions in a different array.

Similar methods are used to translate other objects. To change the position of a circle or ellipse, we translate the center coordinates and redraw the figure in the new locations. For a spline curve, we translate the points that define the curve path and then reconstructed the curve path and then reconstructed the curve sections between the new coordinate positions.

Two-Dimensional Rotation

We generate a **rotation** transformation of an object by specifying a **rotation axis** and a **rotation angle.** All points of the object are then transformed to new positions by points through the specified angle about the rotation axis.

A two-dimensional rotation of an object is obtained by repositioning the object along a circular path in the ***xy*** plane. In this case, we are rotating the object about a rotation axis that is perpendicular to the ***xy*** plane (parallel to the coordinate z axis). Parameters for the twodimensional rotation are the rotation angle θ and a position (x_r, y_r), called a **rotation point (**or **pivot point),** about which the object is to be rotated (Fig. 3-3). The pivot point is the intersection position of the rotation axis with the ***xy*** plane. A positive value for the angle θ defines a counterclockwise rotation about the pivot point, as in Fig. 3-3, and a negative value rotates objects in the clockwise direction.

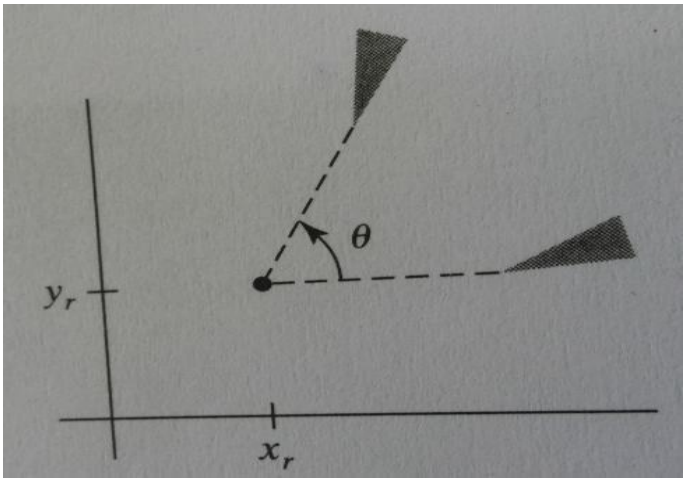


Figure 3-3 Rotation of an object through angle θ about the pivot point (x_r, y_r).

To simplify the explanation of the basic method, we first determine the transformation equations for rotation of a point position **P** when the pivot point is at the coordinate origin. The angular and coordinate relationships of the original and transformed point positions are shown in Fig. 3-4. In this figure, r is the constant distance of the point from the origin, angle ϕis the original angular position of the point from the horizontal, and θ is the rotation angle. Using standard trigonometric identities, we can express the transformed coordinates in terms of angles θ and ϕas

$$x^1 = r\cos(\phi + \theta) = r\cos\phi\ \cos\theta - r\sin\phi\sin\theta \qquad (3\text{-}4)$$

$$y^1 = r\cos(\phi + \theta) = r\cos\phi\ \sin\theta + r\sin\phi\cos\theta$$

The original coordinates of the point in polar coordinates of the point in polar coordinates are

$$x = r\cos\phi,\ \ y = r\sin\phi \qquad (3\text{-}5)$$

Substituting expressions 3-5 into 3-4, we obtain the transformation equations for rotating a point at the position (x, y) through an angle θ about the origin:

$$x^1 = x\cos\theta - y\sin\theta$$

$$y^1 = x\sin\theta + y\cos\theta \qquad (3\text{-}6)$$

With a column-vector representation 3-2 for coordinate positions, we can write the rotation equations in the matrix form

$$\mathbf{P^1 = R \cdot P} \qquad (3\text{-}7)$$

where the rotation matrix is

$$\boldsymbol{R} = \begin{matrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{matrix} \quad (3\text{-}8)$$

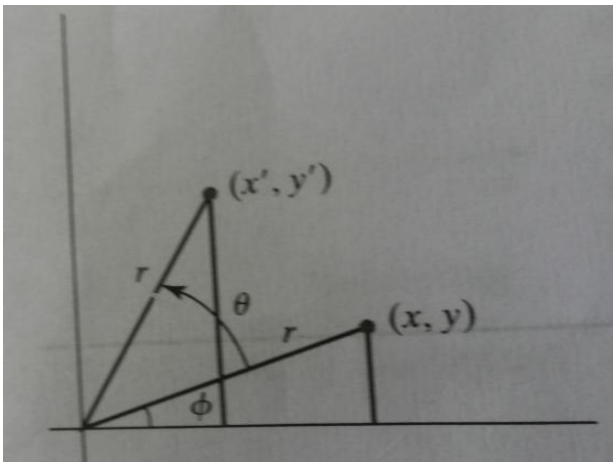


Figure 3-4 Rotating a point from position *(x, y)* through an angle θ about the rotation point *(x_r, y_r)*

A column vector representation for a coordinate position **P,** as in equations 3-2, is standard mathematical notation. However, early graphics systems sometimes used a row-vector representation for point positions. This changes the order in which the matrix multiplication for a rotation would be performed. But now, graphics packages such as OpenGL, Java, PHIGS, and GKS all follow the standard column-vector convention.

Rotation of a point about an arbitrary pivot position is illustrated in Fig.3-5. Using the trigonometric relationships indicated by the two right triangles in this figure, we can realize Equations. 3-6 to obtain the transformation equations for rotation of a point about any specified rotation position ***(x_r, y_r)***:

$x^1 = x_r + (x - x_r)\cos\theta - (y - y_r)\sin\theta$ (3-9) $y^1 = y_r + (x - x_r)\cos\theta + (y - y_r)\cos\theta$

These general rotation equations differ from equations 3-6 by the inclusion of additive terms, as well as the multiplicative factors on the coordinate values. The matrix expression 3-7 could be modified to include pivot coordinates by including the matrix addition of a column vector whose elements contain the additive (translational) terms in equations 3-9. There are better ways, however, to formulate such matrix equations, and in section 3-2 we discuss a more consistent scheme for representing the transformation equations.

As with translations, rotations are rigid-body transformations that move objects without deformation. Every point on an object is rotated through the same angle. A straight-line segment is rotated by applying the rotation equations 3-9 to each of the two endpoints and redrawing the

line between the new endpoint positions. A polygon is rotated by displacing each vertex using the specified rotation angle and then regenerating the polygon using the new vertices. We rotate a curve by repositioning the defining points for the curve and then redrawing it. A circle or an ellipse, for instance, can be rotated about a non-central pivot point by moving the center position through the arc that subtends the specified rotation angle. And we could rotate an ellipse about its center coordinates simply by rotating the major and minor axes.

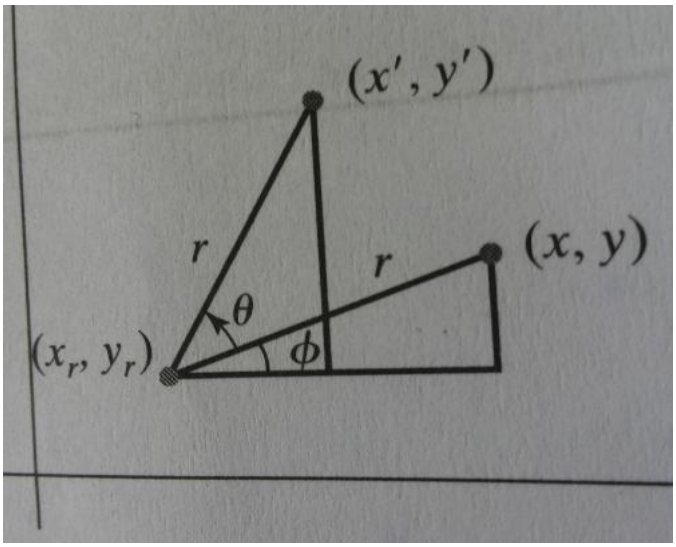


Figure 3-5 Rotating a point from position *(x,y)* to position *(x^1, y^1)* through an angle θ about rotation point *(x_r, y_r)*.

Two-Dimensional Scaling

To alter the size of an object, we apply **scaling** transformations. A simple two-dimensional scaling operation is performed by multiplying object positions ***(x, y)*** by **scaling factors** s_x and s_y to produce the transformed coordinates ***(x^1, y^1)***:

$$x^1 = x \cdot s_x, \qquad y^1 = y \cdot s_y \tag{3-10}$$

Scaling factors$s_x$scales an object in the *x* direction, while s_y scales in the y direction. The basic two-dimensional scaling equations 3-10 can also be written in the following matrix form.

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{3-11}$$

Or

$$P^1 = S \cdot P \tag{3-12}$$

Where **S** is the 2 by 2 scaling matrix in Equation 3-11.

Any positive values can be assigned to the scaling factors s_x and s_y. Values less than 1 reduce the size of objects; values greater than 1 produce enlargement. Specifying a value of for both s_x and s_y leaves the size of objects unchanged. When s_x and s_y are assigned the same value, a **uniform scaling** that is often used to design applications, where pictures are constructed from a few basic shapes that can be adjusted by scaling and positioning transformations (Fig. 3-6). In some systems, negative values can also be specified for the scaling parameters. This not only resizes an object, it reflects it about one or more the coordinate axes.

(a) (b)

Fig 3-6 Turning a square (a) into a rectangle (b) with scaling factors $s_x = 2$ and $s_y = 1$.

Objects transformed with Eq. 3-11 are both scaled and repositioned. Scaling factors with absolute values less than 1 move objects closer to the coordinate origin, while absolute values

greater than 1 move coordinate positions farther from the origin. Figure 3-7 illustrates scaling of a line by assigning the value 0.5 to both s_x and s_y in Eq. 3-11. Both the line length and the distance from the origin are reduced by a factor ½.

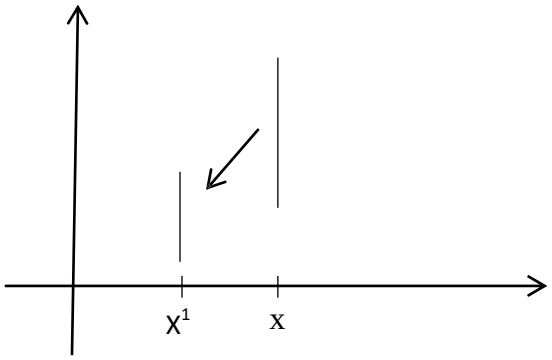


Figure 3-7 a single line scaled with Eq. 3-12 using $s_x = s_y = 0.5$ is reduced in size and moved closer to the coordinate origin.

We can control the location of an object by choosing a position, called the **fixed point,** that is to remain unchanged after the scaling transformation. Coordinates for the fixed point, (x_f, y_f), are often chosen at some object position, such as its centroid, but any other spatial position can be selected. Objects are now resized by scaling the distances between object points and the fixed point (Fig. 3-8). For a coordinate position (x, y), the scaled coordinates (x^1, y^1) are then calculated from the following relationships.

$$x^1 - x_f = (x - x_f)s_x, \qquad y^1 - y_f = (y - y_f)s_y \tag{3-13}$$

We can rewrite Eqs. 3-13 to separate the multiplicative and additive terms as

$$x1 = x \cdot s_x + x_f(1 - s_x), \qquad y^1 = y \cdot s_y + y_f(1 - s_y) \tag{3-14}$$

where the additive terms $x_f(1 - s_x)$ and $y_f(1 - s_y)$ are constants for all points in the object.

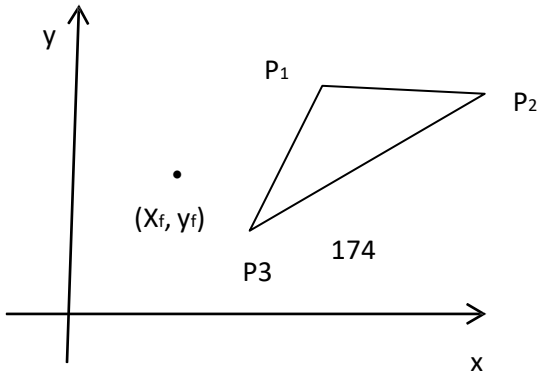


Figure 3-8 Scaling relative to s chosen fixed point (x_f, y_f). The distance from each polygon vertex point is scaled by transformation equations 3-13.

Including coordinates for a fixed point in the scaling equations in similar to including coordinates for a pivot in the rotation equations. We can set up a column vector whose elements are the constant terms in Eqs. 3-14, then add this column vector to the products **S . P** in Eq. 3-12. In the next section, we discuss a matrix formulation for the transformation equations that involves only matrix multiplication.

Polygons are scaled by applying transformations 3-14 to each vertex, then regenerating the polygon using the transformed vertices. For other objects, we apply the scaling transformation equations to the parameters defining the objects. To change the size of a circle, we can scale its radius and calculate the new coordinate positions around the circumference. And to change the size of an ellipse, we apply scaling parameters to its two axes and then plot the new ellipse positions about its coordinates.

7.3 Matrix Representation and Homogeneous Coordinates

Many graphics applications involve sequences of geometric transformations. An animation might require an object to be translated and rotated at each increment of the motion. In design and picture construction applications, we perform transformations, rotations, and scaling to fit the picture components into their proper positions. And the viewing transformations involve sequences of translations and rotations to take us from the original scene specification to the display on an output device. Here we consider how matrix representations discussed in the previous sections can be formulated so that such transformation sequences can be efficiently processed.

We have seen in section 3.1 that each of the three basic two-dimensional transformations (translation, rotation, scaling) can be expressed in the general matrix form.

$$\mathbf{P^1 = M_1 \cdot P + M_2} \quad (3\text{-}15)$$

With coordinate positions **P** and $\mathbf{P^1}$ represented as column vectors. Matrix $\mathbf{M_1}$ is a 2 by 2 array containing multiplicative factors, and $\mathbf{M_2}$ is a two-element column matrix containing translational terms. For translation, $\mathbf{M_1}$ is the identity matrix. For rotation or scaling, $\mathbf{M_2}$ contains with these equations, such as scaling followed by rotation then translation, we could calculate the transformed coordinates one step at a time. First, coordinate positions are scaled, and then these scaled coordinates are rotated, and finally combine the transformations so that the final coordinate positions are obtained directly from the initial coordinates, without calculating intermediate coordinate values. We can do this by reformulating Eq. 3-15 to eliminate the matrix addition operation.

Homogeneous Coordinates

Multiplicative and translational terms for a two-dimensional geometric transformation can be combined into a single matrix if we expand the representations to 3 by 3 matrices. Then we can use the third column of a transformation matrix for the translation terms, and all transformation equations can be expressed as matrix multiplications. But to do so, we also need to expand the matrix representation for a two-dimensional coordinate position to a three-element column matrix. A standard technique for accomplishing this is to expand each two-dimensional coordinate-position representation ***(x, y)*** to a three-element representation $\boldsymbol{(x_h, y_h, h)}$ called **homogeneous coordinates,** where the **homogeneous parameter** h is a nonzero values such that

$$x = \frac{x_h}{h}, \quad y = \frac{y_h}{h} \quad (3\text{-}16)$$

Therefore, a general two-dimensional homogeneous coordinate representation could also be written as ***(h.x. h.y, h).*** For geometric transformations, we can choose the homogeneous parameter ***h*** to be any nonzero value. Thus, there are an infinite number of equivalent homogeneous representations for each coordinate point ***(x, y)***. A convenient choice is simply to set ***h = 1***. Each two-dimensional position is then represented with homogeneous coordinates ***(x, y, 1)***. Other values for parameter ***h*** are needed, for example, in matrix formulations of threedimensional viewing transformations.

The term *homogeneous coordinates* is used in mathematics to refer to the effect of this representation on Cartesian equations. When a Cartesian point ***(x, y)*** is converted to a

homogeneous representations (x_h, y_h, h), equations containing x and y, such as $f(x, y) = 0$, become homogeneous equations in the three parameters x_h, y_h, h. This just means that if each of the three parameters is replaced by any value $v$times that parameter; the value $v$can be factored out of the equations.

Expressing positions in homogeneous coordinates allows us to represent all geometric transformation equations as matrix multiplications, which is the standard method used in graphics systems. Two-dimensional coordinate positions are represented with three-element column vectors, and two-dimensional transformation operations are expressed as 3 by 3 matrices.

Two-dimensional Transition Matrix

Using a homogeneous-coordinates approach, we can represent the equations for a twodimensional translation of a coordinate position using the following matrix multiplication.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (3\text{-}17)$$

This translation operation can be written in the abbreviated form

$$\mathbf{P}^1 = \mathbf{T}(t_x, t_y) \cdot \mathbf{P} \quad (3\text{-}18)$$

With $\mathbf{T}(t_x, t_y)$as the 3 by 3 translation matrix in Eq. 3-17 .in situations where there is no ambiguity about the translation parameters, we can simply represent the translation matrix as **T.**

Two-dimensional Rotation Matrix

Similarly, two-dimensional rotation transformation equations about the coordinate origin can be expressed in the matrix form

$$\begin{bmatrix} x' \\ y' \\ s \end{bmatrix} = \begin{bmatrix} \cos & -\sin & 0 \\ \sin & \cos & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (3\text{-}19)$$

Or as

$$\mathbf{P}^1 = \mathbf{R}(\theta) \cdot \mathbf{P} \quad (3\text{-}20)$$

The rotation transformation operator **R(θ)** is the 3 by 3 matrix in Eq. 3-19 with rotation parameter **θ.** We can also write this rotation matrix simply as **R.**

In some graphics libraries, a two-dimensional rotation function generates only rotations about the coordinate origin, as in Eq. 3-19. A rotation about any other pivot point must then be performed as a sequence of transformation operations. An alternative approach in a graphics package is to provide additional parameters in the rotation routine for the pivot-point coordinates. A rotation routine that includes a pivot-point parameter then sets up a general rotation matrix without the need to invoke a succession of transformation functions.

Two-Dimensional Scaling Matrix

Finally, a scaling transformation relative to the coordinate origin can now be expressed as the matrix multiplication

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (3\text{-}21)$$

Or

$$\mathbf{P}' = \mathbf{S}(s_x, s_y) \cdot \mathbf{P} \qquad (3\text{-}22)$$

The scaling operator $\mathbf{S}(s_x, s_y)$ is the 3 by 3 matrix in Eq. 3-21 with parameters s_x and s_y. And, in most cases, we can represent the scaling matrix simply as **S.**

Some libraries provide a scaling function that can generate only scaling with respect to the coordinate origin, as in Eq. 3-21. In this case, at scaling transformation relative to another reference position is handled as a succession of transformation operations. However, other systems do include a general scaling routine that can construct the homogeneous matrix for scaling with respect to as designed fixed point.

7.4 Two-Dimensional Composite transformations

Using matrix representations, we can set up a sequence of transformations as a **composite transformation matrix** by calculating the product of the individual transformations. Forming products of transformation matrices is often referred to as a **concatenation, or composition,** of matrices. Since a coordinate position is represented with a homogeneous column matrix, we must

premultiply the column matrix by the matrices representing any transformation sequence. And, since many positions in a scene are typically transformed by the same sequence, it is more efficient to first multiply the transformation matrices to form a single composite matrix. Thus, if we want to apply two transformations to point position **P,** the transformed location would be calculated as

$$\mathbf{P}_1 = \mathbf{M}_2 . \mathbf{M}_1 . \mathbf{P}$$

$$= \mathbf{M} . \mathbf{P} \qquad (3\text{-}23)$$

The coordinate position is transformed using the composite matrix **M,** rather than applying the individual transformations $\mathbf{M}_1$and then $\mathbf{M}_2$**.**

Composite Two-Dimensional Translations

If two successive translation vectors (t_{1x}, t_{1y}) and (t_{2x}, t_{2y}) are applied to a two-dimensional coordinate position **P,** the final transformed location $\mathbf{P}^1$is calculated as

$$\begin{aligned} \mathbf{P}' &= \mathbf{T}(t_{2x},t_{2y}) \cdot \{\mathbf{T}(t_{1x},t_{1y}) \cdot \mathbf{P}\} \\ &= \{\mathbf{T}(t_{2x},t_{2y}) \cdot \mathbf{T}(t_{1x},t_{1y})\} \cdot \mathbf{P} \end{aligned} \qquad (3\text{-}24)$$

Where **P** and $\mathbf{P}^1$are represented as three-element, homogeneous-coordinate column vectors. We can verify this result by calculating the matrix product for the two associative groupings. Also, the composite transformation matrix for this sequence of translation is

$$\begin{bmatrix} 1 & 0 & t_{2x} \\ 0 & 1 & t_{2y} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & t_{1x} \\ 0 & 1 & t_{1y} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_{1x} + t_{2x} \\ 0 & 1 & t_{1y} + t_{2y} \\ 0 & 0 & 1 \end{bmatrix} \qquad (3\text{-}25)$$

Or

$$\mathbf{T}(t_{2x},t_{2y}) \cdot \mathbf{T}(t_{1x},t_{1y}) = \mathbf{T}(t_{1x} + t_{2x}, t_{1y} + t_{2y}) \qquad (3\text{-}26)$$

which demonstrates that two successive translations are additive.

Composite Two-Dimensional Rotations

Two successive rotations applied to a point **P** produce the transformed position

$$P1 = R(\theta_2) .\{ R(\theta_1) . P\}$$

$$= \{ R(\theta_2) . R(\theta_1)\} . P \quad (3\text{-}27)$$

By multiplying the two rotation matrices, we can verify that two successive rotations are additive:

$$R(\theta_1) \cdot R(\theta_2) = R(\theta_1 + \theta_2) \quad (3\text{-}28)$$

So that the final rotated coordinates of a point can be calculated with the composite rotation matrix as

$$P' = R(\theta_1 + \theta_2) \cdot P \quad (3\text{-}29)$$

Composite Two-Dimensional Scalings

Concatenating transformations matrices for two successive scaling operations in two dimensions produces the following composite scaling matrix.

$$\begin{bmatrix} s_{2x} & 0 & 0 \\ 0 & s_{2y} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_{1x} & 0 & 0 \\ 0 & s_{1y} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_{1x} \cdot s_{2x} & 0 & 0 \\ 0 & s_{1y} \cdot s_{2y} & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3\text{-}30)$$

Or

$$S(s_{2x}, s_{2y}) \cdot S(s_{1x}, s_{1y}) = S(s_{1x} \cdot s_{2x}, s_{1y} \cdot s_{2y})$$

The resulting matrix in this case indicates that successive scaling operations are multiplicative. That is, if we were to triple the size of an object twice in succession, the final size would be nine times that of the original.

General Two-Dimensional pivot-Point Rotation

When a graphics package provides only a rotate function with respect to the coordinate origin, we can generate a two-dimensional rotation about any other pivot-point (x_r, y_r) by performing the following sequence of translate-rotate-translate operations.

(1) Translate the object so that the pivot-point position is moved to the coordinate origin.

(2) Rotate the object about the coordinate origin.

(3) Translate the object so that the pivot point is returned to its original position.

This transformation sequence is illustrated in Fig. 3-9. The composite transformation matrix for this sequence is obtained with the concatenation

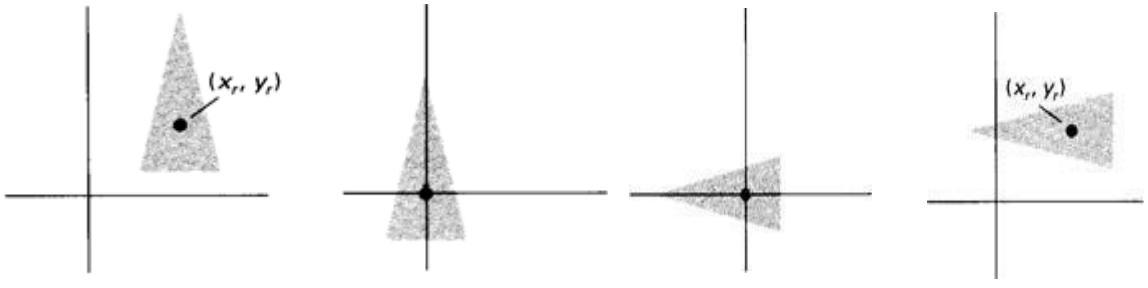
$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix} \quad (3\text{-}31)$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta)+y_r\sin\theta \\ \sin\theta & \cos\theta & y_r(1-\cos\theta)-x_r\sin\theta \\ 0 & 0 & 1 \end{bmatrix}$$

Which can be expressed in the form,

$$\mathbf{T}(x_r, y_r)\cdot\mathbf{R}(\theta)\cdot\mathbf{T}(-x_r, -y_r) = \mathbf{R}(x_r, y_r, \theta) \quad (3\text{-}32)$$

Where $\mathbf{T}(-x_r, -y_r) = \mathbf{T}^{-1}(-x_r, -y_r) = \mathbf{R}(x_r, y_r, \theta)$. In general, a rotate function in a graphics library could be structured to accept parameters for pivot-point coordinates, as well as rotation angle, and to generate automatically the rotation matrix of Eq. 3-31



(a) Original position Of Object and Pivot-point to position (x_r, y_r)

(b) Translation of Object pivot-point (x_r, y_r) is at Origin

(c) Rotation about Origin

(d) Translated of Object so that pivot-point is returned

Figure 3-9 A transformation sequence for rotating an object about a specified pivot point using the rotation matrix R (θ) of transformation 3-19.

General Two-Dimensional Fixed-Point Scaling

Figure 3-10 illustrates a transformation sequence to produce a to-dimensional scaling with respect to a selected fixed position **(x_f, y_f),** when we have a function that can scale relative to the coordinate origin only. This sequence is

(1) Translate the object so that the fixed point coincides with the coordinate origin.

(2) Scale the object with respect to the coordinate origin.

(3) Use the inverse of the translation in step (1) to return the object to its original position.

Concatenating the matrices for these three operations produces the required scaling matrix:

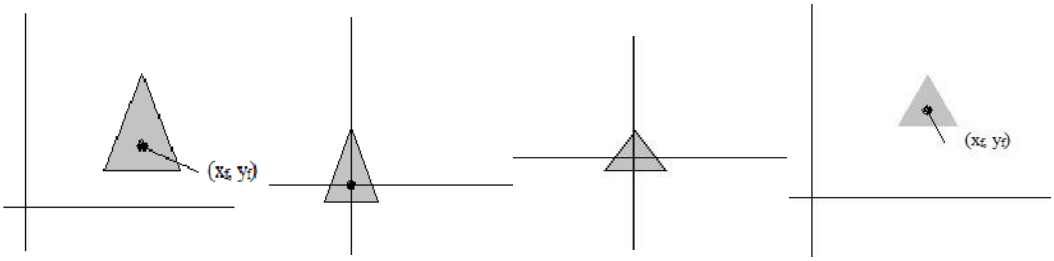
$$\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix} \tag{3-33}$$

$$= \begin{bmatrix} s_x & 0 & x_f(1-s_x) \\ 0 & s_y & y_f(1-s_y) \\ 0 & 0 & 1 \end{bmatrix}$$

Or

$$\mathbf{T(x_f, y_f) \cdot S(s_x, s_y) \cdot T(-x_f, -y_f) = S(x_f, y_f, s_x, s_y)} \tag{3-34}$$

This transformation is automatically generated in systems that provide a scale function that accepts coordinates for the fixed point.



(a) Original positionof Object and Fixed Point

(b) Translate Object so that Fixed Point (x_f, y_f) is at Origin

(c) Scale Objectwith respect to Origin

(d) Translate object so that the Fixed point to position (x_f, y_f)

Figure 3-10 A transformation sequence for scaling an object with respect to a specified fixed position using the scaling matrix S (s_x, s_y) of transformation 3-21.

General Two-Dimensional Scaling Directions

Parameters s_x and s_y scale objects along the *x* and *y* directions. We can scale an object in other directions by rotating the object to align the desired scaling directions with the coordinate axes before applying the scaling transformations.

Suppose we want to apply scaling factors with values specified by parameters s_1 and s_2 in the directions shown in Fig 3-11. To accomplish the scaling without changing the orientation of the object, first perform so that the directions for s_1 and s_2 coincide with the x and y axes, respectively. Then the scaling transformation $S(s_1, s_2)$is applied, followed by an opposite rotation to return points to their original orientations. The composite matrix resulting from the product of these three transformations is

$$R^{-1}(\theta)\,S(s_1,s_2)\,R(\theta) = \begin{pmatrix} s_1\cos^2\theta + s_2\sin^2\theta & (s_2 - s_1)\cos\theta\sin\theta & 0 \\ (s_2 - s_1)\cos\theta\sin\theta & s_1\sin^2\theta + s_2\cos^2\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (3\text{-}35)$$

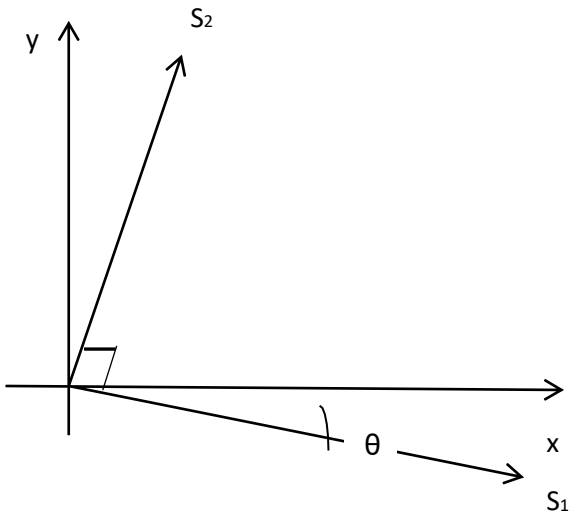


Figure 3-11 Scaling parameters s_1 and s_2 along orthogonal directionsdefined by the angular displacement θ.

Other Two-Dimensional Transformations

Basic transformations such as translation, rotation, and scaling are standard components of graphics libraries. Some packages provide a few additional transformations that are useful in certain applications. Two such transformations are reflection and shear.

Reflection

A transformation that produces a mirror image of an object is called a **reflection.** For a twodimensional reflection, this image is generated relative to an **axis of reflection** by rotating the object 180^0about the reflection axis. We can choose an axis of reflection in the ***xy***plane, the rotation path about this axis is in Plane perpendicular to the ***xy***plane. For reflection axes that are perpendicular to the ***xy***plane, the rotation path is in the ***xy***plane. Following is the example of one common reflection.

Reflection about the line y = 0 (the x axes) is accomplished with the transformation matrix

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3\text{-}36)$$

This transformation retains x values, but ―flips‖the y values of coordinate positions. The resulting orientation of an object after it has been reflected about the ax axis in Fig. 3-12. To envision the rotation transformation path for this reflection, we can think of the flat object moving out of the ***xy*** plane and rotating 180°through three-dimensional space about the x axis and back into the ***xy*** plane on the other side of the x axis.

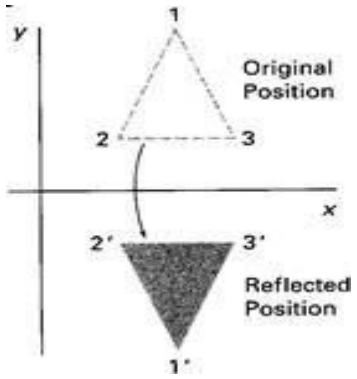


Figure 3-12 Reflection of an object about the x axis.

Shear

A transformation that distorts the shape of an object such that the transformed shape appears as if the object were composed of internal layers that had been caused to slide over each other is called a **shear.** Two common shearing transformations are those that shift coordinate x values and those that shift y values.

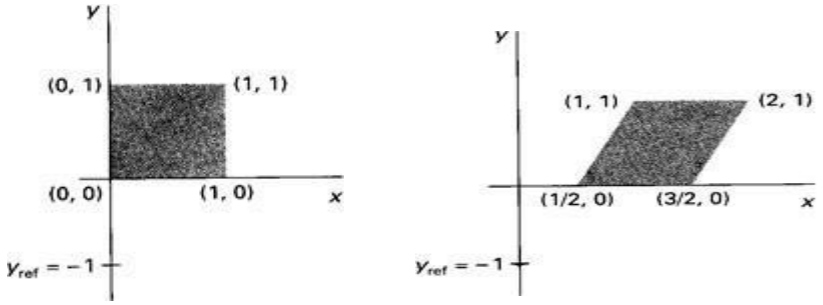
A x-direction shear relative to the x axis is produced with the transformation matrix

$$T_{shx} = \begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3\text{-}37)$$

which transforms coordinate positions as,

$$x' = x + sh_x \cdot y, \qquad y' = y$$

Any real number can be assigned to the shear parameter sh_x. A coordinate position (x, y) is then shifted horizontally by an amount proportional to its perpendicular distance (y value) from the x axis. Setting parameter sh_x to the value 2, for example, changes the square in Fig. 3-13 into a parallelogram. Negative values for sh_x shift coordinate positions to the left.



(a) (b)

Figure 3-13 A unit square (a) is coverted to a parallegram (b) using the *x* direction shear matrix 3-37 with $sh_x = 2$.

5. Summary

The basic geometric transformations are translation, rotation, and scaling. Translation moves an object in a straight-line path from one position to another. Rotation moves an object from one position to another along a circular path around a specified rotation axis. For two-dimensional applications, the rotation path is in the *xy*plane about an axis that is parallel to the z axis. Scaling transformations change the dimensions of an object relative to a fixed position.

Composite transformations are formed as matrix multiplications of translation, rotation, scaling and other transformations. We can use combinations of translation and rotation for animation applications, and we can use combinations of rotation and scaling to scale objects in any specified direction. In general, matrix multiplications are not commutative. We obtain different results, for example, if we change the order of a translate-rotate sequence. A transformation sequence involving only translations and rotations is a rigid-body transformation, since angles

and distances are unchanged. Also, the upper-left sub-matrix of a rigid-body transformation is an orthogonal matrix.

Other geometric transformations include reflections and shears. Reflections are transformations that rotate an object 180° about a reflection axis. This produces a mirror image of the object with respect to axis. Shear transformations distort the shape of an object by shifting one or more coordinate values by an amount proportional to the distance from a shear reference line.

6. Keywords

Geometric transformations, translation, Scaling, rotation, translation vector, translation distance, Pivot point, uniform scaling, differential scaling, homogeneous coordinates, Composite transformation, Reflection, Shear.

7. Exercises

1) Explain briefly the Basic Two-dimensional Geometric transformations.
2) Explain Two-dimensional Rotation matrix.
3) Explain briefly Two-Dimensional Composite Transformations.
4) Explain briefly Two-Dimensional pivot point Rotation.
5) Explain briefly Two-Dimensional Fixed Point Scaling.
6) Explain Reflection and Shear in Two-Dimensional transformations.

8. References

1) Ralf Steinmentz, klara Naestedt: Multimedia Fundamentals: vol 1-media Coding and Content processing,2nd edition, PHI, Indian Reprint 2008.
2) Prabhat K. Andleigh, Kiran Thakrar, Multimedia System Design, PHI,2003
3) Ze-Nian Li-mark S Drew, Fundamentals of multimedia, PHI, New Drelhi, 2011.
4) Donald hearn and M. Pauline Baker, Computer graphics, 3rd edition, Pearson.

UNIT-8 Advanced Methods and Algorithms for Viewing

Structure

14. Learning objectives

After studying this unit you will be able to,

- Discuss ―The Two-Dimensional Viewing Pipeline‖,
- Describe ―The clipping Window‖,
- Differentiate ―Normalization and Viewport Transformations‖,
- Explain ―OpenGL Two-Dimensional Viewing Functions‖, ―Clippingalgorithms‖,
- Explain ―Two-Dimensional point Clipping‖ and
- Give an account on ―Two-Dimensional Line Clipping algorithms‖.

1. Introduction

~~Typically, a graphics package allows a user to specify which part of picture is to be displayed~~ and where that part is to be placed on the display device. Any convenient Cartesian coordinate system, referred to as the world-coordinate reference frame, can be used to define the picture. For a two-dimensional picture, a view is selected by specifying a region of the ***xy*** plane that contains the total picture or any part of it. A user can select a single area for display, or several areas could be selected for simultaneous display or for an animated panning sequence across a

scene. The picture parts within the selected area are then mapped onto the specified areas of the device coordinates. When multiple view areas are selected, these areas can be placed in separate display locations, or some areas could be inserted into other, larger display areas. Twodimensional viewing transformations from world to device coordinates involve translation, rotation, and scaling operations, as well as procedures for deleting those parts of the picture that are outside the limits of a selected scene area.

8.2 The Two-Dimensional Viewing Pipeline

A section of a two-dimensional scene that is selected for display is called a **clipping window,** because all parts of the scene outside the selected section are ―clipped‖ off. The only part of the scene that shows up on the screen is what is inside the clipping window. Sometimes the clipping window is alluded to as the *world window* or the *viewing window.* And, at one time, graphics systems referred to the clipping window simply as ―the window‖, but there are now so many windows in use on computers that we need to distinguish between them. For example, a windowmanagement system can create and manipulate several areas on a video screen, each of which is called ―awindow‖, for the display of graphics and text (**Fig 4.1**). So, we will always use the term *clipping window* to refer to a selected section of a scene that is eventually converted to pixel patterns within display window on the monitor. Graphics packages allow us also to control the placement within the display window using another ―window‖called **viewport.** Objects inside the clipping window are mapped to the viewport, and it is the viewport that is then positioned within the display window. The clipping window selects *what* we want to see; the viewport indicates *where* it is to be viewed on the output device.

By changing the position of a viewport, we can view objects as different positions on the display area of an output device. Multiple viewports can be used to display different sections of a scene at different screen positions. Also, by varying the size of viewports, we can change the size and proportions of displayed objects. We achieve zooming effects by successively mapping differentsized clipping windows onto a fixed-size viewport. As the clipping windows are made smaller, we zoom in on some part of a scene to view details that are not shown with larger clipping windows. Similarly, more overview is obtained by zooming out from a section of a scene with successively larger clipping windows. And panning effects are achieved by moving a fixed-size clipping window across the various objects in a scene.

Figure 4-1 A video screen showing multiple, simultaneous display windows.

Usually, clipping windows and viewports are rectangles in standard position, with the rectangle edges parallel to the coordinate axes. Other window or viewport geometries, such as general polygon shapes and circles, are used in some applications, but these shapes take longer to process. We first consider only rectangular viewports and clipping windows, as illustrated in Figure 4-2

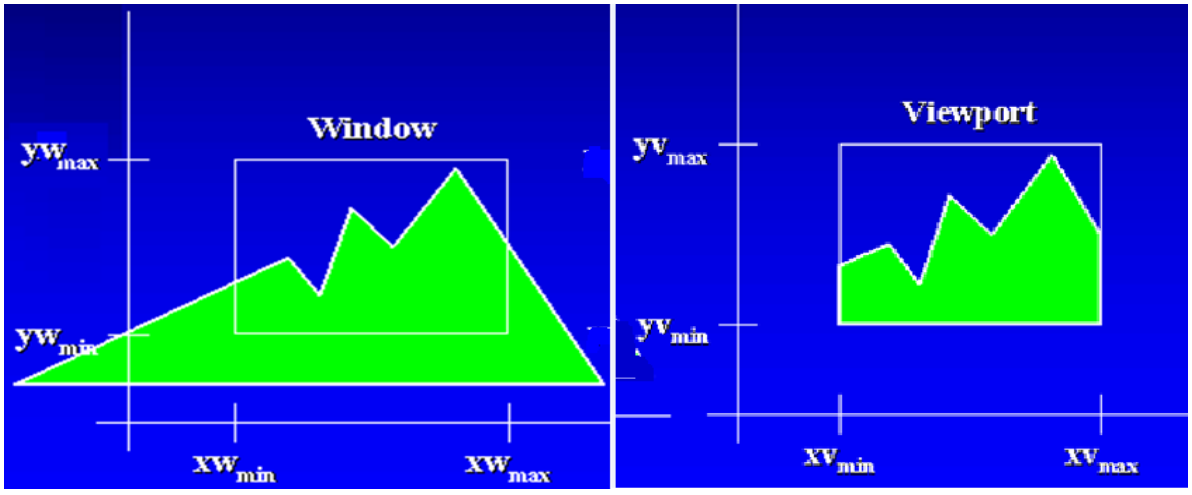


Figure 4-2 A clipping window and associated viewport,specified as rectangles aligned with the coordiate axis.

The mapping of a two-dimensional, world-coordinate scene description to device coordinates is called a **two-dimensional viewing transformations.** Some times this transformation is simply referred to as the *window-to-viewport transformation* or *windowing transformation.* But, in general, viewing involves more than just the transformation from clipping-window coordinates to viewpport coordinates. In anology with three-dimensional viewing, we can describe the steps for two-dimensional viewing as indicated in Fig 4.3. Once a world-coordinate scene has been constructed, we could set up a separate two-dimensional, **viewing-coordinate reference frame**

for specifying the clipping window. But the clipping window is often just defined in world coordinates, so that viewing coordinates for two-dimensional applications are the same as world coordinates.

To make the viewing process independent of the requirements of any output device, graphics systems convert object descriptions to normalized coordinates in the range from 0 to 1, and others use a normalized range from -1 to 1. Depending upon the graphics library in use, the viewport is defined in normalized coordinates or in screen coordinates after the normalization process. At the final step of the viewing transformation, the contents of the viewport are transferred to positions within the display window.

Clipping is usually performed in normalized coordinates. This allows us to reduce computations by first concatenating the various transformation matrices. Clipping procedures are of fundamental importance in computer graphics. They are used not only in viewing transformations, but also in window-manager systems, in painting and drawing packages to erase picture sections, and in many other applications.

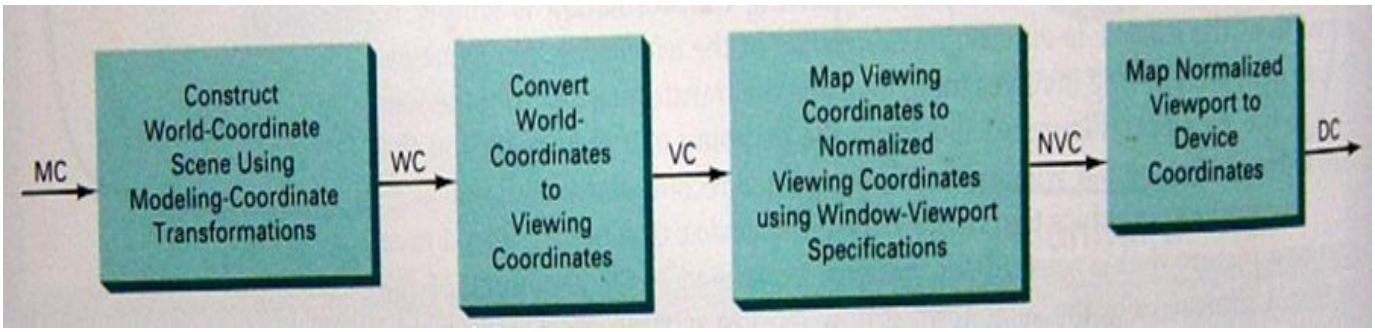


Figure 4-3 Two-dimensional viewing-transformation pipeline.

8.3 The Clipping Window

To achieve a particular viewing effect in an application program, we could design our own clippling window with any shape, size, and orientation we choose. For example, we might like to use a star pattern, or an ellipse, or a figure with spline boundaries as a clipping window. But clipping a scene using a concave polygon or a clipping window with nonlinear boundaries requires more processing than clipping against a rectangle. We need to perform more computations to determine where an object intersects a circle than to find out where it intersects a straight line. And the simplest window edges to clip against are straight lines that are parallel to

the coordinates axes. Therefore, graphics packages commonly allow only rectangular clipping windows aligned with the *x* and *y* axes.

If we want some other shape for a clipping window, then we must implement our own clipping and coordinate-transformation algorithms. Or we could just edit the picture to produce a certain shape for the display frame around the scene. For example, we could trim the edges of a picture with any desired pattern by overlaying polygons that are filled with the background color. In this way, we could generate any desired border effects or even put interior holes in the picture.

Reactangular clipping windows in standardposition are easily defined by giving the coordinates of two opposite corners of each rectangle. If we would like to get a rotated view of a scene, we could either define a rectangular clipping window in a rotated viewing-coordinate frame or, equivalently, we could rotate the world-coordinate scene. Some systems provide options for selecting a rotated, two-dimensional viewing frame, but usully the clipping window must be specified in world-coordinates.

Viewing-Coordinate Clipping Window

A general approach to the two-dimensional viewing transformation is to set up a *viewingcoordinate system* within the world-coordinate frame. This viewing frame provides a reference for specifying a rectangular clipping window with any selected orientation and position.,(Fig.4.4). To obtain a view of the world-coordinate scene as determined by the clipping window of Fig 4.4, we just need to transfer the scene description to viewing coordinates. Although many graphics packages do not provide functions for specifying a clipping window in a two-dimensional viewing-coordinate system, this is the standard approach for defining a clipping region for a three-dimensional scene.

We choose an origin for two-dimensional viewing-coordinate frame at world position $\mathbf{P_0} = (x_0, y_0)$, **and** we can establish the orientation using a world vector **V** that defines the y_{view} direction. Vector **V** is called the two-dimensional **view up vector.** An alternative method for specifying the orientation of the viewing frame is to give a rotation angle, we can then obtain the view up vector. Once we have established the parameters that define the viewing-coordinate frame, we use the procedures from the previous section to transform the scene description to the viewing system. This involves a sequence of transformations equivalent to superimposing the viewing frame on the world frame.

The first step in the transformation sequence is to translate the viewing origin to the world origin. Next, we rotate the viewing system to align it with the world frame. Given the orirntation vector **V,** we can calculate the components of unit vectors **v** = (v_x, v_y) and **u** = $(u_x, u_y)$for the y_{view} and x_{view} axes, respectively. These unit vectors are used to form the first and second rows of the rotation matrix **R** that alligns the viewing $x_{view}y_{view}$ axes with the world x_wy_w axes.

Object positions in world coordinates are then converted to viewing coordinates with the composite two-dimensional trasnformation matrix.

Where **T** is the translation matrix that takes the viewing origin $\mathbf{P_0}$to the world origin, and **R** is the rotation matrix that rotates the viewing frame of reference into coincidence with the worldcoordinate system. Figure 4-5 illustrates the steps in this coordinate transformation.

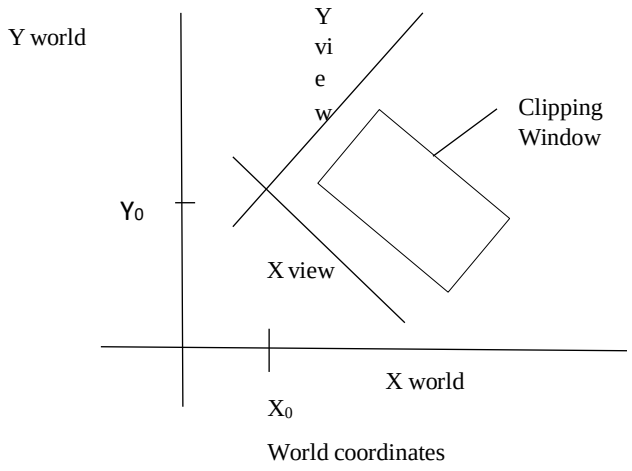
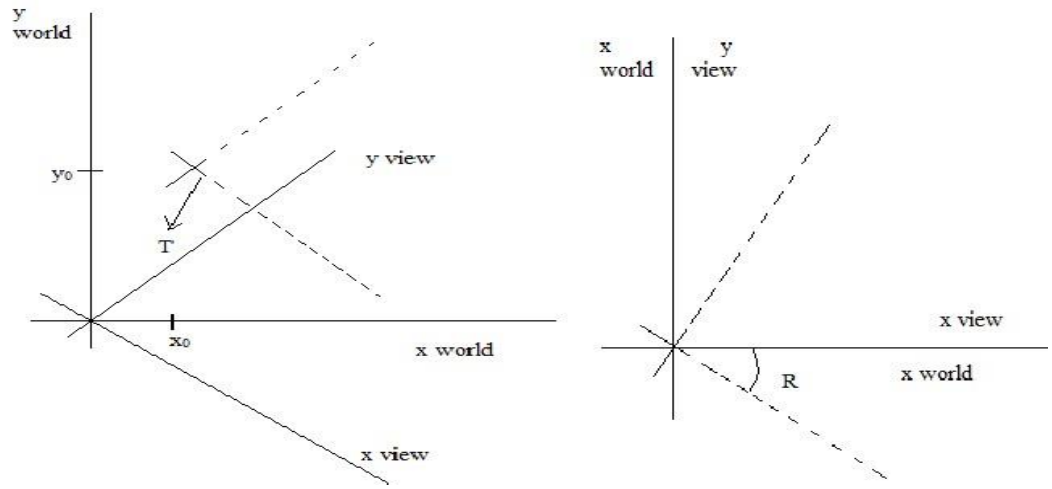


Figure 4-4 A rotated clipping window defined in viewing coordinates.



(a) (b)

Figure 4-5 A viewing coordinate frame is moved into coincidence with the world frame by (a) applying a translation matrix T to move the viewing origin to the world origin, then (b) appying a rotation matrix R to align the axes of the two systems

World-Coordinate Clipping Window

A routine for defining a standard, rectangular clipping window in world coordinates is typically provided in a graphics-programming library. We simply specify two world-coordinate positions, which are then assigned to the two opposite corners of a standard rectangle. Once the clipping window has been established, the scene description is processed through the viewing routines to the output device.

If we want to obtain a rotated view of a two-dimensional scene, as discussed in the previous section, we perform exactly the same steps as described there, but without considering a viewing frame of reference. Thus, we simply rotate (and possibly translate) objects to a desired position and set up the clipping window- all in world coordinates. As an example, we could display a rotated view of a triangle in Fig. 4-6 (a) by rotating it into the position we want and setting up a standard clipping rectangle. In analogy with the coordinate transformation described in the previous section, we could also translate the triangle to the world origin and define a clipping window around the triangle. In that case, we define an orientation vector and choose a reference point such as the triangle‘s centroid. Then we translate the reference point to the world origin triangle in the desired orientation, we can use a standard clipping window in world coordinates to capture the view of the rotated triangle. The transformed position of the triangle and the selected clipping are shown in Fig. 4-6 (b).

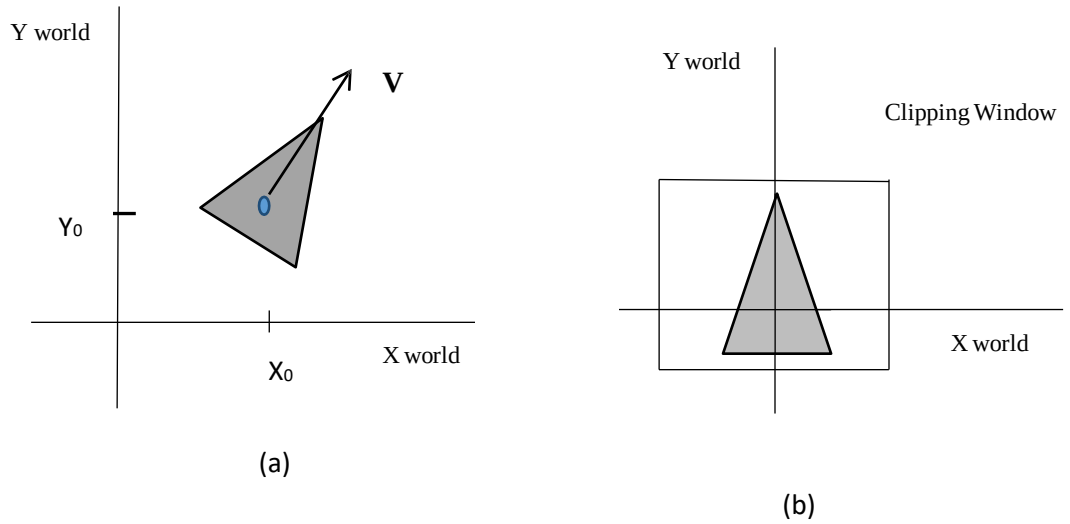


Figure 4-6 a triangle (a) with a selected reference point and orientation vector, is translated and rotated to position (b) within a clipping window.

8.4 Normalization and Viewport Transformation

With some graphics packages, the normalization and window-to-viewport transformation combined into one operation. In this case, the viewport coordinates are often given in the range from 0 to 1 so that the viewport is positioned within a unit square. After clipping, the unit square containing the viewport is mapped to the output display device. In other systems, the normalization and clipping routines are applied before the viewport transformation. For these systems, the viewport boundaries are specified in screen coordinates relative to the displaywindow position.

Mapping the Clipping Window into a Normalized Viewport

To illustrate the general procedures for the normalization and viewport transformations, we first consider a viewport defined with normalized coordinate values between 0 and 1. Object descriptions are transferred to this normalized space using a transformation that maintains the same relative placement of a point in the viewport as it had in the viewport as it had in the clipping window. If a coordinate position is at the center of the clipping window, for instance, it would be mapped to the center of the viewport. Fig. 4-7 illustrates this window-to-viewport

mapping. Position (x_w, y_w) in the clipping window is mapped into position (x_v, y_v) in the associated viewport

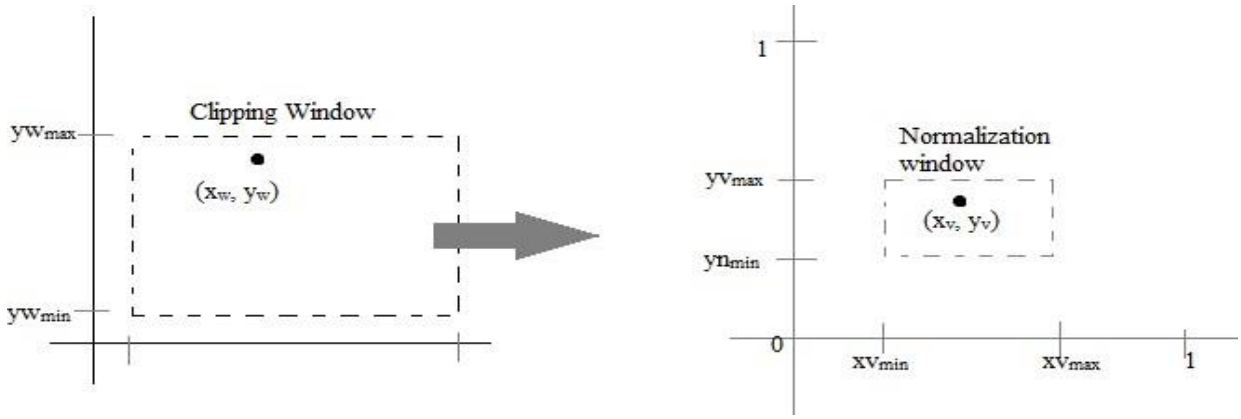


Figure 4-7 a point (x_w, y_w) in a world-coordinate clipping window is mapped to viewpoint coordinates (x_v, y_v), within a unit square, so that the relative positions of the two points in their respective rectangle are the same.

To transform the world-coordinate point into the same relative position within the viewport, we require that,

$$\frac{x_v - x_{v\min}}{x_{v\max} - x_{v\min}} = \frac{x_w - x_{w\min}}{x_{w\max} - x_{w\min}} \qquad \frac{y_v - y_{v\min}}{y_{v\max} - y_{v\min}} = \frac{y_w - y_{w\min}}{y_{w\max} - y_{w\min}} \tag{4-2}$$

Solvingthese expressions for the viewport position *(xv, yv)*, we have

$$X_v = s_x x_w + t_x \qquad Y_v = s_y y_w + t_y \tag{4-3}$$

Where the scaling factors

$$s_x = \frac{x_{v\max} - x_{v\min}}{x_{w\max} - x_{w\min}} \qquad s_y = \frac{y_{v\max} - y_{v\min}}{y_{w\max} - y_{w\min}} \tag{4-4}$$

$$t_x = x_{v\min} - x_{w\min} s_x \qquad t_y = y_{v\min} - y_{w\min} s_y \tag{4-5}$$

$$t_x = x_{wmax} x_{vmin} - x_{wmin} x_{vmax}$$

$$t_y = y_{wmax} y_{vmin} - y_{wmin} y_{vmax}$$

$$= \frac{x_{wmax} - x_{wmin}}{y_{wmax} - y_{wmin}} = s_y$$

Since we are simply mapping world-coordinate positions into a viewport that is positioned near the world origin, we can also derive Eqs. 4-3 using any transformation sequence that converts the rectangle for clipping window into the viewport rectangle. For example, we could obtain the transformation from world coordinates to viewport coordinates with the sequence

(1) Scale the clipping window to the size of the viewport using a fixed-point position (xw_{min}, yw_{min})

(2) Translate (xw_{min}, yw_{min}) to (xv_{min}, yv_{min}).

The scaling transformation in step (1) can be represented with the two-dimensional matrix

$$S = \begin{bmatrix} sx & 0 & sxmin(1-sx) \\ 0 & sy & ywmin(1-sy) \\ 0 & 0 & 1 \end{bmatrix} \quad (4\text{-}6)$$

Where s_x and s_y are the same as in Eqs. 4-4. The two-dimensional matrix representation for the translation of the lower-left corner of the clipping window to the lower-left viewport corner is

$$T = \begin{bmatrix} 1 & 0 & xv_{min} - xw_{min} \\ 0 & 1 & yv_{min} - yw_{min} \\ 0 & 0 & 1 \end{bmatrix} \quad (4\text{-}7)$$

And

$$M_{window,\ normviewp} = T = \begin{bmatrix} sx & 0 & tx \\ 0 & sy & ty \\ 0 & 0 & 1 \end{bmatrix} \quad (4\text{-}8)$$

This gives us the result as in Eqs. 4.3,any other clipping-window reference point, such as the topright corner or the window center, could be used for the scale-translate operations. Or, we could first translate any clipping-window position to the corresponding location in the viewport, and then scale relative to that viewport location.

The window-to-viewport transformation maintains the relative placement of object descriptions. An object inside the clipping window is mapped to a corresponding position inside the viewport. Similarly, an object outside the clipping window is outside the viewport.
Relative proportions of objects, on the other hand, are maintained only if the aspect ratio of the viewport is the same as the aspect ratio of the clipping window. in other words, we keep the same object proportions if the scaling factors s_x and $s_y$are the same. Otherwise, world objects will

be stretched or contracted in either the *x* and *y* directions (or both) when displayed on the output device.

The clipping routines can be applied using either the clipping-window boundaries or the viewport boundaries. After clipping, the normalized coordinates are transformed into device coordinates. And the unit square can be mapped onto the output device using the same procedures as in the window-to-viewport transformation, with the area inside the unit square transferred to the total display area of the output device.

Mapping the Clipping Window into a Normalized Square

Another approach to two-dimensional viewing is to transform the clipping window into a normalized square, clip in normalized coordinates, and then transfer the scene description to a viewport specified in screen coordinates. This transformation is illustrated in Fig. 4-8 with normalized coordinates in the range from 0 to 1. The clipping algorithms in this transformation sequence are now standardized so that objects outside the boundaries x = ±1 are detected and removed from the scene description. At the final step of the viewing transformation, the objects in the viewport are positioned within the display window.

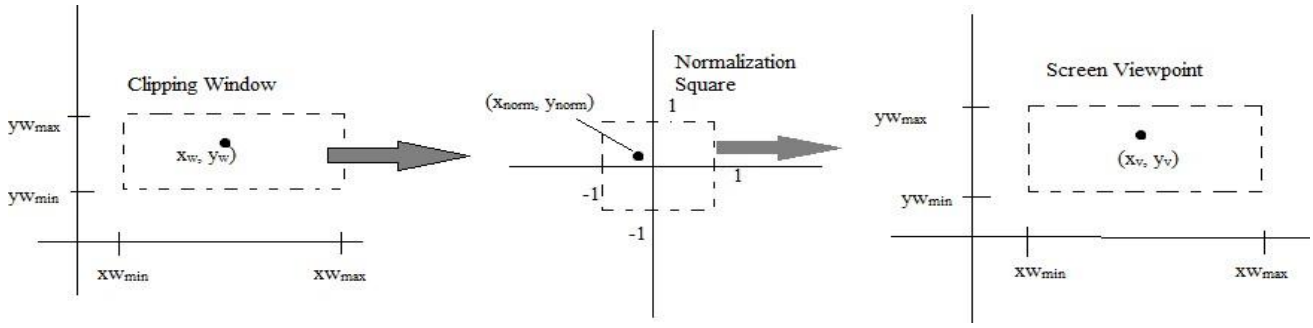


Figure 4-8 a point *(x_w, y_w)* in a clipping window is mapped to a normalized coordinate position *(x_{norm}, y_{norm})*, then to a screen-coordinate position (x_v, y_v) in a viewport. Objects are clipped against the normalization square before the transformation to viewport coordinates.

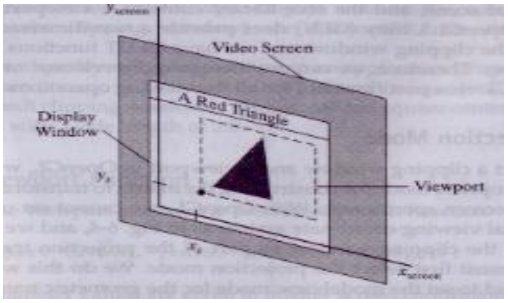


Figure 4-9 A viewport at coordinate position (x_s, y_s) within a display window.

We transfer the contents of the displaying window into the normalization square using the same procedures as in the window-to-viewport transformation. The matrix for the normalization transformation is obtained from Eq. 4-8 by substituting -1 for $xv_{min}$and $yv_{min}$and substituting +1 for $xv_{max}$and yv_{max}. Making these substitutions in the expressions for t_x, t_y, s_x, and s_y, we have,

$$M_{window,\ normview} = \begin{bmatrix} \frac{2}{xw_{max} - xw_{min}} & 0 & -\frac{xv_{max} + xv_{min}}{xw_{max} - xw_{min}} \\ 0 & \frac{2}{yw_{max} - yw_{min}} & -\frac{yw_{max} + yw_{min}}{yw_{max} - yw_{min}} \\ 0 & 0 & 1 \end{bmatrix} \quad (4\text{-}9)$$

Similarly, after the clipping algorithms have been applied, the normalized square with edge length equal to 2 is transformed into a specified viewport. This time, we get the transformation matrix from Eq. 4-8 by substituting -1 for xw_{min} and yw_{min} and substituting +1 for $yw_{max}$and yw_{max}:

$$M_{normsquare,\ window} = \begin{bmatrix} \frac{xw_{max} - xw_{min}}{2} & 0 & \frac{xv_{max} + xv_{min}}{2} \\ 0 & \frac{yw_{max} - yw_{min}}{2} & \frac{yw_{max} + yw_{min}}{2} \\ 0 & 0 & 1 \end{bmatrix} \quad (4\text{-}10)$$

The last step in the viewing process is to position the viewport area in the display window.

Typically, the lower-left corner of the viewport is placed at a coordinate position specified relative to the lower-left corner of the display window. Fig. 4-9 demonstrates the positioning of a viewport within a display window.

As before, we maintain the initial proportions of objects by choosing the aspect ratio of the viewport to be the same as the clipping window. Otherwise, objects will be stretched or contracted in the *x* or y directions. Also, the aspect ratio of the display window can affect the proportions of objects. If the viewport is mapped to the entire area of the display window and the size of the display window is changed, objects may be distorted unless the aspect ratio of the viewport is also adjusted.

Display of Character Strings

Character strings can be handled in one of two ways when they are mapped through the viewing pipeline to a viewport. The simplest mapping maintains a constant character size. This method could be employed with bitmap character patterns. But outline fonts could be transformed the same as other primitives; we just need to transform the defining positions for the line segments in the outline character shapes. Algorithms for determining the pixel patterns for the transformed characters are then applied when the other primitives in the scene are processed.

Split-Screen Effects and Multiple output devices

By selecting different clipping windows and associated viewports for a scene, we can provide simultaneous display of two or more objects, multiple picture parts, or different views of a single scene. And we can position these views in different parts of a single display window or in multiple display windows on the screen. In a design application, for example, we can display a wire-frame view of an object in one viewport, while displaying a fully rendered view of the object in another viewport. And we could list other information or menus in a third viewport.

It is also possible that two or more output devices could be operating concurrently on a particular system, and we can set up a clipping-window/viewport pair for each output device. A mapping to a selected output device is sometimes referred to as a **workstation transformation.** In this case, viewports could be specified within a unit square, which is then mapped to a chosen output device. Some graphics systems provide a pair of workstation functions for this purpose. One

function is used to designate a clipping window for a selected output device, identified by a *workstation number,* and the other function is used to set the associated viewport for the device.

8.5 OpenGLTwo-Dimensional Viewing Functions

~~Actually, the basic OpenGL library has no functions specifically for two-dimensional viewing,~~ since it is assigned primarily for three-dimensional applications. But we can adapt the threedimensional viewing routines to a two-dimensional scene, and the core library contains a viewport function. In addition, the OpenGL Utility (GLU) does provide a two-dimensional function for specifying the clipping window, and we have GLUT functions for handling display windows. Therefore, we can use these two-dimensional routines, along with the OpenGL viewport function, for all the viewing operations we need.

OpenGLProjection Mode

Before we select a clipping window and a viewport OpenGL, we need to establish the appropriate mode for constructing the matrix to transform from world coordinates to screen coordinates. With OpenGL, we cannot set up a separate two-dimensional viewing-coordinate system as in **Fig.4.4,** and we must set the parameters for the clipping window as part of the projection transformation. Therefore, we must first select the projection mode. We do this with the same function we used to set the model-view mode for the geometric transformations. Subsequent commands for defining a clipping window and viewport will then be applied to the projection matrix.

glMatrixMode (GL_PROJECTION)

This designates the projection matrix as the current matrix, which is originally set to the identity matrix. However, if we are going to loop back through this statement for other views of a scene, we can also set the initialization as

glLoadIdentity ( );

This ensures that each time we enter the projection mode the matrix will be reset to the identity, so that the new viewing parameters are not combined with the previous ones.

GLU Clipping-Window Function

To define a two-dimensional clipping window, we can use the OpenGL Utility function:

gluOrtho2D (xw_{min}, xw_{max}, yw_{min}, yw_{max});

Coordinate positions for the clipping-window boundaries are given as double-precious numbers. This function specifies an orthogonal projection for mapping the scene to the screen. For a

threedimensional scene, this means that objects would be projected along parallel lines that are perpendicular to the two-dimensional *xy* display screen. But, for a two-dimensional application, objects are already defined in the *xy* plane. Therefore, the orthogonal projection has no effect on our two-dimensional scene other than to convert object positions to normalized coordinates. Nevertheless, we must specify the orthogonal projection because our two-dimensional scene is processed through the full three-dimensional OpenGL viewing pipeline. In fact, we could specify the clipping window using three-dimensional OpenGL core-library version of the gluOrtho2D function.

Normalized coordinates in the range from -1 to 1 are used in the OpenGL clipping routines. And the gluOrtho2D function sets up a three-dimensional version of transformation matrix 4-9 for mapping objects within the clipping window to normalized coordinates. Objects outside the normalized square (and outside the clipping window) are eliminated from the scene to be displayed.

If we do not specify a clipping window in an application program, the default coordinates are (xw_{min}, yw_{min})= (-1.0, -1.0) and (xw_{max}, yw_{max}) = (1.0, 1.0). Thus the default clipping window is the normalized square centered on the coordinate origin with a side length of 2.0.

OpenGLViewport Function

We specify the viewport parameters with the OpenGL function

```
glViewport ( xvmin, yvmin, vpWidth, vpHieght
);
```

Where all parameter values are given in integer screen coordinates relative to the display window. Parameters xvmin and yvmin specify the position of the lower-left corner of the viewport to the lower-left corner to the display window. And the pixel width and height of the viewport are set with parameters vpWidth and vpHeight. If we do not invoke the glViewport function in a program, the default viewport size and position are the same as the size and position of the display window.

After the clipping routines have been applied, positions within the normalized square are transformed into the viewport rectangle using matrix 4-10**.** Coordinates for the upper-right corner of the viewport are calculated for this transformation matrix in terms of the viewport width and height.

$$xv_{min} = xv_{min} + vpWidth, \qquad yv_{max} = yv_{min} + vpHieght \tag{4-11}$$

For the final transformation, pixel colors for the primitives within the viewport are loaded into the refresh buffer at the specified screen locations.

Multiple viewports can be created in openGL for a variety of applications. We can obtain the parameters for the currently active viewport using the query function.

glGetIngerv (GL_VIEWPORT, vpArray) ; wherevpArray is a single-subscript, four-element array. This Get function returns the parameters for the current viewport to vpArray in the order xvmin, yvmin, vpWidth, vpHeight. In an interactive application, for example, we can use this function to obtain parameters for the viewport that contains the screen cursor.

Creating GLUT Display Window

We use the GLUT routines for creating and manipulating display windows so that our example programs will be independent of any specific machine. To access these routines, we first need to initialize GLUT with the following function.

glutInit (&argc, argv) ;

Parameters for this initialization function are the same as those for the main procedure, and we can use glutInit to process command-line arguments.

We have three functions in GLUT for defining a display window and choosing its dimensions and position:

glutInitWindowPosition (xTopLeft, ytopLeft) ;
glutInitWindowSize (dwWidth, dwHeight) ; glutCreateWindow
(—Title of Display Window‖);

The first of these functions gives the integer, screen-coordinate position for the top-left corner of the display window, relative to the top-left corner of the screen. If either coordinate is negative, the display-window position on the screen is determined by the window-management system. With the second function, we choose a width and height for the display window is positive integer pixel dimensions. If we do not use these two functions to specify a size and position, the default size is 300 by 300 and default position is (-1, -1), which leaves the positioning of the display window to the window management system. In any case, the display-window size and position specified with GLUT routines might be ignored, depending on the state or the other requirements currently in effect for the window-management system. Thus the window system might position and size the display window differently. The third function creates the display window, with the specified size and position, and assigns a title, although the use of the little also

depends on the windowing system. At this point, the display window is defined but not shown on the screen until all the GLUT setup operations are complete.

Setting the GLUT Display-Window Mode and Color

Various display-window parameters are selected with the GLUT function

```
glutInitDisplayMode (mode) ;
```

We use this function to choose a color mode (RGB or index) and different buffer combinations, and the selected parameters are combined with the logical or operation. The default mode is single buffering and the RGB (or RGBA) color mode, which is the same as setting this mode with the statement

```
glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB) ;
```

The color mode specification GLUT_RGB is equivalent to GLUT_RGBA. A back-ground color for the display window is chosen in RGB mode with OpenGL routine

```
glClearColor (red, green, blue, alpha) ;
```

In color-index mode, we set the display-window color with

```
glClearIndex (index) ;
```

where parameter index is assigned an integer value corresponding to a position within the color table.

GLUT Display-Window Identifier

Multiple display windows can be created for an application, and each is assigned a positiveinteger **display-window identifier,** starting with the value 1 for the first window that is created.

At the time that we initiate a display window, we can record its identifier with the statement
windowID = glutCreateWindow (—A Display Window‖) ;

Once we have saved the integer display-window identifier in variable name windowID, we can use the identifier number to change display parameters or to delete the display window.

Deleting a GLUT Display Window

The GLUT library also includes a function for deleting a display window that we have created. If we know the display window‘s identifier, we can eliminate it with the statement.

```
glutDestroyWindow (WindowID) ;
```

Current GLUT Display Window

When we specify any display-window operation, it is applied to the **current display window,** which is either the last display window that we created or the one we select with the following command.

```
glutSetWindow (windowID) ;
```

And, at any time, we can query the system to determine which window is the current display window:

```
currentWindowID = glutGetWindow ( );
```

A value 0 is returned by this function if there are no display window or if the current display window was destroyed.

Relocating and Resizing a GLUT Display Window

We can reset the screen location for the current display window with

```
glutpositionWindow (xNewTopLeft, yNewTopLeft);
```

where the coordinates specify the new position for the upper-left display-window corner, relative to the upper-left to the upper-left corner of the screen. Similarly, the following function resets the size of the current display window.

```
glutReshapeWindow (dwNewWidth, dwNewHeight);
```

And with the following command, we can expand the current display window to fill the screen.

```
glutFullScreen ( );
```

The exact size of the display window after execution of this routine depends on the windowmanagement system. And a subsequent calls either glutPositionWindow or glutReshapeWindow will cancel the requests for an expansion to full-screen size.

Whenever the size of a display window is changed, its aspect ratio may change and objects may be distorted from their original shapes.

```
glutReshapeFunc (winReshapeFcn);
```

This GLUT routine is activated when the size of a display window is changed, and the new width and height are passed to its argument. The function winReshapeFucn is the ―callback function‖ for the ―reshape event‖. We can then use this callback function to change the parameters for the

viewport so that the original aspect ratio of the scene is maintained. In addition, we could also reset the clipping-window boundaries, change the display-window color, adjust other viewing parameters, and perform any other tasks.

Managing Multiple GLUT Display Windows

The GLUT library also has a number of routines for manipulating a display window in various ways. These routines are particularly useful when we have multiple display windows on the screen and we want to rearrange them or locate a particular display window.

We use the following routine to convert the current display window to an icon in the form of a small picture or symbol representing the window.

```
glutIconifyWindow ( );
```

This icon will be labeled with the same name that we assigned to the window, but we can change the name for the icon with

```
glutSetIconTitle (—IconName‖);
```

And we can change the name of the display window with a similar command:

```
glutSetWindowTitle ( —New Window Name‖);
```

With multiple display windows open on the screen, some windows may overlap or totally obscure other display windows. We can choose any display window to be in front of all other windows by first designating it as the current window, then issuing the —pop-window‖ command: glutSetWindow (windowID); glutPopWindow ( );

In a similar way, we can —push‖ the current display window to the back, so that it is behind all other display windows. This sequence of operations is glutSetwindow (windowID); glutpusuWindow ( );

We can also take the current window off the screen with

```
glutHideWindow ( );
```

And we can return a —hidden‖ display window or one that has been converted to an icon, by designating it as the current display window and then invoking the function

```
glutShowWindow ( );
```

GLUT Subwindows

Within a selected display window, we can set up any number of second-level display windows, called *subwindows.* This provides a means for partitioning display windows into different display

sections. We create a subwindow with the following function glutCreateSubWindow (windowID, xBottomLeft, yBottomLeft, width, height);

Parameter windowID identifies the display window in which we want to set up the subwindow. With the remaining parameters, we specify its size and the placement of the lower-left corner of the subwindow relative to the lower-left corner of the display window.

Subwindow is assigned a positive integer identifier in the same way that first-level display windows are numbered. And we can place a subwindow inside another subwindow. Also, each subwindow can be assigned an individual display mode and other parameters. We can even reshape, reposition, push, pop, hide, and show subwindows, just we can with first-level display windows. But we cannot convert a GLUT subwindow to an icon.

Selecting a Display-Window Screen-Cursor Shape

We can use the following GLUT routine to request a shape for the screen cursor that is to be used with the current window.

glutSetCursor (shape);

the possible cursor that we can select are an arrow pointing in a chosen direction, a bidirectional arrow, a rotating arrow, a crosshair, a wristwatch, a question mark, or even skull and crossbones. For example, we can assign the symbolic constant GLUT_CURSOR_UP_DOWN to parameter shape to obtain an up-down, bidirectional arrow. A rotating arrow is chosen with GLUT_CURSOR_CYCLE, a wristwatch shape is selected with GLUT_CURSOR_WAIT, and skull and crossbones are obtained with the constant GLUT_CURSOR_DESTROY. A cursor shape can be assigned to a display window to indicate a particular kind of application, such an animation. However, the exact shapes that we can use are system dependent.

Viewing Graphics Objects in a GLUT Display Window

After we have created a display window and selected its position, size, color, and other characteristics, we indicate what is to be shown in that window. If more than one display window has been created, we first designate the one we want as the current display window. Then we invoke the following function to assign something to that window.

glutDisplayFunc (pictureDescrip);

The argument is a routine that describes what is to be displayed in the current window. This routine, called pictureDescrip for this example is referred to as a *callback function* because it is the routine that is to be executed whenever GLUT determines that the display-window contents

should be renewed. Routine pictureDescrip usually contains the OpenGL primitives and attributes that define a picture, although it could specify other constructs such as menu display.

If we have set up multiple display windows, then we repeat this process for each of the display windows or subwindows. Also, we may need to call glutDisplayFunc after the glutPopWindow command if the display window has been damaged during the process of redisplaying the windows. In this case, the following function is used to indicate that the contents of the current display window should be renewed.

glutPostRedisplay ( );

This routine is also used when as additional object such as a pop-up menu is to be shown in a display window.

Executing the Application program

When the program setup is completed and the display windows have been created and initialized, we need to issue the final GLUT command that signals execution of the program;

glutMainLoop ( );

At this time, display windows and their graphic contents are sent to the screen. The program also enters the **GLUT processing loop** that continually checks for new ―events‖, such as interactive input from a mouse or a graphics tablet.

---

6. ClippingAlgorithms

~~Generally, any procedures that eliminates those portions of a picture that are either inside or~~ outside of a specified region of space is referred to as a **clipping algorithm** or simply **clipping.** Usually a clipping region is a rectangle in standard position, although we could use any shape for a clipping application.

The most common application of clipping is in the viewing pipeline, where clipping is applied to extract a designated portion of a scene (either two-dimensional or three-dimensional) for display on an output device. Clipping methods are also used to antialiasing object boundaries, to construct objects using solid-modeling methods, to manage a multi-windowenvironment, and to allow parts of a picture to be moved, copied, or erased in drawing and painting programs.

Clipping algorithms are applied in two-dimensional viewing procedures to identify those parts of a picture that are within the clipping window. Everything outside the clipping window is then eliminated from the scene description that is transferred to the output device for display. An

efficient implementation of clipping in the viewing pipeline is to apply the algorithms to the normalized boundaries of the clipping window. this reduces calculations, because all geometric and viewing transformations matrices can be concatenated and applied to a scene description before clipping is carried out. The clipped scene can then be transferred to screen coordinates for final processing.

In the following sections, we explore two-dimensional algorithmsfor

- Point Clipping
- Line Clipping
- Fill-area Clipping
- Curve Clipping
- Text Clipping

Point, line, and polygon clipping are standard components of graphics packages. But similar methods can be applied to other objects, particular conics, such as circles, and spheres, in addition to spline curves and surfaces. Usually, however, objects with nonlinear boundaries are approximated with straight-line segments or polygon surfaces to reduce computations.

Unless otherwise stated, we assume that the clipping region is a rectangular window in standard position, with boundary edges at coordinate positions xw_{min}, yw_{max}, yw_{min} *and* yw_{max}. These boundary edges typically correspond to a normalized square, in which the *x* and *y* values range either from 0 to1 or from -1 to 1.

7. Two-dimensional point Clipping

For a clipping rectangle in standard position, we save a two-dimensional point **P** = (x, y) for display if the following inequalities are satisfied:

$$xw_{min} \leq x \leq xw_{max}$$

$$syw_{min} \leq y \leq yw_{max} \qquad (4\text{-}12)$$

If any one of the four inequalities is not satisfied, the point is clipped (not saved for display).

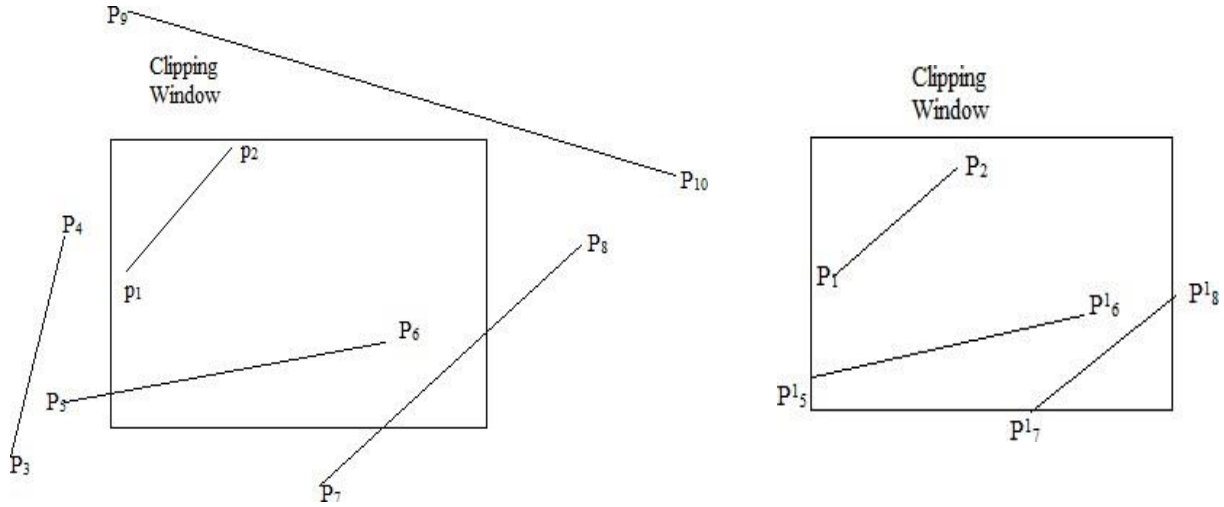
Although point clipping is applied less often than line or polygon clipping, it is useful in various situations, particularly when pictures are modeled with particle systems. For example, point

clipping can be applied to scenes involving clouds, sea foam, or explosions that are modeled with ―particles‖,such as the center coordinates for small circles or spheres.

8. Two-Dimensional Line Clipping

Figure 4-11 illustrates possible positions for straight-line segments in relationship to a standard clipping window. A line-clipping algorithm processes each line in a scene through a series of tests and intersection calculations to determine whether the entire line or any part of it is to be saved. The expensive part of a line-clipping procedure is in calculating the intersection positions of a line with the window edges. Therefore, a major goal for any line-clipping algorithm is to minimize the intersection calculations. To do this, we can first perform tests to determine whether a line segment is completely inside the clipping window, but it is more difficult to identify all lines that are entirely outside the window. If we are unable to identify a line as completely inside or completely outside a clipping rectangle, we must then perform intersection calculations to determine whether any part of the line crosses the window interior.

We test a line segment to determine if it is completely inside or outside a selected clippingwindow edge by applying the point-clipping tests of the previous section. When both endpoints of a line segment are inside all four clipping boundaries, such as the line from $\mathbf{P_1}$to $\mathbf{P_2}$in Fig. 4.11, the line is completely inside the clipping window and we save it. And when both endpoints of a line segment are outside any one of the four boundaries, that line is completely outside the window and it is eliminated from the scene description. But if both these tests fail, the line segment intersects at least one clipping boundary and it may or may not cross into the interior of the clipping window.



(a) BeforeClipping

(b) After clipping

Figure 4-11 Clipping straight-line segments using a standard rectangular clipping window.

One way to formulate the equation for a straight-line segment is to use the following parametric representation, where the coordinate positions (x_0, y_0) and (x_{end}, y_{end})designated the two line endpoints.

$$x = x_0 + u(x_{end}, -x_0)$$

$$y = y_0 + u(y_{end} - y_0) \qquad 0 \le u \le 1 \tag{4.13}$$

We can use this parametric representations to determine where a line segmentcrosses each clipping-window edge by assigning the coordinate value for that edge to either x or y and solving for parameter u. As an example, the left window boundary is at position xw_{min}, so we substitute this value for x, solve for u, and calculate the corresponding y-intersection value. If this value of u is outside the range from 0 to 1, the line segment does not intersect that window border line.
But if the value of u is within the range from 0 to 1, part of the line inside that border. We can then process this inside portion of the line segment againstthe other clipping boundaries until either we have clipped the entire line or we find a section that is inside the window.

Processing line segments in a scene using the simple clipping approach described in the preceding paragraph is straightforward, but not very efficient. It is possible reformulate the initial testing and the intersection calculations to reduce processing time for a set of line segments, and a number of faster line clippers have been developed. Some of the algorithms are designed

explicitly for two-dimensional pictures and some are easily adapted to sets of three-dimensional line segments.

Cohen-Sutherland line Clipping

This is not the earliest algorithms to be developed for fast line clipping, and variations of this method are widely used. Processing time is reduced in the Cohen-Sutherland method by performing more tests before proceeding to the intersection calculations. Initially, every line endpoint in a picture is assigned a four-bit binary value, called a **region code,** and each bit position is used to indicate whether the point is inside or outside one of the clipping-window boundaries. We can reference the window edges in any order, and fig 4.12 illustrates one possible ordering, with the bit positions numbered 1 through 4 from right to left. Thus, for this ordering, the rightmost position (bit 1) references the left clipping window boundary, and the leftmost position (bit-4) references the top window boundary. A value of 1 (or true) in any bit position indicates that the endpoint is outside of that window border. Similarly, a value of 0 (*or false*) in any bit position indicates that the endpoint is not outside (it is inside or on) the corresponding window edge. Sometimes, a region code is referred to as an **—out"** codebecause a value of 1 in any bit position indicates that the spatial point is outside the corresponding clipping boundary.

Each clipping-window edge divides two-dimensional space into an inside half space and an outside half space. Together, the four window borders create nine regions, and Fig 4.13 lists the value for the binary code in each of these regions. Thus, an endpoint that is below and to the left of the clipping window is assigned the region code 0101, and the region-code value for any endpoint inside the clipping window is 0000.

Bit values in a region are determined by comparing the coordinate values (x,y) of an endpoint to the clipping boundaries. Bit 1 is set to 1 if $x < xw_{min}$, and the other three bit values are determined similarly. Instead of using inequality testing, we can more efficiently determine the values for a region-code using bit-processing operations and the following two steps: (1) Calculate differences between endpoint coordinates and clipping boundaries. (2) Use the resultant sign bit of each difference calculation to set the corresponding value in the region code. For the ordering scheme shown in Fig. 4.12, bit 1 is the sign bit of $x - xw_{min}$; bit 2 is the sign bit of xw_{max} ; bit 3 is the sign bit of $y - yw_{min}$; and bit 4 is the sign bit of $yw_{max} - y$.

Once we have established region codes for all line endpoints, we can quickly determine which lines are completely inside the clip window and which are clearly outside. Any lines that are completely contained within the window edges have a region-code of 0000 for both endpoints, and we save these line segments. Any line that has a region-code value of 1 in the same bit position for each endpoint is completely outside the clipping rectangle, and we eliminate that line segment. As an example, a line that has a region code of 1001 for one endpoint and a code of 0101 for the other endpoint is completely to the left of the clipping window, as indicated by the value of 1 in the first bit position of each region code.

We can perform the inside-outside tests for line segments using logical operators. When the ***or*** operation between two endpoint region codes for a line segment is *false* (0000), the line is inside the clipping window. Therefore, we save the line and proceed to test the next line in the scene description. When the ***and***operation between two endpoint region codes for a line is *true (not 0000),* the line is completely outside the clipping window, and we can eliminate it from the scene description.

Lines that cannot be identified as being completely inside or completely outside a clipping window by the region-code tests are next checked for intersection with the window border lines without entering the interior of the window. Therefore, several intersection calculations might be necessary to clip a line segment, depending on the order in which we process the clipping boundaries. As we process each clipping-window edge, a section of the line is clipped, and the remaining part of the line is checked against the other window borders. We continue eliminating sections until either the line is totally clipped or the remaining part of the line is inside the clipping window. For the following discussion, we assume that the window edges are processed in the order: left, right, bottom, top. To determine whether a line crosses a selected clipping boundary, we can check corresponding bit values in the two endpoint region codes. If one of these bit values is 1 and the other is 0, the line segment crosses that boundary.

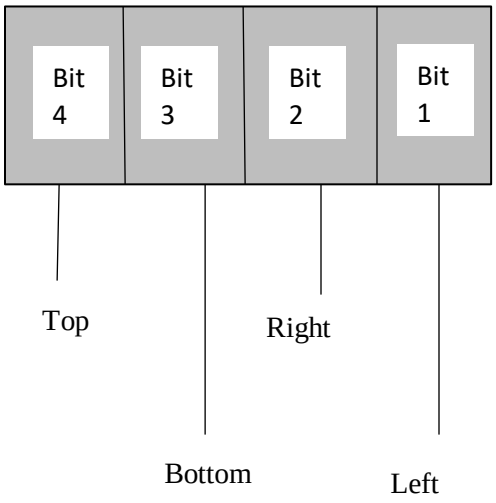


Figure 4-12 A possible ordering for the clipping window boundaries corresponding to the bit positions in the Cohen-Sutherland endpoint region code.

Fig. 4-14 illustrates two line segments that cannot be immediately identified as completely inside or completely outside the clipping window. The region codes for the line $\mathbf{P_1}$to $\mathbf{P_2}$are 0100 and 1001. Thus, $\mathbf{P_1}$ is inside the left clipping boundary and $\mathbf{P_2}$is outside the boundary. We then calculate the intersection position $\mathbf{P'_2}$, and we clip off the line section from $\mathbf{P_2}$to $\mathbf{P'_2}$. The remaining portion of the line inside the right border line, and so we next check the bottom order. Endpoint $\mathbf{P_1}$ is below the bottom clipping edge and $\mathbf{P'_2}$is above it, so we determine the intersection position at this boundary ($\mathbf{P'_1}$**).** We eliminate the line section from $\mathbf{P_1}$to $\mathbf{P'_1}$ and proceed to the top window edge. There we determine the intersection position to be $\mathbf{P''_2}$**.** The final step is to clip off the section above the top boundary and save the interior segment from $\mathbf{P'_1}$to $\mathbf{P''_2}$ For the second line, we find that point **P** is outside the left boundary and **P** inside.

2. 3 4

Thus, we calculate the intersection position $\mathbf{P'_3}$ and eliminate the line section from $\mathbf{P_3}$ to$\mathbf{P'_3}$**.** By checking region codes for the endpoints $\mathbf{P'_3}$ and $\mathbf{P_4}$, we find that the remainder of the line is below the clipping window and can be eliminated also.

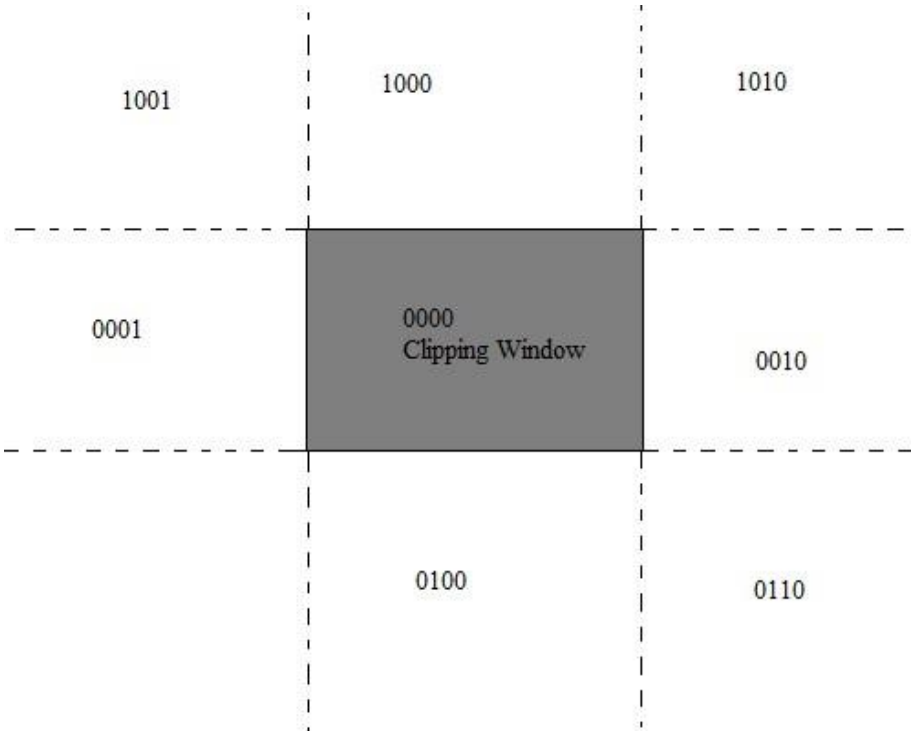


Figure 4.13 The nine binary region codes for identifying the position of a line endpoint, relative to the clipping-window boundaries.

It is possible, when clipping a line segment using this approach, to calculate an intersection position at all four clipping boundaries, depending on how the line endpoints are processed and what ordering we use for the boundaries. Figure 4-15 shows the four intersection positions that could be calculated for a line segment that is processed against the clipping-window edges in the order left, right, top, bottom. Therefore, variations of this basic approach have been developed in an effect to reduce the intersection calculations.

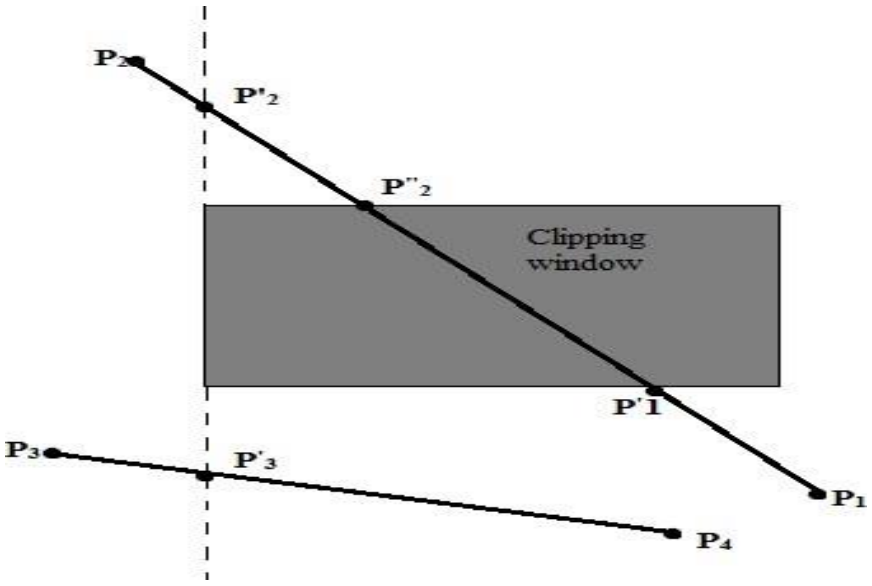


Figure 4-14 Lines extending from one clipping-window region to another may cross into the clipping window, or they could intersect one or more clipping boundaries without entering the window interior.

To determine a boundary intersection for a line segment, we can use the slope-intercept form of the line equation. For a line with endpoint coordinates (x_0, y_0) and (x_{end}, y_{end}), the y coordinate of the intersection point with a vertical clipping border line can be obtained with the calculation.

$$y = y_0 + (x - x_0) \tag{4-14}$$

where the x value is set to either xw_{min} or xw_{max}, and the slope of the line is calculated as $m = (y_{end} - y_0)/(x_{end} - x_0)$. Similarly, if we are looking for the intersection with a horizontal border, the x coordinate can be calculated as

$$x = x_0 + \frac{y - y_0}{\quad}$$

With y set to either yw_{min} or to yw_{max}.

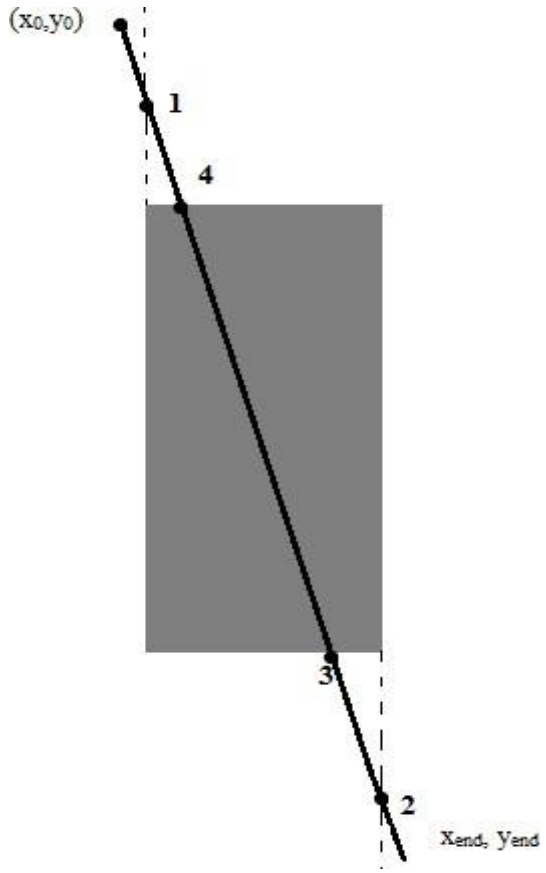


Figure 4-15 Four intersection positions (labeled from 1 to 4) for a line segment that is clipped against the window boundaries in the order left, right, top, bottom.

8.9 Summary

The two-dimensional viewing-transformation pipeline is a series of operations that result in the display of a world coordinate picture that has been defined in the *xy*plane. After we construct the scene, it can be mapped to a viewing-coordinate reference frame, then to a normalized coordinate system where clipping routines can be applied. Finally, the scene is transferred to device coordinates for display. Normalized coordinates can be specified in the range from 0 to 1, they are used to make graphics packages independent of the output-device requirements.

We select part of a scene for display on an output device using a clipping window, which can be described in the world coordinate system or in a viewing coordinate frame defined relative to world coordinates. The contents of the clipping window are transferred to a viewport for display on an output device. In some systems, a viewport is specified within normalized coordinates. Other systems specify the viewport in device coordinates. Typically, the clipping window and viewport are rectangles whose edges are parallel to the coordinate axes. An object is mapped to

the viewport so that it has the same relative position in the viewport as it has in the clipping window. To maintain the relative proportions of an object, the viewport must have the same aspect ratio as the corresponding clipping window. And we can set up any number of clipping windows and viewports of a scene.

Clipping algorithms are usually implemented in normalized coordinates, so that all geometric transformations and viewing operations that are independent of device coordinates can be concatenated into one transformation matrix.

10. Keywords

Clipping window, viewing functions, Projection mode, Clipping-window function, Viewport function, Clipping Algorirhm, line clipping, Point clipping, Cohen-Sutherland line clipping.

11. Exercises

1) Explain brifely the Two-dimensional viewing pipeline.
2) What is clipping window? Explain viewport-coordinate and world-coordinate clipping window.
3) Discusss in detail about Normalization and Viewport Transformation.
4) Explain different types of Two-dimensional viewing funsctions.
5) What is clipping algorithm? Explain Two-dimensional point clipping algorithm.
6) Explain Cohen-Sutherland line clipping algorithm.

12. References

1) Ralf Steinmentz, klara Naestedt: Multimedia Fundamentals: vol 1-media Coding and Content processing,2nd edition, PHI, Indian Reprint 2008.
2) Prabhat K. Andleigh, Kiran Thakrar, Multimedia System Design, PHI,2003
3) Ze-Nian Li-mark S Drew, Fundamentals of multimedia, PHI, New Drelhi, 2011.
4) Donald hearn and M. Pauline Baker, Computer graphics, 3rd edition, Pearson.

MODULE 3

UNIT 9: Introduction to Animation

Structure:

1. Learning Objectives
2. Introduction
3. Definition
4. The History ofAnimation
5. Line art and Animation
6. Unit Summary
7. Keywords
8. Exercise
9. References

1. Learning Objectives:

After studying this unit, you will be able to

- Know about computer animation
- Explain about the concepts of animation
- Elucidate about history of animation

1. Introduction:

Computer animation is the use of computers to create animations. Animation is the illusion of continuous movement of objects or camera viewpoint created by a series of still images. Each still image is called a frame. Most people perceive motion as smooth (continuous) when presented by a sequence of images at a rate of at least 10 frames per second.

Animation is a beautiful way to create films. Also, it is eye-catching and expressive. Animation is nothing but moving pictures. Now, the moving picture can be anything. It can be a leaf falling on the ground, or it can a full feature movie like Shrek. There are various objects (medium) that can be used to make an animation. Hence, any medium can make a type of animation. For example, some people make beautiful animated films using just sand. While some movies are made using complex software. An animated movie can be made in minutes or can take years, it all depends upon the project. There are many types of animation, as the medium can be anything. It can be pencil drawings, can be computer drawings, can be computer models or can be ordinary objects like clay, sand, coins, buttons, etc. However, all different types of

animation fall basically under three main categories, and so these three categories can be called the basic animation types. They are stop-motion, 2D animation and 3D animation.

Stop-Motion

So, what is stop-motion? First let us have a mental picture about this animation, and then let us discuss it. Have you seen Pingu the Penguin animation? No? Then spend a minute watching a video of Pingu online. Okay, now that you have seen it, let us discuss stop-motion animation. Stop-motion is basically a type of animation in which inanimate objects are used. Inanimate objects are moved and captured (a photo is taken). Then they are moved a bit and a shot is taken, this goes on till a motion is complete. Then all the photos are played in a sequence and we see an animation, an illusion of life.

Now, the inanimate object can be anything. So, there is claymation, which is a popular type of stop-motion. There are animated stop-motion films made with many many inanimate things. There is sand-animation, button-animation, paper cut-out animation, pencil-animation, etc. Now, the difference between capturing a video and stop-motion is that the filmmaker can move the objects according to their will and convenience, as opposed to live-action videos.
Some examples of stop-motion movies are The Adventures of Prince Achmed, The Corpse Bride and Rudolph the Red-Nosed Reindeer.

2DAnimation

2D animation can be basically divided into two main categories. One is cell animation, which is classical animation. This is the way all old 2D movies were made, but due to the advent of computer technology the other category of 2D animation which is computer generated has become very popular. Cell animation is done by drawing frame-by-frame. This means on paper, drawings are created and then these drawings are captured by a camera. An animation clip is created using photos of these drawings.

The computer generated 2D animation is basically created by drawing and painting sketches on the computer using a software, without any actual paper sketches involved. One of the most popular animation software used to create computer 2D animation is Adobe Flash.

However, there is a third category of 2D animation which synthesizes both the above types of 2D animation. In this animation type half animation is computer drawn, while half is done using a software. For example, pencil drawings scanned in computer and then colored using a software.

Some examples of classical cell animation are Disney's Steamboat Willie, Aladdin and Pinocchio. Some of the computer generated animation series are The Powerpuff Girls, Dexter's Laboratory and The Jetsons.

3DAnimation

Today 3D animation is dominating when it comes to full-length feature films. 3D animation is basically created using a software which allows to create models and animate them in a 3D environment, on three axis X, Y, and Z, while for 2D animation there are only two axis. In 2D animation you cannot look at a character in various angles by rotating it, but in 3D animation this is possible.

3D animation is completely done using software. Maya and 3D Max are two of the most popular 3D animation software. There is also Blender which is easier to understand. Blender is simple software and easy to grasp, while to comprehend Maya and Max you will need a lot of practice. There is also another type of animation which is called motion capture. However, motion capture is actually not a type of animation but a mix medium of live action and 3D animation. In this medium basically a motion is captured and then rendered using a software, to give an effect of an animation clip.

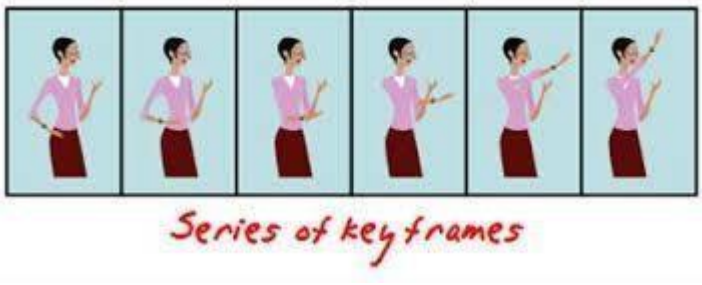
Some examples of popular 3D animation are Shrek, Ice Age and Finding Nemo.

Fig: 3Danimation

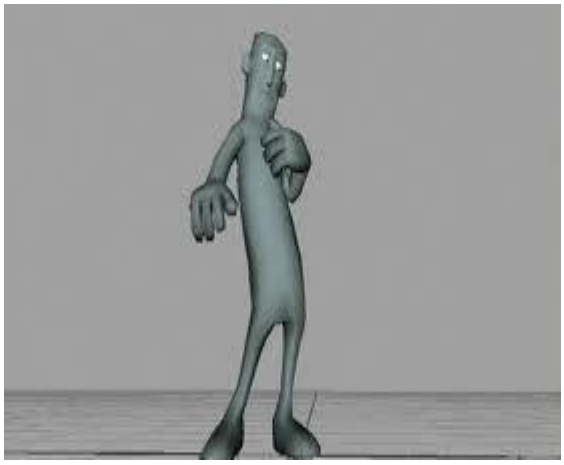
Other Types of Animation:

- **Real time animation:** An animation is real time if a computer is computing and displaying the animation at the same speed as it is designed to be seen at. Typically only simple animations can be displayed in real time.

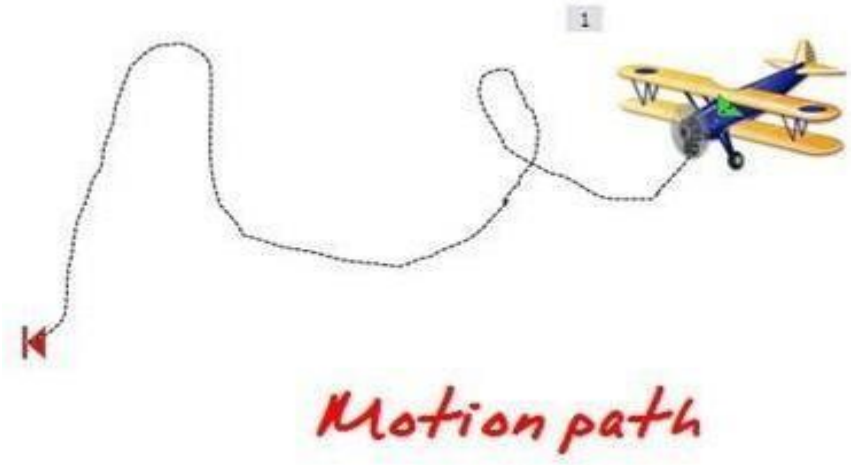
- **Keyframe animation:** A technique for producing animations whereby important positions, sizes and orientations of objects at particular points in time are identified and everything else in-between is filled in by interpolation.



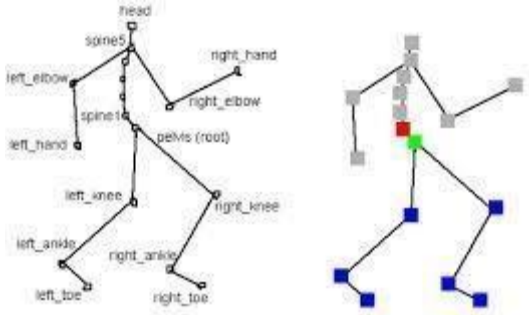
- **Character animation:** Animation focussed on the display of expressions, emotions and behaviours normally associated with intelligent life forms.



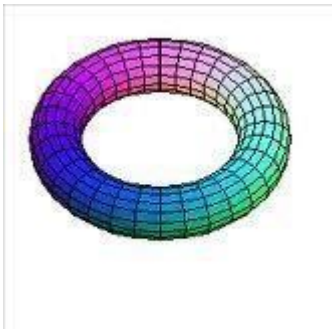
- **Motion path animation:** A technique where objects or cameras move along a path.



- **Hierarchical animation:** Animation of hierarchical objects.

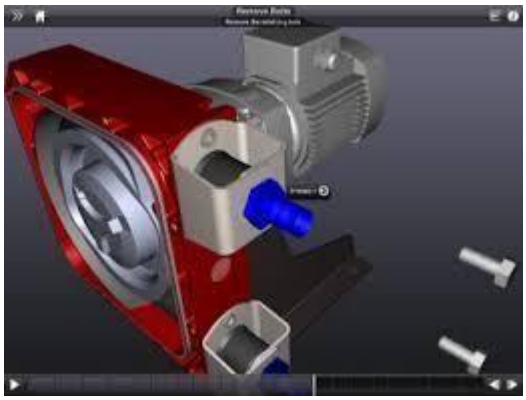


Shape animation: There are various forms but the most well known is morphing where one shape changes into another shape.



- **Procedural animation:** Animations typically require considerable data in their production. Procedural animation aims to compute animation data rather than have an animator specify it.

- **Simulation:** May be characterised as scientific animation. Typically simulations will be using data computed from the laws of physics.



- **Camera animation:** Typified by architectural walkthroughs. In its pure form the only thing which moves is the camera.

9.2 Definition:

Animation is the process of creating a continuous motion and shape change illusion by means of the rapid display of a sequence of static images that minimally differ from each other. The illusion as in motion pictures in general is thought to rely on the phi phenomenon. Animators are artists who specialize in the creation of animation.

Animations can be recorded on either analogue media, such as a flip book, motion picture film, video tape, or on digital media, including formats such as animated GIF, Flash animation or digital video. To display it, a digital camera, computer, or projector are used.

Animation creation methods include the traditional animation creation method and those involving stop motion animation of two and three-dimensional objects, such as paper cutouts, puppets and clay figures. Images are displayed in a rapid succession, usually 24, 25, 30, or 60 frames per second.

Animation has certainly come a long way in the decades since its debut in the early 1900s. The techniques used by animators to bring characters and stories to life have improved immeasurably over the years, yet there remains only three primary types of animation: traditional, stop-motion, and computer.

The differences between the three major forms of animation are significant, and covered in the following article:

Traditional Animation:

Arriving on the scene at roughly the same time as its live-action counterparts, traditionally animated films have certainly come a long way since the early days of crude drawings and experimental narratives. Traditional animation made its debut in 1906 with a short film featuring different facial expressions. The genre allows for the illusion of animated movement due to the frame-by-frame manipulation of drawings and illustrations. Although computer technology has assisted animators in their efforts over the years, the basic means by which an animated film comes to life has essentially remained the same.

The popularization of the cel-animation process in the early _20s proved instrumental in the genre‘s meteoric rise to infamy, with the technique ensuring that animators no longer had to draw the same image over and over again – as see-through ―cels‖containing a character or object in motion could be laid on top of a stationary background. The release of Snow White and the Seven Dwarfs in 1937 marked the first time that traditionally animated films began to be taken seriously by the Hollywood community and audiences alike.

In the years since, traditionally animated films have remained popular at cinemas the world over – with the wild success of the genre affording filmmakers the opportunity to break out of the mold from time to time (ie 1972‘s Fritz the Cat became the first animated feature to land an ―X‖rating). Disney‘s domination over the 2D animated realm has ensured that their name has become synonymous with animated films, although it‘s certainly worth noting that some of the most popular cartoons from the last couple of decades have come from other studios (including The Rugrats Movie, Beavis and Butt-head Do America, and the Land Before Time series).

Stop-MotionAnimation:

Far less prevalent, on the other hand, is stop-motion animation. Stop-motion actually predates traditional, hand-drawn animation: The first attempt, The Humpty Dumpty Circus, was released in 1898. But stop-motion animated features have never quite managed to garner the acclaim and widespread appeal of their 2D counterparts. There‘s little doubt that the biggest hindrance to stop-motion animation‘s success is its time consuming nature, as animators must move an object one frame at a time to mimic movement. Considering movies generally contain 24 frames per second, it can take hours to capture just a few seconds worth of footage.

Although the first full-length stop-motion cartoon was released in 1926 (Germany‘s The Adventures of Prince Achmed), the genre‘s widest exposure came in the 1950s with the release

of the *Gumby* television series. After that point, stop-motion animation started to be seen less as a gimmicky fad and more as a viable alternative to hand-drawn animation – with 1965‘s Willy McBean and his Magic Machine, produced by legendary stop-motion duo Arthur Rankin and Jules Bass, the first full-length stop-motion film to be produced within the United States.

The prominence of Rankin/Bass Christmas specials in the _60s and _70s only added to stop-motion animation‘s growing popularity, yet it was the increased use of stop-motion within special effects field that cemented its place as an invaluable resource – with George Lucas‘ pioneering work in both the *Star Wars* films and in his effects company Industrial Light and Magic setting a standard that the rest of the industry struggled to match.

Stop-motion has, as of late, lost a lot of its luster in the wake of computer animation‘s meteoric rise, yet the genre has seen something of a resurgence in the past few years – with the popularity of movies like Coraline and Fantastic Mr. Fox ensuring that stop-motion will likely continue to endure in the years to come.

ComputerAnimation:

Before it became a pervasive, all-encompassing force within the cinematic community, computer animation was primarily used as a tool by filmmakers to enhance their traditionallyconceived special effects work. As such, computer-generated imagery was used sparingly in the

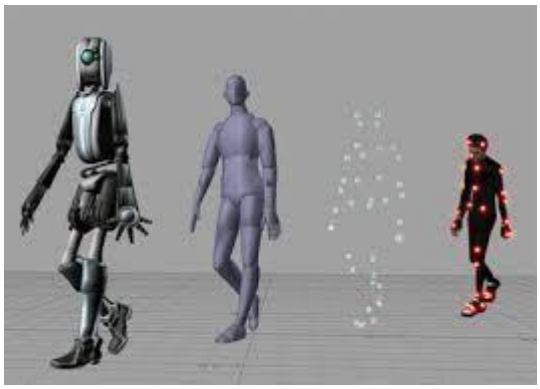
_70s and _80s – with 1982‘s Tron marking the first time it was used on an extensive basis within a full-length feature.

Computer animation received a substantial boost in 1986 with the release of Pixar‘s first short, Luxo Jr. – which went on to receive a nomination for Best Animated Short Film and proved that computers could provide more than just behind-the-scenes special effects support.

The increased sophistication of both hardware and software was reflected in the progressively eye-popping nature of computer-generated imagery, with 1991‘s Terminator 2: Judgment Day and 1993‘s Jurassic Park standing as landmark examples of what computers were capable of.

It wasn‘t until Pixar released the world‘s first computer-animated feature in 1995, Toy Story, that audiences and executives alike first began to see the possibilities offered by the technology, and it wasn‘t long before other studios began clamoring to get into the CGI game. The three dimensional appearance of computer-generated cartoons instantly assured their success over their 2-D counterparts, as viewers found themselves transfixed by the novelty of the lifelike images and jaw-dropping visuals.

And although Pixar remains the undisputed champion of the computer-generated landscape, there have certainly been plenty of equally successful examples of the genre in recent years – with, for instance, the Shrek series raking in well over two billion dollars worldwide. It's also impossible to downplay the significance of Avatar's incredible, chart-topping success. The movie - which boasts some of the most impressive computer-animated visuals to date - will likely set a new standard that future CGI-heavy films will struggle to match.



Applications of animation:

- Films
- TV programs
- Computer games
- Web contents

9.3 The History of Animation:

Computer animation began in the 1960s, with abstract color designs in motion. This technology has advanced so much that artists draw and modify pictures using computer imagery.

Using computer programs to change backgrounds and choose colors, the finished product is visualized on a television monitor, and saved on computer disc. Multimedia computer systems integrate materials from a variety of formats. Film animation applies techniques of cinematography to the graphic and plastic arts in order to give the illusion of life and movement to cartoons, drawings, paintings, puppets, and three-dimensional objects. Animated graphics have served to delineate the web as an artist's medium. However, old-style cell animation is the mainstay of professional artists in creating characters, and computers are only used to move the objects and backgrounds within a scene. Below are articles with information on computer animation, animated graphics and special effects. The advent of computer aided design (CAD) programs has propelled many educational and commercial endeavors to the next level of study, production, and efficiency. Medical students, for example, can perform complex "virtual surgeries" with the use of such systems. In addition, flight simulators, which also use such programs, aid in the training of astronauts and pilots. The video game industry has grown enormously as games become increasingly realistic and interactive through 3-D imaging. A modern form of 3D vision is virtual reality.

Early examples of attempts to capture the phenomenon of motion into a still drawing can be found in paleolithic cave paintings, where animals are often depicted with multiple legs in superimposed positions, clearly attempting to convey the perception of motion.

An earthen goblet discovered at the site of the 5,200-year-old Burnt City in southeastern Iran, depicts what could possibly be the world‘s oldest example of animation. The artifact bears five sequential images depicting a Persian Desert Ibex jumping up to eat the leaves of a tree.

Ancient Chinese records contain several mentions of devices that were said to "give an impression of movement" to human or animal figures, but these accounts are unclear and may only refer to the actual movement of the figures through space.

In the 19th century, the phenakistoscope (1832), zoetrope (1834) and praxinoscope (1877), as well as the common flip book, were early animation devices that produced an illusion of movement from a series of sequential drawings, but animation did not develop further until the advent of motion picture film and cinematography in the 1890s.

The cinématographe was a projector, printer, and camera in one machine that allowed moving pictures to be shown successfully on a screen which was invented by history's earliest film makers, Auguste and Louis Lumière, in 1894. The first animated projection (screening) was created in France, by Charles-Émile Reynaud, who was a French science teacher. Reynaud

created the Praxinoscope in 1877 and the Théâtre Optique in December 1888. On 28 October 1892, he projected the first animation in public, Pauvre Pierrot, at the Musée Grévin in Paris. This film is also notable as the first known instance of film perforations being used. His films were not photographed, but drawn directly onto the transparent strip. In 1900, more than 500,000 people had attended these screenings.

The first film that was recorded on standard picture film and included animated sequences was the 1900 Enchanted Drawing, which was followed by the first entirely animated film - the 1906 Humorous Phases of Funny Faces by J. Stuart Blackton, who, because of that, is considered the father of American animation.

In Europe, the French artist, Émile Cohl, created the first animated film using what came to be known as traditional animation creation methods - the 1908 Fantasmagorie. The film largely consisted of a stick figure moving about and encountering all manner of morphing objects, such as a wine bottle that transforms into a flower. There were also sections of live action in which the animator‘s hands would enter the scene. The film was created by drawing each frame on paper and then shooting each frame onto negative film, which gave the picture a blackboard look.

The author of the first puppet-animated film (The Beautiful Lukanida (1912)) was the Russian-born (ethnically Polish) director Wladyslaw Starewicz, known as Ladislas Starevich.

The more detailed hand-drawn animations, requiring a team of animators drawingeach frame manually with detailed backgrounds and characters, were those directed by Winsor McCay, a successful newspaper cartoonist, including the 1911 Little Nemo, the 1914 Gertie the Dinosaur, and the 1918 The Sinking of the Lusitania.

During the 1910s, the production of animated short films, typically referred to as "cartoons", became an industry of its own and cartoon shorts were produced for showing in movie theaters. The most successful producer at the time was John Randolph Bray, who, along with animator Earl Hurd, patented the cel animation process which dominated the animation industry for the rest of the decade.

El Apóstol (Spanish: "The Apostle") was a 1917 Argentine animated film utilizing cutout animation, and the world's first animated feature film. Unfortunately, a fire that destroyed producer Frederico Valle's film studio incinerated the only known copy of El Apóstol, and it is now considered a lost film.

Computer animation has become popular since Toy Story (1995), the first feature-length animated film completely made using this technique.

In 2008, the animation market was worth US$68.4 billion. Animation as an art and industry continues to thrive as of the mid-2010s, because well-made animated projects can find audiences across borders and in all four quadrants. Animated feature-length films returned the highest gross margins (around 52%) of all film genres in the 2004-2013 timeframe.

9.4 LineArt andAnimation:

Line art or line drawing is any image that consists of distinct straight and curved lines placed against a (usually plain) background, without gradations in shade (darkness) or hue (color) to represent two-dimensional or three-dimensional objects. Line art can use lines of different colors, although line art is usually monochromatic. Line art emphasizes form and outline, over color, shading, and texture. However, areas of solid pigment and dots can also be used in addition to lines. The lines in a piece of line art may be all of a constant width (as in some pencil drawings), of several (few) constant widths (as in technical illustrations), or of freely varying widths(as in brush work or engraving).

Line art may tend towards realism (as in much of Gustave Doré's work), or it may be a caricature, cartoon, ideograph, or glyph.

Before the development of photography and of halftones, line art was the standard format for illustrations to be used in print publications, using black ink on white paper. Using either stippling or hatching, shades of gray could also be simulated.

One of the most fundamental elements of art is the line. An important feature of a line is that it indicates the edge of a two-dimensional (flat) shape or a three-dimensional form. A shape can be indicated by means of an outline and a three-dimensional form can be indicated by contour lines.

5. Unit Summary:

This unit introduced concepts of animation and history behind animation techniques.

6. Keywords:

Animation, Line Art and Animation

7. Exercise:

1) What is animation? Discuss the applications of animation.

2) List and explain the basic types of animation.

3) Explain the history behind animation.

4) Write short note on lineart.

9.8 References:

1.Prabat K Andleigh and Kiran Thakrar, ―Multimedia Systems and Design‖, PHI, 2003. (UNIT 3 to 5)

2.Donald Hearn and M.Pauline Baker, ―Computer Graphics C Version‖, Pearson Education, 2003.

3. Designing Interfaces by Jenifer Tidwell

4. Digital Multimedia by Nigel Chapman

5.Information Architecture for the World Wide Web: Designing Large-Scale Web Sites by Peter Morville, Louis Rosenfeld

6. Information Visualization, Second Edition: Perception for Design by Colin Ware

7. Letting Go of the Words: Writing Web Content that Works by Janice Redish

8. Multimedia Applications (X.media.publishing) by Ralf Steinmetz

9. Now You See It: Simple Visualization Techniques for Quantitative Analysis by Stephen Few

10. Visual Thinking: for Design by Colin Ware

11.Writing for New Media, Third Edition: Content Development for Bloggers and Professionals by Timothy Paul Garrand

12. Real-Time Video Compression: Techniques and Algorithms By Raymond Westwater Publisher: Springer 1997

UNIT 10: Techniques behindAnimation

Structure:

1. Learning Objectives
2. Difference between film and animation
3. Principles of animation
4. Approaches of animation
5. Basic animation techniques
6. Unit Summary
7. Keywords
8. Exercise
9. References

---

1. Learning Objectives:

After studying this unit, you will be able to

- Know the difference between animation and film
- Explain about approaches and techniques of animation

1. Difference between film and animation:

Animation is made by exposing a series of pictures or frames, which results in an illusion of apparent movement. At this age of Technology, we have High Performance Workstation, which helped artists produce High Quality and more realistic animations. Animation movies is a result of one‘s vision and imagination. The most common examples of animation are the cartoon.

A **film**, also called a **movie** or **motion picture**, is a series of still images which, when shown on a screen, creates the illusion of moving images due to phi phenomenon. This optical illusion causes us to perceive continuous motion between separate objects viewed rapidly in succession. A film is created by photographing actual scenes with a motion picture camera; by photographing drawings or miniature models using traditional animation techniques; by means of CGI and computer animation; or by a combination of some or all of these techniques and other visual effects. Contemporary definition of cinema is the art of simulating experiences, that communicate ideas, stories, perceptions, feelings, beauty or atmosphere by the means of recorded or programmed moving images along with other sensory stimulations.

10.2 Principles of animation:

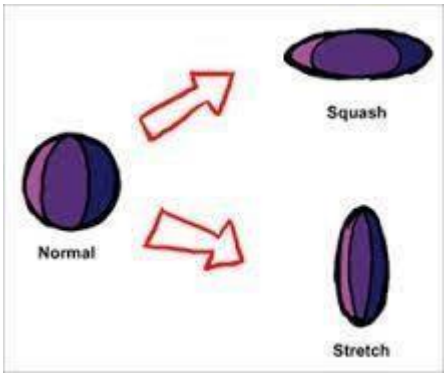
The following 12 basic principles of animation were developed by the 'old men' of Walt Disney Studios, amongst them Frank Thomas and Ollie Johnston, during the 1930s. Disney's Twelve Basic Principles of Animation is a set of principles of animation introduced by the Disney animators Ollie Johnston and Frank Thomas in their 1981 book The Illusion of Life: Disney Animation. Johnston and Thomas in turn based their book on the work of the leading Disney animators from the 1930s onwards, and their effort to produce more realistic animations. The main purpose of the principles was to produce an illusion of characters adhering to the basic

laws of physics, but they also dealt with more abstract issues, such as emotional timing and character appeal.

The book and some of its principles have been adopted by some traditional studios, and have been referred to by some as the "Bible of animation." In 1999 the book was voted number one of the "best animation books of all time" in an online poll. Though originally intended to apply to traditional, hand-drawn animation, the principles still have great relevance for today's more prevalent computer animation.

1. SQUASHAND STRETCH:

This action gives the illusion of weight and volume to a character as it moves. Also squash and stretch is useful in animating dialogue and doing facial expressions. How extreme the use of squash and stretch is, depends on what is required in animating the scene. Usually it's broader in a short style of picture and subtler in a feature. It is used in all forms of character animation from a bouncing ball to the body weight of a person walking. This is the most important element you will be required to master and will be used often.



2. ANTICIPATION:

This movement prepares the audience for a major action the character is about to perform, such as, starting to run, jump or change expression. A dancer does not just leap off the floor. A backwards motion occurs before the forward action is executed. The backward motion is the anticipation. A comic effect can be done by not using anticipation after a series of gags that used anticipation. Almost all real action has major or minor anticipation such as a pitcher's windup or a golfers' back swing. Feature animation is often less broad than short animation unless a scene requires it to develop a characters personality.

3. STAGING:

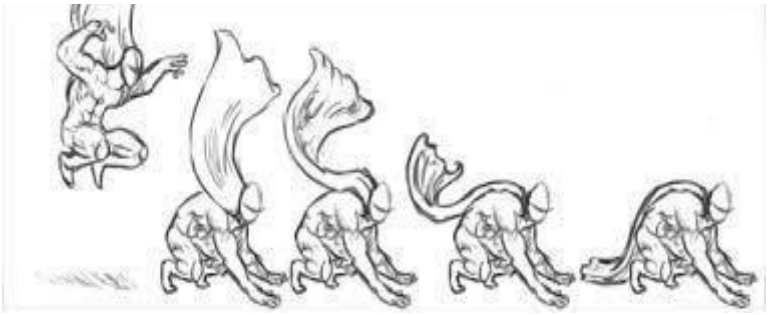
A pose or action should clearly communicate to the audience the attitude, mood, reaction or idea of the character as it relates to the story and continuity of the story line. The effective use of long, medium, or close up shots, as well as camera angles also helps in telling the story. There is a limited amount of time in a film, so each sequence, scene and frame of film must relate to the overall story. Do not confuse the audience with too many actions at once. Use one action clearly stated to get the idea across, unless you are animating a scene that is to depict clutter and confusion. Staging directs the audience's attention to the story or idea being told. Care must be taken in background design so it isn't obscuring the animation or competing with it due to excess detail behind the animation. Background and animation should work together as a pictorial unit in a scene.

4. STRAIGHT AHEAD AND POSE TO POSEANIMATION:

Straight ahead animation starts at the first drawing and works drawing to drawing to the end of a scene. You can lose size, volume, and proportions with this method, but it does have spontaneity and freshness. Fast, wild action scenes are done this way. Pose to Pose is more planned out and charted with key drawings done at intervals throughout the scene. Size, volumes, and proportions are controlled better this way, as is the action. The lead animator will turn charting and keys over to his assistant. An assistant can be better used with this method so that the animator doesn't have to draw every drawing in a scene. An animator can do more scenes this way and concentrate on the planning of the animation. Many scenes use a bit of both methods of animation.

5. FOLLOW THROUGH AND OVERLAPPINGACTION:

When the main body of the character stops all other parts continue to catch up to the main mass of the character, such as arms, long hair, clothing, coat tails or a dress, floppy ears or a long tail (these follow the path of action). Nothing stops all at once. This is follow through. Overlapping action is when the character changes direction while his clothes or hair continues forward. The character is going in a new direction, to be followed, a number of frames later, by his clothes in the new direction. "DRAG," in animation, for example, would be when Goofy starts to run, but his head, ears, upper body, and clothes do not keep up with his legs. In features, this type of action is done more subtly. Example: When Snow White starts to dance, her dress does not begin to move with her immediately but catches up a few frames later. Long hair and animal tail will also be handled in the same manner. Timing becomes critical to the effectiveness of drag and the overlapping action.



6. SLOW-OUT AND SLOW-IN:

As action starts, we have more drawings near the starting pose, one or two in the middle, and more drawings near the next pose. Fewer drawings make the action faster and more drawings make the action slower. Slow-ins and slow-outs soften the action, making it more lifelike. For a gag action, we may omit some slow-out or slow-ins for shock appeal or the surprise element. This will give more snap to the scene.

7.ARCS:

All actions, with few exceptions (such as the animation of a mechanical device), follow an arc or slightly circular path. This is especially true of the human figure and the action of animals. Arcs give animation a more natural action and better flow. Think of natural movements in the terms of a pendulum swinging. All arm movement, head turns and even eye movements are executed on an arcs.

8. SECONDARYACTION:

This action adds to and enriches the main action and adds more dimension to the character animation, supplementing and/or re-enforcing the main action. Example: A character is angrily walking toward another character. The walk is forceful, aggressive, and forward leaning. The leg action is just short of a stomping walk. The secondary action is a few strong gestures of the arms working with the walk. Also, the possibility of dialogue being delivered at the same time with tilts and turns of the head to accentuate the walk and dialogue, but not so much as to distract from the walk action. All of these actions should work together in support of one another. Think of the walk as the primary action and arm swings, head bounce and all other actions of the body as secondary or supporting action.

9. TIMING:

Expertise in timing comes best with experience and personal experimentation, using the trial and error method in refining technique. The basics are: more drawings between poses slow and smooth the action. Fewer drawings make the action faster and crisper. A variety of slow and fast timing within a scene adds texture and interest to the movement. Most animation is done on twos (one drawing photographed on two frames of film) or on ones (one drawing photographed on each frame of film). Twos are used most of the time, and ones are used during camera moves such as trucks, pans and occasionally for subtle and quick dialogue animation. Also, there is timing in the acting of a character to establish mood, emotion, and reaction to another character or to a situation. Studying movement of actors and performers on stage and in films is useful when animating human or animal characters. This frame by frame examination of film footage will aid you in understanding timing for animation. This is a great way to learn from the others.

10. EXAGGERATION:

Exaggeration is not extreme distortion of a drawing or extremely broad, violent action all the time. It‘s like a caricature of facial features, expressions, poses, attitudes and actions. Action traced from live action film can be accurate, but stiff and mechanical. In feature animation, a character must move more broadly to look natural. The same is true of facial expressions, but the action should not be as broad as in a short cartoon style. Exaggeration in a walk or an eye movement or even a head turn will give your film more appeal. Use good taste and common sense to keep from becoming too theatrical and excessively animated.

11. SOLID DRAWING:

The basic principles of drawing form, weight, volume solidity and the illusion of three dimension apply to animation as it does to academic drawing. The way you draw cartoons, you draw in the classical sense, using pencil sketches and drawings for reproduction of life. You transform these into color and movement giving the characters the illusion of three-and fourdimensional life. Three dimensional is movement in space. The fourth dimension is movement in time.

12. APPEAL:

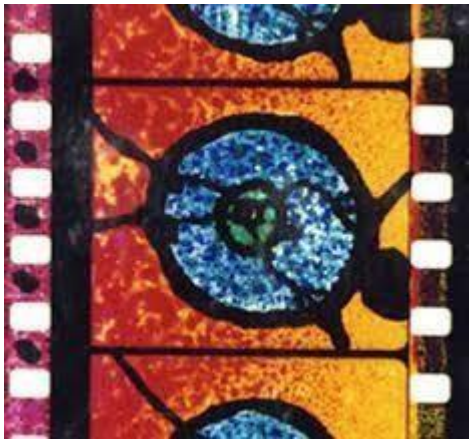
A live performer has charisma. An animated character has appeal. Appealing animation does not mean just being cute and cuddly. All characters have to have appeal whether they are heroic, villainous, comic or cute. Appeal, as you will use it, includes an easy to read design, clear drawing, and personality development that will capture and involve the audience's interest. Early cartoons were basically a series of gags strung together on a main theme. Over the years, the artists have learned that to produce a feature there was a need for story continuity, character development and a higher quality of artwork throughout the entire production. Like all forms of story telling, the feature has to appeal to the mind as well as to the eye.

---

3. Approaches of animation:

- Hydrotechnics: A technique that includes lights, water, fire, fog, and lasers, with highdefinition projections on mist screens.

- Drawn on film animation: A technique where footage is produced by creating the images directly on film stock, for example by Norman McLaren, Len Lye and Stan Brakhage.



- Paint-on-glass animation: A technique for making animated films by manipulating slow drying oil paints on sheets of glass, for example by Aleksandr Petrov.

- **Erasure animation**: A technique using traditional 2D media, photographed over time as the artist manipulates the image. For example, William Kentridge is famous for his charcoal erasure films, and Piotr Dumała for his auteur technique of animating scratches on plaster.
- Pinscreen animation: Makes use of a screen filled with movable pins that can be moved in or out by pressing an object onto the screen. The screen is lit from the side so that the pins cast shadows. The technique has been used to create animated films with a range of textural effects difficult to achieve with traditional cel animation.

- Sand animation: Sand is moved around on a back- or front-lighted piece of glass to create each frame for an animated film. This creates an interesting effect when animated because of the light contrast.

- Flip book: A flip book (sometimes, especially in British English, called a flick book) is a book with a series of pictures that vary gradually from one page to the next, so that when

the pages are turned rapidly, the pictures appear to animate by simulating motion or some other change. Flip books are often illustrated books for children, but may also be geared towards adults and employ a series of photographs rather than drawings. Flip books are not always separate books, but may appear as an added feature in ordinary books or magazines, often in the page corners. Software packages and websites are also available that convert digital video files into custom-made flip books.

- Character animation
- Multi-sketching
- Special effects animation

10.4 Techniques of animation:

There are four basic techniques used in animation. These are:

- Drawn animation
- Cut-out animation
- Model animation or stop motion animation
- Computer animation or computer generated imagery (CGI)

1. ***Drawn animation:***
2. *This covers any form where one drawing is replaced by another in a sequence. Each drawing is slightly different from the one before. It works the way a flipbook does. These animated films are made up of thousands of drawings which are shown on screen very quickly one after the other. It takes a very long time to film from start to finish and requires many animators to complete the work.*

1.1.1.3 Cut-out animation:

This covers any form of animation where cut-out shapes are moved around or replaced by other cut-outs. Flat objects like buttons, matchsticks and string can also be used in this form of animation. Cut-outs can also be laid on top of drawings. It is very quick and easy to do but difficult to have more than one or two objects moving at the same time. Cut-out animation can appear very stiff and awkward.

1.1.1.4 *Model or stop-motion animation:*

This involves the filming of three-dimensional models. The materials used could include plasticine, clay or wire - in fact anything that can be bent or formed into another shape. The puppets are positioned and filmed before being moved ever so slightly and filmed again. These shots are put together as a piece of film and will give the impression of the models moving.

Models can be used over and over again and copies made of them to shoot different scenes at the same time so that the filming takes less time. This type of animation needs a lot of time and hard work. The makers of 'James and the Giant Peach' were only able to complete 45 seconds of stop-motion animation a week - 10 seconds a day. This was because each puppet had so many joints that needed moving for each frame - the centipede alone had 72!

1.1.1.5 *Computer animation or Computer Generated Imagery (CGI):*

This refers to the drawing of three-dimensional models and sets on the computer. Images can be scanned into the computer using digital photography or made within the computer itself. Human characters can be built from clay whilst sets and furnishings are modelled using design systems similar to architects drawings. These models are scanned into the computer as wireframe models, which are gradually built up into a coloured and textured form.

Computer animation:

Computer animation encompasses a variety of techniques, the unifying factor being that the animation is created digitally on a computer. 2D animation techniques tend to focus on image manipulation while 3D techniques usually build virtual worlds in which characters and objects move and interact. 3D animation can create images that seem real to the viewer.

2D animation:

2D animation figures are created and/or edited on the computer using 2D bitmap graphics or created and edited using 2D vector graphics. This includes automated computerized versions of traditional animation techniques such as interpolated morphing, onion skinning and interpolated rotoscoping. 2D animation has many applications, including analog computer animation, Flash animation and PowerPoint animation. Cinemagraphs are still photographs in the form of an animated GIF file of which part is animated.

3D animation:

3D animation is digitally modeled and manipulated by an animator. The animator usually starts by creating a 3D polygon mesh to manipulate. A mesh typically includes many vertices that are connected by edges and faces, to give the visual appearance of form to a 3D object or 3D environment. Sometimes, the mesh is given an internal digital skeletal structure called an armature that can be used to control the mesh by weighting the vertices. This process is called rigging and can be used in conjunction with keyframes to create movement.

Other techniques can be applied, such as mathematical functions (e.g., gravity, particle simulations), simulated fur or hair, and effects such as fire and water simulations. These techniques fall under the category of 3D dynamics.

---

5. Unit Summary:

This unit introduced the principles and approaches to animation. The unit introduced the different techniques for animation.

6. Keywords:

Flim, Animation, Principles of Animation,Approaches of Animation, Techniques of Animation.

7. Exercise:

1) How animation is used in film making?
2) Explain the key principles of animation.
3) List and explain the approaches to animation.
4) Explain the different techniques to animation.

8. References:

1.Prabat K Andleigh and Kiran Thakrar, ―Multimedia Systems and Design‖, PHI, 2003. (UNIT 3 to 5)

2.Donald Hearn and M.Pauline Baker, ―Computer Graphics C Version‖, Pearson Education, 2003.

3. Designing Interfaces by Jenifer Tidwell
4. Digital Multimedia by Nigel Chapman

5.Information Architecture for the World Wide Web: Designing Large-Scale Web Sites by Peter Morville, Louis Rosenfeld

6. Information Visualization, Second Edition: Perception for Design by Colin Ware
7. Letting Go of the Words: Writing Web Content that Works by Janice Redish
8. Multimedia Applications (X.media.publishing) by Ralf Steinmetz
9. Now You See It: Simple Visualization Techniques for Quantitative Analysis by Stephen Few
10. Visual Thinking: for Design by Colin Ware

11.Writing for New Media, Third Edition: Content Development for Bloggers and Professionals by Timothy Paul Garrand

12. Real-Time Video Compression: Techniques and Algorithms By Raymond Westwater
Publisher: Springer 1997

UNIT 11: Advanced animation techniques

Structure:

1. Learning Objectives
2. Advanced animation techniques
3. Bitmapped and shape elements
4. Recording animation
5. Unit Summary
6. Keywords
7. Exercise
8. References

1. Learning Objectives:

After studying this unit, you will be able to

- Know about advanced animation techniques
- Elucidate about advanced bitmapped and shape elements

1. Advanced animation techniques:

There are various types of animation techniques practiced by film makers all over the world. Classical and digital 2D animation, digital 3D Animation, stop-motion, clay animation, cut-out animation, paint-on-glass animation, drawn-on-film animation, and experimental animation are just a few among the many existing forms of animation.

Deciding a suitable animation technique for your project depends on various factors that include budget, look and feel, output quality, aesthetic, stylization, and story requirements.

Classical 2D animation is also known as hand-drawn 2D animation or traditional animation. In this technique animators need to make at least 12 drawings on paper for one second length of film. The drawings are later scanned or captured for post-production using computer. This technique was the dominant form of animation in film and TV series, until the development of CGI animation.

In digital 2D animation technique, animation frames are drawn directly on software using mouse or pen tablet. This technique is used mostly for TV series and web animation.

Digital 3D animation is one of the sought-after techniques in the current scenario. Using this technique, 3D models are created, textured, rigged, and animated in the virtual space.

In stop-motion animation, one needs to set the character or object, in the desired state or pose against its background to expose a frame, and then do slight modifications in progression and take another frame. The process is repeated until the desired length of animation is achieved and shot.

Clay animation is one of the many forms of stop-motion animation. Because of its popularity and the extensive use of clay, usually plasticine, it is recognized as an independent technique and genre.

Cut-out animation is a stop-motion technique for producing animations, using flat characters, props and backgrounds made out of different materials such as paper, card, stiff fabric or even photographs.

Paint-on-glass animation is a technique for making animated films by manipulating slowdrying oil paints on sheets of glass.

Drawn-on-film animation is also known as direct animation or animation without camera. Here footage is produced by creating images directly on film stock, as opposed to any other form of animation where the images or objects are photographed frame by frame with an animation camera.

Experimental animation has no limitation in terms of techniques or ideas. Animators use their instinct to use materials of their choice to achieve the final animation.

11.2 Bitmapped and shape elements:

Vector graphics is the use of geometrical primitives such as points, lines, curves, and shapes or polygons all of which are based on mathematical expressions—to represent images in computer graphics. Vector graphics are based on vectors (also called paths), which lead through locations called control points or nodes. Each of these points has a definite position on the x and y axes of the work plane and determines the direction of the path; further, each path may be assigned a stroke color, shape, thickness, and fill. These properties don't increase the size of

vector graphics files in a substantial manner, as all information resides in the document's structure, which describes solely how the vector should be drawn.

The term vector graphics is typically used only for 2D (planar) graphics objects, in order to distinguish them from 2D raster graphics, which are also very common. 3D graphics as commonly implemented today (e.g. in OpenGL) are typically described using primitives like 3D points and polygons connecting these (which in turn describe surfaces); these 3D primitives are much more similar to vector graphics than to raster graphics, but aren't explicitly called vector graphics. The equivalent of raster graphics in the 3D world are voxel-based graphics.

11.3 Recording animation:

The creation of non-trivial animation works (i.e., longer than a few seconds) has developed as a form of filmmaking, but with certain unique aspects. One thing live-action and animated feature-length films do have in common is that they are both extremely labor-intensive and horrendously expensive.

The most important difference is that once a film is in the production phase, the marginal cost of one more shot is much, much higher for animated films than for live-action films. It is relatively easy for a director to ask for one more take during principal photography of a liveaction film, but *every* take on an animated film must be manually rendered by animators (although the task of rendering slightly different takes has been made less tedious by modern computer animation). It is pointless for a studio to pay the salaries of dozens of animators to spend weeks creating a visually dazzling five-minute scene, if that scene fails to effectively advance the plot of the film. Thus, animation studios starting with Disney began the practice in the 1930s of maintaining story departments where storyboard artists develop every single scene through storyboards, then handing the film over to the animators only after the production team is satisfied that all the scenes will make sense as a whole. While live-action films are now also storyboarded, they necessarily enjoy much more latitude to depart from storyboards (i.e., realtime improvisation).

Another problem unique to animation is the necessity of ensuring that the style of an animated film is consistent from start to finish, even as films have grown longer and teams have grown larger. Animators, like all artists, necessarily have their own individual styles, but must

subordinate their individuality in a consistent way to whatever style was selected for a particular film. Since the early 1980s, feature-length animated films have been created by teams of about 500 to 600 people, of whom 50 to 70 are animators. It is relatively easy for two or three artists to match each other's styles, but it is much harder to keep dozens of artists synchronized with one other.

This problem is usually solved by having a separate group of visual development artists develop an overall look and palette for each film before animation begins. Character designers on the visual development team draw model sheets to show how each character should look like with different facial expressions, posed in different positions, and viewed from different angles. On traditionally animated projects, maquettes were often sculpted to further help the animators see how characters would look from different angles.

Unlike live-action films, animated films were traditionally developed beyond the synopsis stage through the storyboard format; the storyboard artists would then receive credit for writing the film. In the early 1960s, animation studios began hiring professional screenwriters to write screenplays (while also continuing to use story departments) and such screenplays had become commonplace for animated films by the late 1980s.

4. Unit Summary:

This unit introduced the about advanced animation techniques.

5. Keywords:

Advanced techniques, Bitmapped and Shape elements, Recording Animation.

6. Exercise:

1) Write short note on bitmapped and shape elements.
2) Explain advanced animation techniques.

11.7 References:

1.Prabat K Andleigh and Kiran Thakrar, ―Multimedia Systems and Design‖, PHI, 2003. (UNIT 3 to 5)

2.Donald Hearn and M.Pauline Baker, ―Computer Graphics C Version‖, Pearson Education, 2003.

3. Designing Interfaces by Jenifer Tidwell

4. Digital Multimedia by Nigel Chapman

5.Information Architecture for the World Wide Web: Designing Large-Scale Web Sites by Peter Morville, Louis Rosenfeld

6. Information Visualization, Second Edition: Perception for Design by Colin Ware

7. Letting Go of the Words: Writing Web Content that Works by Janice Redish

8. Multimedia Applications (X.media.publishing) by Ralf Steinmetz

9. Now You See It: Simple Visualization Techniques for Quantitative Analysis by Stephen Few

10. Visual Thinking: for Design by Colin Ware

11.Writing for New Media, Third Edition: Content Development for Bloggers and Professionals by Timothy Paul Garrand

12. Real-Time Video Compression: Techniques and Algorithms By Raymond Westwater Publisher: Springer 1997

UNIT -12: Classification of Animation

Structure:

1. Learning Objectives
2. Classification of Animation,
3. Difference between conventional method of animation and digital animation
4. Types of animation
5. Unit Summary
6. Keywords
7. Exercises
8. References

1. Learning Objectives:

After studying this unit, you will be able to

- Understand about Classification of Animation
- Discuss about conventional and digital animation
- Explain different types of animation

12.1. Classification of Animation:

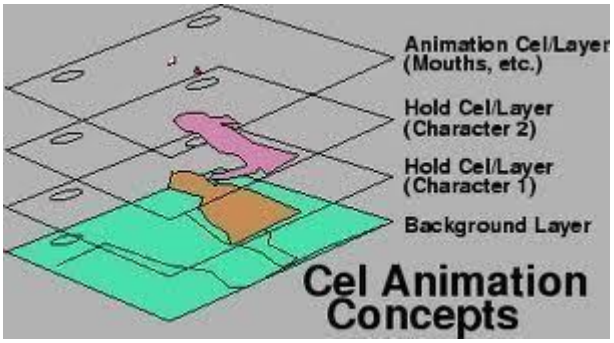
Basic Types of Animation:

The basic types of animation are cel, stop and computer animation. These three types of animation are the primary keynote for animation effect.

Cel Animation:

Cel animation refers to the traditional way of animation in a set of hand drawings. In this process, various pictures are created which are slightly different but progressive in nature, to depict certain actions. Trace these drawings on a transparent sheet. This transparent sheet is known as cel and is a medium for drawing frames. Now draw outlines for the images and color

them on the back of the cel. The cel is an effective technique that helps to save time by combining characters and backgrounds. You can also put the previous drawings over other backgrounds or cels whenever required. Here, you need not draw the same picture again as it has the facility of saving previous animations that can be used when required. Coloring a background may be a more difficult task than a single drawing, as it covers the whole picture. Background requires shading and lighting and will be viewed for a longer duration. Then use your camera to photograph these drawings. Today, cel animations are made more attractive by using the drawings together with music, matching sound effects and association of timing for every single effect. E.g. To display a cartoon show, 10-12 frames are played in rapid succession per second to give a representation of movement in a cel animation.



StopAnimation:

Stop animation or stop motion animation is a technique to make objects move on their own. Here, a few images are drawn with some different positions and photographed separately. Puppetry is one of the most used frame-to-frame animation types. Some famous movies that are animated via stop motion effects are King Kong, The Dinosaur and the

ComputerAnimation:

Computer Animation is the latest technique that includes 2D and 3D animation. These not only enhance the hand-drawn characters but also make them appear real as compared to the above mentioned animations.

2D Animation: It is used through Powerpoint and Flash animations. Though its features are similar to cel animation, 2D animation has become popular due to simple application of scanned drawings into the computer like in a cartoon film.

3D Animation: It is used in filmmaking where we require unusual objects or characters that are not easy to display. Use of 3D animation can create a crowd of people in a disaster like earthquake, flood or war. There are different shapes, support of mathematical codes, display of actions and colors which are mind-blowing as if copied from an actual picture.

The above mentioned 3 types of animations have brought a new era of amazing technology in the field of Internet (website design and graphics), film industry and media. Many of us can try out making their own animation! In addition, animation is one of the popular Internet marketing strategies that make visitors stay on your site for a longer time.

12.2 Difference between conventional method of animation and digital animation:

In truth it's easy to distinguish between the two: traditional animation uses methods that don't involve any kind of digital tools, while computer animation methods use - you guessed it computers. Another way of distinguishing the two is physical vs. virtual; traditional animation uses physical materials and activities, while computer animation uses virtual materials in a digital space.

2D cel animation and stop-motion animation both fall under the category of traditional animation, even if both may use digital methods of filming in the end. What matters is the method of producing the animation itself; cel animation generally involves hand-drawing, handinking, and hand-painting each frame on physical paper and cels, while stop-motion involves working with physical models and objects captured on camera one frame at a time.

Computer animation can be either 2D or 3D. 2D computer animation often involves a virtualization of the traditional 2D animation workspace, bringing pen and paper into the digital environment to recreate cartoon animation workflows and styles. 3D computer animation tends to involve a hybrid of workflows following traditional timelines adapted to working in a virtual 3D space. Either way, if you're animating on-screen you're working with computer animation.

In many cases it's hard to classify an animation as one or the other, as many animators take a hybrid path in which some parts of an animation are produced using traditional styles, then completed or enhanced using digital methods.

Traditional animation (also called cel animation or hand-drawn animation) was the process used for most animated films of the 20th century. The individual frames of a traditionally animated film are photographs of drawings, first drawn on paper. To create the illusion of movement, each drawing differs slightly from the one before it. The animators' drawings are traced or photocopied onto transparent acetate sheets called cels, which are filled in with paints in assigned colors or tones on the side opposite the line drawings. The completed character cels are photographed one-by-one against a painted background by a rostrum camera onto motion picture film.

The traditional cel animation process became obsolete by the beginning of the 21st century. Today, animators' drawings and the backgrounds are either scanned into or drawn directly into a computer system. Various software programs are used to color the drawings and

simulate camera movement and effects. The final animated piece is output to one of several delivery media, including traditional 35 mm film and newer media such as digital video. The "look" of traditional cel animation is still preserved, and the character animators' work has remained essentially the same over the past 70 years. Some animation producers have used the term "tradigital" to describe cel animation which makes extensive use of computer technology.

Examples of traditionally animated feature films include Pinocchio (United States, 1940), Animal Farm (United Kingdom, 1954), and L'Illusionniste (British-French, 2010). Traditionally animated films which were produced with the aid of computer technology include The Lion King (US, 1994), Akira (Japan, 1988), Sen to Chihiro no Kamikakushi (Spirited Away) (Japan, 2001), Les Triplettes de Belleville (France, 2003), and The Secret of Kells (Irish-French-Belgian, 2009).

- Full animation refers to the process of producing high-quality traditionally animated films that regularly use detailed drawings and plausible movement, having a smooth animation. Fully animated films can be made in a variety of styles, from more realistically animated works such as those produced by the Walt Disney studio (Beauty and the Beast, Aladdin, Lion King) to the more 'cartoon' styles of the Warner Bros. animation studio. Many of the Disney animated features are examples of full animation, as are non-Disney works such as The Secret of NIMH (US, 1982), The Iron Giant (US, 1999), and Nocturna (Spain, 2007).
- Limited animation involves the use of less detailed and/or more stylized drawings and methods of movement usually a choppy or "skippy" movement animation. Pioneered by the artists at the American studio United Productions of America, limited animation can be used as a method of stylized artistic expression, as in Gerald McBoing Boing (US, 1951), Yellow Submarine (UK, 1968), and much of the anime produced in Japan. Its primary use, however, has been in producing cost-effective animated content for media such as television (the work of Hanna-Barbera, Filmation, and other TV animation studios) and later the Internet (web cartoons).
- Rotoscoping is a technique patented by Max Fleischer in 1917 where animators trace liveaction movement, frame by frame. The source film can be directly copied from actors' outlines into animated drawings, as in The Lord of the Rings (US, 1978), or used in a stylized and expressive manner, as in Waking Life (US, 2001) and A Scanner Darkly (US, 2006). Some other examples are: Fire and Ice (US, 1983) and Heavy Metal (1981).
- Live-action/animation is a technique combining hand-drawn characters into live action shots. One of the earlier uses was in Koko the Clown when Koko was drawn over live action

footage. Other examples include Who Framed Roger Rabbit (US, 1988), Space Jam (US, 1996) and Osmosis Jones (US, 2001).

Computer animation encompasses a variety of techniques, the unifying factor being that the animation is created digitally on a computer. 2D animation techniques tend to focus on image manipulation while 3D techniques usually build virtual worlds in which characters and objects move and interact. 3D animation can create images that seem real to the viewer.

2D animation:

2D animation figures are created and/or edited on the computer using 2D bitmap graphics or created and edited using 2D vector graphics. This includes automated computerized versions of traditional animation techniques such as interpolated morphing, onion skinning and interpolated rotoscoping. 2D animation has many applications, including analog computer animation, Flash animation and PowerPoint animation. Cinemagraphs are still photographs in the form of an animated GIF file of which part is animated.

2D Terms

• Final line advection animation, a technique that gives the artists and animators a lot more influence and control over the final product as everything is done within the same department:

In Paperman, we didn‘t have a cloth department and we didn‘t have a hair department. Here, folds in the fabric, hair silhouettes and the like come from of the committed design decision-making that comes with the 2D drawn process. Our animators can change things, actually erase away the CG underlayer if they want, and change the profile of the arm. And they can design all the fabric in that Milt Kahl kind-of way, if they want to.

3D animation:

Main articles: Computer animation and 3D computer graphics

3D animation is digitally modeled and manipulated by an animator. The animator usually starts by creating a 3D polygon mesh to manipulate. A mesh typically includes many vertices that are connected by edges and faces, to give the visual appearance of form to a 3D object or 3D environment. Sometimes, the mesh is given an internal digital skeletal structure called an

armature that can be used to control the mesh by weighting the vertices. This process is called rigging and can be used in conjunction with keyframes to create movement.

Other techniques can be applied, such as mathematical functions (e.g., gravity, particle simulations), simulated fur or hair, and effects such as fire and water simulations. These techniques fall under the category of 3D dynamics.

3DTerms

- Cel-shaded animation is used to mimic traditional animation using CG software. Shading looks stark, with less blending of colors. Examples include, Skyland (2007, France), Appleseed Ex Machina (2007, Japan), The Legend of Zelda: Wind Waker (2002, Japan)
- Machinima – Films created by screen capturing in video games and virtual worlds.
- Motion capture is used when live-action actors wear special suits that allow computers to copy their movements into CG characters. Examples include Polar Express (2004, US), Beowulf (2007, US),A Christmas Carol (2009, US), The Adventures of Tintin (2011, US)
- Photo-realistic animation is used primarily for animation that attempts to resemble real life, using advanced rendering that mimics in detail skin, plants, water, fire, clouds, etc. Examples include Up (2009, US), Kung-Fu Panda (2008, US), Ice Age (2002, US).

12.3 Types of animation:

Different Types of Animation:

At the broadest sense, there are 3 types of animation:
2D, 3D and Stop Motion
Any way to manipulate a sequence of images, frame by frame, is considered a Type of Animation. All animations falls into one of these three categories. The boundaries between them are, however, blending with great speed.

2D animation:

The term "2D" refers to animation that is created using two dimensional drawings.Classic hand drawn animation is the main example for this type. Think Disney features, like the recent

Princess and the Frog. Computer assisted animation Flash or AfterEffects cut-out animation is also considered 2D. Many TV series are done in Flash these days.

3D animation:

"3D" refers to Computer Generated Images (CGI) that create the illusion of three dimensional space with great accuracy. Films like Toy Story and Up are 3D - CGI animation.

Computer special effects, also fall within this category. All the magical creatures and powerful spells in Harry Potter are done this way. We should bear in mind though, that in the end, 3D animation is also just a sequence of flat, two dimensional images projected on the screen.

Stereoscopic 3D, such as in Avatar, is the combination of two slightly different images that create the spatial effect, just like our own eyes do in the real world. It is not a type of animation though, but rather a type of projection, like DVD, or a flip book.

Stop Motion Types of animation:

Everything that is shot live, frame by frame, in front or under a camera is called Stop Motion or "Stop Frame" Animation. Clay animations are usually the first thing that comes to mind in this category. The different stop motion types obviously looks very three dimensional, the same way video looks three dimensional, but the term "3D" is reserved for computer animation only. Some physical special effects (big parts of the original Star Wars trilogy) are done in stop motion.

Stop-motion has a special and powerful appeal of its own; the reality of the objects on screen enhances our expectations of them. Pixilation is the most direct use of this power animating real, live humans.

WebAnimation:

A general name for the types of small animations that illustrate or decorate websites.Usually Gif or Flash, you find these on blogs, in forums, in MySpace, avatars that web surfers use, animated logos, banners that promote just about anything.Well, It MOVES, you know?

JabAnimation:

The name of an online animation site where you can upload you photo and insert yourself into a funny pre-existing animation. This seems to be turning into a generic name for this kind of web application.Cinematic and Animated Story Boards.Also called animatik, animatique, or videoboards,These are rudimentary forms of a movie, which serve as a pre-production tool for complicated projects. They are used for planning both live action and animation projects, from commercials to feature films. The more complex the production, the more detailed the animatik will be. Good and detailed planning saves a lot of cash later!

VJ Animation:

A visual experience that accompanies the music in clubs, concerts, weddings and so on.Usually abstract graphics that run in loops, but occasionally you can come across more complex creations.

ExperimentalAnimation:

There's a new type of animation I've seen a lot lately - Post-It Animation. Animators use the colorful post-it squares as pixels, arrange them on a wall and animate them around.

This sort of imaginative use of an ordinary object is the heart of Experimental Animation. You see, you really CAN animate practically anything.Food, furniture, pin screens, beads, chalk marks on black board, people, puppets, mixed techniques, layered images, earth, plastic, wool ANYTHING! It's usually students who have the time, hunger, curiosity and most importantly facilities, to play around and invent a new type of animation.

These works rarely reach the cinema; most of them are short films of no more than a few minutes.

4. Unit Summary:

This unit provides a brief introduction about animation classification and types.

5. Keywords:

Classification of Animation, Conventional method , Digital Animation, Animation Types.

12.6 Exercise:

1) Explain the classification of animation.

2) Explain basic types of animation.

12.7 References:

1.Prabat K Andleigh and Kiran Thakrar, ―Multimedia Systems and Design‖, PHI, 2003. (UNIT 3 to 5)

2.Donald Hearn and M.Pauline Baker, ―Computer Graphics C Version‖, Pearson Education, 2003.

3. Designing Interfaces by Jenifer Tidwell
4. Digital Multimedia by Nigel Chapman

5.Information Architecture for the World Wide Web: Designing Large-Scale Web Sites by Peter Morville, Louis Rosenfeld

6. Information Visualization, Second Edition: Perception for Design by Colin Ware
7. Letting Go of the Words: Writing Web Content that Works by Janice Redish
8. Multimedia Applications (X.media.publishing) by Ralf Steinmetz
9. Now You See It: Simple Visualization Techniques for Quantitative Analysis by Stephen Few
10. Visual Thinking: for Design by Colin Ware

11.Writing for New Media, Third Edition: Content Development for Bloggers and Professionals by Timothy Paul Garrand

12.Real-Time Video Compression: Techniques and Algorithms By Raymond Westwater Publisher: Springer 1997

Module- 4

Unit 13: Animation and File Formats

Structure:

1. Learning Objectives
2. Animation and file formats
3. Hardware and software requirements
4. Difference between 2D and 3D animation film, cartoon movie, animation and broadcasting
5. Unit Summary
6. Keywords
7. Exercises
8. References

9. Learning Objectives:

After studying this unit, you will be able to

- Explain about Various file formats of animated files
- Discuss about hardware and software required for animation
- Understand difference between 2D and 3D animation.

1. Animation and file formats:

The format used for the animation is entirely based around how complex the animation is, where the animation is designed to be used; such as a webpage, and how large the file is for the animation. Different formats for animations include animated GIFs, Dynamic HTML such as Java, Shockwave and Flash.

Dynamic HTML:

Dynamic HTML is a type of HTML, which is used along side client scripting languages, such as Java Script. It can be used to create simple animations such as a button and drop down menus. Although a massive downside to using Dynamic HTML is since there are various types of browsers such as Google Chrome and FireFox, information is displayed differently meaning Dynamic HTML may not always be effective.

Flash:

Flash allows users to create animations frame by frame, including using certain materials such as bitmaps and scripting such as in ActionScript. Flash is also useful for animations since it can include the use of sound and video. It is also possible to create a website using Flash, however since the entire website uses flash it may require high internet speeds in order to load. Flash also needs a Flash player installed onto the computer in order to show content, although most computers already tend to have this installed. Flash files (FLA) have to be converted into SWF format before they can be used on the internet. The SWF format was originally composed to store animations in, including sounds, however it can also be used for other purposes such as website building and process forms.

Shockwave:

Shockwave is used more for 3D graphics and streaming videos. It is designed to work with a Director application, which can then compile various types of assets into a multimedia product, on a much larger scale then Flash.

Animated GIFs:

GIF images which are animated normally have various images combined into a single GIF file. Applications, such as GIF89A, cycle through the various images in the GIF file in order to create an animation. While GIFs have a limited amount of flexibility and less amount of control compared to other formats, although since it is supported by almost every single web browser, it has become extremely popular. GIF files also tend to be smaller then other animation files.

Animation File Formats:

Short Name	**Long Name**	**Typical File Extensions**	**Content**
ANI	Microsoft Windows Animated Cursor	.ani	Color/Animated Cursor
APNG	Animated Portable Network Graphics	.png	Bitmaps/Animation

EVA	Extended Vector Animation	.eva	Vector Animation
FLA	Adobe (Macromedia) Flash FLA Project File Format	.fla	Animation/Multimedia/Video
FLC	FLC	.flc	Animation
FLI	FLI	.fli	Animation
GIF	Graphics Interchange Format	.gif	Bitmaps/Animation
MNG	Multi-image Network Graphics	.mng	Bitmaps/Animation
SWF	Small Web Format (Flash) - originally "ShockWave Flash"	.swf	Animation/Multimedia/Video
SWI	SWiSH Project File	.swi	Animation/Multimedia/Video
WebP	WebP	.webp	Bitmap/Animation

13.2 Hardware and software requirements:

Hardware essentials for computer animation:

Hardware requirements to create a successful (but basic) computer animation are fairly non-demanding with only a monitor, a mouse and a decent computer being needed to create impressive digital animation (far from the times of the past). However, with more specialised equipment animation can be brought to a new level of impressiveness. A list of some hardware basics and advanced –

Computer, monitor & mouse – These basic requirements are the staple for any beginner in animation and professional alike. The computer to run the sometimes component demanding animation software, the monitor to display and adjust your animations and the mouse to navigate the software.

Image/video capturing device – This is required to do such animation as stop-motion and to display CGI effects in a real life backdrop (such as explosions generated with appropriate software and displayed in captured footage) Although not essential, is still handy to have.

Graphics tablet and stylus – Using these, drawing more detailed characters; objects is made easier. This hardware is also familiar & more user friendly with artist who are used to drawing on paper. Not all animation software supports this feature often needing the objects drawn elsewhere such as Illustrator and then imported into the animation software.

Motion capture suites – These suites allow for an actor‘s motions and facial expression to be tracked via the suites built in sensors. This data collected by the suite can be used to animate a created character with life-like movements. Not for amatuers but still cool.

External HD – Animation, especially long in duration or high in quality can take up a large amount of memory and so its handy to have a means to store data.

Software Requirements for Animation:

The following software capabilities are highly desirable in a multimedia computer system with animation facility:

1. **Animation creation software: -** It allows the user to create animation sequences from scratch by using a mouse and various simple objects, such as lines, circles and polygons, with various supporting colors. For example, an animation sequence to illustrate bouncing of a ball can be created as follows○ Select a circle object and fill it with red color.
    - Then start the recorder of the animation creation software.
    - Then select the colored circle and drag it around the screen by using a mouse, recording several positions for it on the screen (each time you release the mouse button, you record the object‘s position on the screen.
    - Finally, stop the recorder and use save command to save the animation.
2. **Screen capture software: -** It is used to capture the displays of a computer screen as graphic images, which can be used in animation.
3. Animation **clips: -** This is a library of animation clips from which one can select and directly import an animation clip and use it in a multimedia application, saving the time and effort, which might otherwise be required to create a similar animation clip. This is also useful. if one does not

have an animation creation software. An animation clips library often provides the facility to add a new animation clip or delete an existing animation clip from the library.

4. **Animation file importing:-** The task of creating a multimedia application incorporating animation can often be greatly simplified, if the application software can import animation files in some standard formats. Common animation file formats include.FLI and FLC.
5. **Software support for high resolution: -** If the animation sequences of a multimedia application are made up of very high quality images, it is important to have not only the necessary hardware, but also software support for displaying high resolution images, which have many colors to maintain the high quality of animation display.
6. **Recording and playback capability: -** It allows the user to control the recording and display of an animation sequence. For example, it provides the user with options to _pause' and _replay' the animation sequence.
7. **Transition effects: -** Animation can be even more interesting, if it is enhanced with transition effects, such as fade-in and fade-out, layering, zooming and rotation of objects.

13.3 Difference between 2D and 3D animation - film, cartoon movie, animation and broadcasting:

Difference between 2D and 3D animation:

Fig: 2D and 3D Teddy Bear

2 dimensional animations deals more with drawing and framing and is the base of 3D animation. 2D animators create frames to define a sequence that are then moved at varying speeds to create the illusion of motion.3D or three-dimensional animation has more depth and is more realistic The texture, lighting and colour of the 3D objects are modified using software programs Film, advertising, video and gaming are the fast growing fields for 3D animation.

2D animation is a traditional animation process that has been in existence for hundreds of years. Modern 2D animated videos can be created either by hand or via computer and requires one image to be followed by another in a slightly different position, followed by yet another image in another position and so on in order to create movement.

They produce a flat image that has movement and is represented in the height and width dimensions, but not depth. Before the use of computer technology became the primary vehicle for image creation and manipulation, these images were created by hand.

Artists drew pencil sketches of every frame which were then transferred in full color onto cels. Each cel was painstakingly drawn and painted and then photographed against a background image in order to create the film‘s frames. The thousands of photographs are compiled and edited to run at the traditional 24 frames per second required of motion pictures.

Today, most 2D animated movies are created using computer animation software, though to varying degrees. Even if the animated images are hand drawn, they are often colorized and transferred to the cels using computers. Others are created entirely on computers.

3D animation differs significantly from 2D, both in terms of the process used to create it but the end result as well. 3D animation produces a more realistic character that is represented in all three dimensions (height, width and depth). All 3D animation with the exception of stop motion animation is created using computer software applications.

The main benefit of 3D animation over 2D animation, beyond the obvious ability to create a more lifelike character with realistic textures, is that 3D objects, once created, can be treated much like a physical object or an actor.

You can change the lighting or move the camera, much like you can in traditional motion picture filming, by simply dragging the object and moving it within the application. With 2D animation, everything is drawn including the camera angle and light source, making it much more difficult to change a scene during the editing process.

Difference between Cartoon and animation:

In essence, there is truly little difference between "traditional" animation or cartoons and computer animation; the primary difference is in the tools used to create these animations, the cost and effort involved in the processes, and the quality of the final output.

Traditional animation is a very hands-on process. This requires a team of artists, cleanup artists, painters, directors, background artists, and film/camera crews, along with the storyboard

artists and script writers to work out the original concepts; for large-scale projects, the amount of time, labor, and equipment involved can be staggering.

Traditional 3D animation was less "3D" and more still-lifes of claymations done by use of stop-motion filming techniques; the true concept of 3D animation didn't really blossom until the use of computers in animation became more practical. Computer animation removes the need for many of the extra tools required to create an animation; all you need, in general, is a computer with enough system requirements to run the 2D or 3D

13.4 Unit Summary:

This unit provides a brief detail about animation file formats , hardware and software requirements for animation finally some difference between cartoon and animation .

5. Keywords:

File Formats, Hardware requirements, Software requirements, 2D Animation, 3D Animation,

6. Exercises:

1) List out the various animation file formats.

2) What are the hardware and software requirements for animation.

3) List the diference between 2D and 3D animation.

4) Write note on cartoon movie.

13.7 References:

~~1.Prabat K Andleigh and Kiran Thakrar, —~~Multimedia ~~Systems and~~ Design‖, ~~PHI, 2003. (UNIT 3 to~~ 5)

2.Donald Hearn and M.Pauline Baker, ―Computer Graphics C Version‖, Pearson Education, 2003.

3. Designing Interfaces by Jenifer Tidwell

4. Digital Multimedia by Nigel Chapman

5.Information Architecture for the World Wide Web: Designing Large-Scale Web Sites by Peter Morville, Louis Rosenfeld

6. Information Visualization, Second Edition: Perception for Design by Colin Ware
7. Letting Go of the Words: Writing Web Content that Works by Janice Redish
8. Multimedia Applications (X.media.publishing) by Ralf Steinmetz
9. Now You See It: Simple Visualization Techniques for Quantitative Analysis by Stephen Few
10. Visual Thinking: for Design by Colin Ware
11.Writing for New Media, Third Edition: Content Development for Bloggers and Professionals by Timothy Paul Garrand
12.Real-Time Video Compression: Techniques and Algorithms By Raymond Westwater Publisher: Springer 1997

Unit 14: Animation Types

Structure:

1. Learning Objectives
2. Animation Types
3. Authoring tool, presentations, applications, interaction
4. 2D and 3D animations- projects simple animations
5. Unit Summary
6. Keywords
7. Exercise
8. References

14.0 Learning Objectives:

After studying this unit, you will be able to

- Understand about animation types
- Discuss about change applications, authoring tools

14.1 Animation Types:

Basic Types of Animation:

Animation originates from the creativity of intelligent people that is displayed through different forms of media. Let's take a look into the the types of animation... Mickey Mouse, Donald Duck, The Simpsons - are favorite cartoons of kids as well as adults. All these cartoon characters are the creation of the wonderful art of animation that captivates our eyes and makes our childhood days full of fun. How are these cartoons displayed on television or Internet? Let‘s find out…..

Basic animation is an easy and single keyframe animation. It is a presentation of various displays and movements, which adds liveliness to your site or film. The Internet users are usually fond of browsing a website that is well animated with good graphics. A web designer cannot ideate his website without the application of basic animation, due to its virtual advantages in the Internet market. In simple words, basic animation is the illusion of different movements, linked together in a proper way so that visitors/audiences get the effect of seeing a well coordinated set of actions.

Generally, this beautiful animation art is created using the Java language. For example. If you want to show a bouncing ball, you need to draw various positions of the ball in different drawings or 'frames' as they are called. In the first drawing, you can show the ball resting on the ground, in the second frame, the ball slightly above the ground, the third one will show the ball 2 - 3 feet above the ground, in the fourth one the ball will come down a bit and so on till finally the ball is on the ground. These drawings are composed together with the help of computer scanning, use of software, matching sound effects, time management and shooting with a camera. In the final result you will find an animation similar to the live action of a boy bouncing the ball up and down on the ground.

3 Basic Types of Animation

The basic types of animation are cel, stop and computer animation. These three types of animation are the primary keynote for animation effect.

Cel Animation

Cel animation refers to the traditional way of animation in a set of hand drawings. In this process, various pictures are created which are slightly different but progressive in nature, to depict certain actions. Trace these drawings on a transparent sheet. This transparent sheet is

known as cel and is a medium for drawing frames. Now draw outlines for the images and color them on the back of the cel. The cel is an effective technique that helps to save time by combining characters and backgrounds. You can also put the previous drawings over other backgrounds or cels whenever required. Here, you need not draw the same picture again as it has the facility of saving previous animations that can be used when required. Coloring a background may be a more difficult task than a single drawing, as it covers the whole picture. Background requires shading and lighting and will be viewed for a longer duration. Then use your camera to photograph these drawings. Today, cel animations are made more attractive by using the drawings together with music, matching sound effects and association of timing for every single effect. E.g. To display a cartoon show, 10-12 frames are played in rapid succession per second to give a representation of movement in a cel animation.

Stop Animation:

Stop animation or stop motion animation is a technique to make objects move on their own. Here, a few images are drawn with some different positions and photographed separately. Puppetry is one of the most used frame-to-frame animation types. Some famous movies that are animated via stop motion effects are King Kong, The Dinosaur and the Missing Link, The Curse of the Were-Rabbit and the Lost World.

ComputerAnimation:

Computer Animation is the latest technique that includes 2D and 3D animation. These not only enhance the hand-drawn characters but also make them appear real as compared to the above mentioned anomations.

2D Animation:

It is used through Powerpoint and Flash animations. Though its features are similar to cel animation, 2D animation has become popular due to simple application of scanned drawings into the computer like in a cartoon film.

3D Animation:

It is used in filmmaking where we require unusual objects or characters that are not easy to display. Use of 3D animation can create a crowd of people in a disaster like earthquake, flood or war. There are different shapes, support of mathematical codes, display of actions and colors which are mind-blowing as if copied from an actual picture.

The above mentioned 3 types of animations have brought a new era of amazing technology in the field of Internet (website design and graphics), film industry and media. Many of us can try out making their own animation! In addition, animation is one of the popular Internet marketing strategies that make visitors stay on your site for a longer time.

2D Animation Techniques

- Classic, hand drawn animation - Disney's Lion King.
- Cut outs - Monty Python (see also in stop motion and in 3D. This technique crosses all barriers!)
- Rotoscope - Waking Life
- Flip book - Keith Haring has made some famous ones.

Computer Assisted Animation (2D)

This term refers to all types of animation that use a computer somewhere in the process.

One could argue that this means ALL ANIMATION today.

Mostly we use it to describe the tools that have come to replace pencil, paper and film, for example:

- Flash animations - Many TV series are now done in Flash, check out this example.
- Coloring and layering hand drawn animation using a computer
- Drawing directly into an animation software with a Pen Tablet **3D Animation Techniques**
- 3D animation- Pixar's Up, Toy Story
- Stereoscopic 3D - Coraline, Avatar
- CGI cut out - South Park
- Motion Capture (an aid tool for 3D animators)- Final Fantasy, Avatar, Gollum in Lord of the Rings.
- Morphing (Remember the changing faces in Michel Jackson's Clip Black or White? that's Morphing.)

Stop Motion Techniques

- Clay or Plasticine ("claymation")- Nick Park's Wallace and Gromit
- Real Clay animation (and lot's of other stuff you won't suspect)- Jan Svankmajer's Dimensions of Dialogue
- Puppet animation- Tim Burton and Henry Selick's The Nightmare Before Christmas

- Pixilation - Peter Gabriel's music video "Sledgehammer"
- Cut outs - Daniel Greave's Flat World is a stunning combination of classic hand drawings with cut outs.

Types of Animation that are done on a light table, shot frame by frame under a camera:

- **Sand animation -** This is sometimes done as a performance art, shown live for an audience, and sometimes it's stop framed into proper film.
- **Oil colors -** Caroline Leaf's The Street, and the frankly-unbelievable Old Man and the Sea by Alexander Petrov.
- **Plasticine** - Ishu Patel's Afterlife

Types of animation named after a software:

Some types of animation are named after the software used to create them. Flash animation has come to mean a certain kind of graphic look and feel, which has also spawned the pleading request "Can you make it NOT look like Flash, PLEASE!"

There are also:

- **GIF animations -** GIF is a type of file format, used for small, light weight animations with no more than a few frames.
- **After Effects animation -** usually means either cut outs done in After Effects, or animation done with the program's Puppet Tool (which is amazing, BTW).
- **Blender, Mudbox, and Maya -** all names of 3D animation softwares.
- **Pivot stick figure -** A freeware for making stick figure animations. So simple, and so popular!
- **Morphing is a type of animation that uses a software to fill in the gap between two images -** MJ's "Black or White" music video.

---

2. Authoring tool, presentations, applications, interaction:

- Multimedia authoring tools provide the framework for organizing and editing the elements of a multimedia project.
- Authoring software provides an integrated environment for combining the content and functions of a project.

- It enables the developer to create, edit, and import data.
- In multimedia authoring systems, multimedia elements and events are often regarded as objects.
- Objects exist in a hierarchical order of parent and child relationships.
- Each object is assigned properties and modifiers.
- On receiving messages, objects perform tasks depending on the properties and modifiers.

Features of Authoring Tools:

- Editing and organizing features.
- Programming features.
- Interactivity features.
- Performance tuning and playback features.
- Delivery, cross-platform, and Internet playability features.

Editing and Organizing Features:

- Authoring systems include editing tools to create, edit, and convert multimedia elements such as animation and video clips.
- The organization, design, and production process for multimedia involves storyboarding and flowcharting.
- Visual flowcharting or overview facility illustrates project structure at a macro level.

Programming Features:

- Visual programming with icons or objects is the simplest and easiest authoring process.
- Visual authoring tools such as Authorware and IconAuthor are suitable for slide shows and presentations.
- Authoring tools offer _very high level language' (VHLL) or interpreted scripting environment.

Interactivity Features:

- Interactivity gives the end user control over the content and flow of information in a project.
- Simple branching is the ability to go to another section of the multimedia production.
- Conditional branching is an activity based on the results of IF-THEN decisions or events.
- Structured language supports complex programming logic, subroutines, event tracking, and message passing among objects and elements.

Performance Tuning and Playback Features:

- Achieving synchronization is difficult, considering that performance of the different computers used for multimedia development and delivery varies.
- Authoring system should facilitate precise timing of events.
- It should enable developers to build a part of a project and then test it immediately.

Delivery, Cross-Platform, and Internet Playability Features:

- Delivering the project may require building a run-time version of the project, using the multimedia authoring software.
- Run-time version or standalone allows a project to play back without the complete authoring software and all its tools and editors.
- It is important to use tools that facilitate easy transfer across platforms.
- Authoring systems provide a means for converting their output to be delivered within the context of HTML or DHTML.

Types of Authoring Tools:

- Card- and page-based tools.
- Icon-based, event-driven tools.
- Time-based tools.

Card- and Page-Based Tools:

- Card- and page-based authoring systems provide a simple and easily understood metaphor for organizing multimedia elements.
- It contains media objects such as buttons, text fields, and graphic objects.
- It provides a facility for linking objects to pages or cards.

Icon-Based, Event-Driven Tools:

- Icon-based, event-driven tools provide a visual programming approach to organize and present multimedia.
- Multimedia elements and interaction cues are organized as objects in a flowchart.
- Flowchart can be built by dragging appropriate icons from a library, and then adding the content.

Time-Based Tools:

- Time-based tools are best suited for messages with a beginning and an end.
- Some time-based tools facilitate navigation and interactive control.
- Macromedia‘s Director and Flash are time-based development environments.

Macromedia Director:

- A multimedia database, _Cast', contains still images, sound files, text, shapes, scripts, movies, and other Director files.
- Score is a sequencer for displaying, animating, and playing Cast members.
- Lingo is an object-oriented scripting language that enables interactivity and programmed control.

Macromedia Flash:

- Flash is used for delivering rich multimedia content to the Web.
- It allows the creation of simple static HTML pages with the Flash Player plug-in.

Cross-Platform Authoring Notes:

- Macintosh and Windows computers use different schemes to manage text and colors.
- While using text fields, ensure that the text displays correctly on both platforms.
- Outline and shadow styles on text should be avoided on Macintosh since they are not currently supported in Windows.

14.3 2D and 3D animations- projects simple animations:

The most obvious difference between the two genres of animation is of course the three dimensional characteristics or the appearance of depth. While 2D animation is a flat animation and all the actions happen in the x-y axes, 3D animation includes an extra dimension and that is the z axis.

The working method for creating 2d cartoon characters and 3d animated figures are entirely different. While in 2D animation the process of cartoon character creating involves sketching the character from different sides with the help of onion skin tools. Creating a 3d model requires digital modeling and is more similar to sculpting a character than drawing one. An animator working in 3d dimensional environment constantly has to be aware of how his/her changes to the model side view affect the front view or any other view for that matter. Creating a perfect looking model is a difficult task since all the different views have to be taken into the consideration. Creating a 3d model is often based on a pre-made two dimensional sketches of the character from different views. After the model is created a material has to be assigned to it and the model has to be textured properly.

Although the process of character creating for 3D animation is usually taking more time than 2D character creation, the process of creating the animation itself can be considered easier by many animators. In 2d animation the animation is created by drawing almost every frame of the animated movie. In 3D , the animation is created by changing the poses and the placement of already created 3d models. The created scene can be viewed from different angles and by that it is easier and faster to create an illusion of change in the environment.

Various techniques that are acclimated in creating 2D abstracts are morphing, twining, onion skinning, Anime, and amid rotoscoping. Admitting 3D action involves agenda clay of characters. Various accomplish that are complex in 3D action are appearance sketching, appearance modeling,, Anime, arena building, texturing, abating and camera setup, rendering, abating and camera setup, rendering,alteration and bond etc. Other techniques, Anime, that can be activated are the use of algebraic functions, apish fur or hair and the use of motion capture. In this way we can abstract greater use of multimedia through 2D and 3D animation.

4. Unit Summary:

This unit provides brief details about the authoring tools, 2d and 3d animations.

5. Keywords:

Types of Animation, Authoring tool, Applications, Interaction, 2D Animation, 3D Animation.

6. Exercises:

1) List the various Animation Types.

2) Write note on Authoring tool, presentations, applications, interaction.

3) Explain 2D and 3D animations.

14.7 References:

1.Prabat K Andleigh and Kiran Thakrar, ―Multimedia Systems and Design‖, PHI, 2003. (UNIT 3 to 5)

2.Donald Hearn and M.Pauline Baker, ―Computer Graphics C Version‖, Pearson Education, 2003.

3. Designing Interfaces by Jenifer Tidwell

4. Digital Multimedia by Nigel Chapman
5.Information Architecture for the World Wide Web: Designing Large-Scale Web Sites by Peter Morville, Louis Rosenfeld
6. Information Visualization, Second Edition: Perception for Design by Colin Ware
7. Letting Go of the Words: Writing Web Content that Works by Janice Redish
8. Multimedia Applications (X.media.publishing) by Ralf Steinmetz
9. Now You See It: Simple Visualization Techniques for Quantitative Analysis by Stephen Few
10. Visual Thinking: for Design by Colin Ware
11.Writing for New Media, Third Edition: Content Development for Bloggers and Professionals by Timothy Paul Garrand
12.Real-Time Video Compression: Techniques and Algorithms By Raymond Westwater Publisher: Springer 1997

Unit 15: Video Content, Applications, Anotomy of Human Being

Structure:

1. Learning Objectives
2. Video content
3. Complex presentations, applications
4. Anatomy of human being
5. 3D animation
6. Media-rich 2D
7. 3D applications - pictures, sound, video, and special effects
8. Unit Summary
9. Keywords
10. Exercises
11. References

15.0 Learning Objectives:

After studying this unit, you will be able to

- Understand about 3D animation and applications
- Discuss about media rich 2D

15.1 Video content:

~~Video, film, and animation are all moving images that are recorded onto videotape, a computer~~ disk, or other media formats. While film has always been an expensive, professional industry, video has been more accessible and less expensive for users. Film and video are separate media, yet with the popularity of digital video (DV) cameras and nonlinear video editing software, many of the differences in quality and usage are becoming obsolete. Video can be recorded with a video camera or captured by a screen recording software tool like Camtasia Studio to display moving images to the user. Animations were originally sequenced drawings that, when viewed in quick succession, gave the appearance of movement. Computer-based animation allows you to create movement of shapes, text, and other illustrations within web pages, videos, and other media. An animation is designed and then drawn from the data, or ―rendered,‖ in a software program to illustrate full-motion moving images or three-dimensional concepts not possible in

two-dimensional video. Animation can be effective when you need to demonstrate a process or procedure. Moving images are useful for technical communicators in many situations, and video and animation are often integrated into the mainstream workflow in many technical workplaces.

Uses of Video and Animation:

Technical fields and industries use moving images for computer-based training (CBT) to illustrate mechanical processes, document safety procedures, provide technical information, and offer software training to users.

Animation is sometimes used in conjunction with video in these applications. CBT programs may use ani-mations to illustrate a concept or process, add an interactive illustration, or provide additional technical information to users between sections of live video.

Video clips are short sections that can be included in a web page, presentations, and other digital documents. Video clips are also quite common across the Internet for news and entertainment, appearing on sites such as cnn.com, sports pages, and YouTube. Technical communicators might be asked to create storyboards for a training video, to design and test the usability of educational software, or to help design videos and animations for software training purposes. As the hardware and software become less expensive and easier to use, full motion and interactive documents have become easier to produce in house and distribute to users outside the organi-zation. The specialized types of video often created by technical communicators include videos used for education and training.

15.2 Complex presentations, applications:

Scope of Animation in Different Professional Fields animation has become a crucial part in each field of Indian business industry.

Animation is not limited to commercial TV conetnt such as cartoon films and 3D film making. It has spreaded its existance in all professional fields such as Entertainment, Marketing, Advertising, Education, Aviation, Designing, Medicine, Media, Information Technology, Internet, Banking, Insurance, Finance, and so on.

In Entertainment field Animation can be used in form of Film Making, Cartoon characters, Game Designing, TV commercial special effects. In Marketing and Advertisments,

animation is crucial part in form of various TV commercial Advertises, online marketing, promotional activities presentations, banner ads. Film Making, TV commercial content and Game Designing are in boom.

Education is becoming very interesting since ever animation has been a part of education called online education or e learning. Children can play with boring and tedious calculations using lovely and lively online education presentations having attractive animated cartoon graphics and fresh bright colors.

Various Presentations such as corporate CD presentation, medical education, documentory films or presentations use animation and graphics techniques. We can have learning sessions or medical education by documentary films created by using 3D Animation and Multimedia .

In Designing field, animators can pour their innovative ideas in Fashion Designing, Jewellery Designing and Interior Designing. In Interior Designing, builders and company management prefer 3D perspectives, 3D walk throughs to create simulation module to plan any project.

Internet is a very impressive and competitive marketplace. Web Designing, banners designing, corporate logo designing, banner advertising where animators and graphics designers are highly demanded.

Animation is used for e-commerce in advertising the products and services of the organization. A marketer can incorporate the multimedia content in the web pages of a company site. Aviation sector can encourage e-ticketing among its customers. Commercial sites offer ECommerce and online shopping of their products by embedding Multimedia and Animation.

In Interior Designing, builders and company management prefer 3D perspectives, 3D walk throughs to create simulation module to plan any project. In jewellery designing you can create attractive jewellery design graphics for simulation which can be finalised for final creation.

In journalism, to maintain public relations, graphs, graphics needed to declare percentage into reports.

Also audio and video editing and multimedia can be incorporated into presentations in various sectors.

In film industry, animations, special effects, graphics, audio and video editing and multimedia are seamlessly embedded.

As animation has entered into different types of professional fields, many new institutes have also emerged where young aspirants are trained to explore prospects for themselves in this field.

The use of graphics, Multimedia and 3D Animation has enhanced the presentation style of various professional fields.

15.3Anatomy of human being:

There are software that provides you with many tools that make it easy to create realistic movement for a character.

For the sake of simplicity, this topic discusses how the MotionBuilder character engine works with biped or human skeletons. There are special conditions and exceptions when animating quadrupeds that are beyond the scope of this section.

Before you can successfully animate a character, it is important to understand how skeletons move and how MotionBuilder can help you recreate believable motion.

Studying the human skeleton

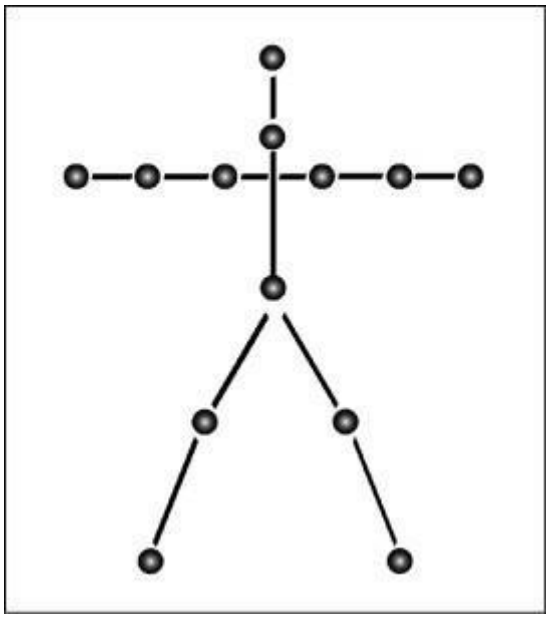
In order to create believable human movement, you must first look at the human skeleton and how it operates.

A human skeleton is basically composed of a series of joints. The joints are connected by bones which are moved by muscles and tendons.

In order to create realistic movement in animation, 3D skeletons mimic the movement of human skeletons using a series of joints, rather than actually replicating the complex system of muscles, tendons, and bones.

A simple 3D skeleton is composed of the main joints, such as the shoulder, elbow, wrist, knee, and so on. Since muscles and tendons move the joints that move a real skeleton, we can achieve an accurate recreation on a model by manipulating the joints directly, without replicating muscles and tendons.

Each point is placed in 3D space and connected together to resemble the joints of a human skeleton. For example, the following illustration shows a very simplistic skeleton created with dots representing the joints, and lines representing the bones. What is drawn between the joints is not important at this stage.



The series of dots represent joints that are connected with lines (representing bones). The next section discusses the relationships and rules needed to make this series of points act like a human skeleton.

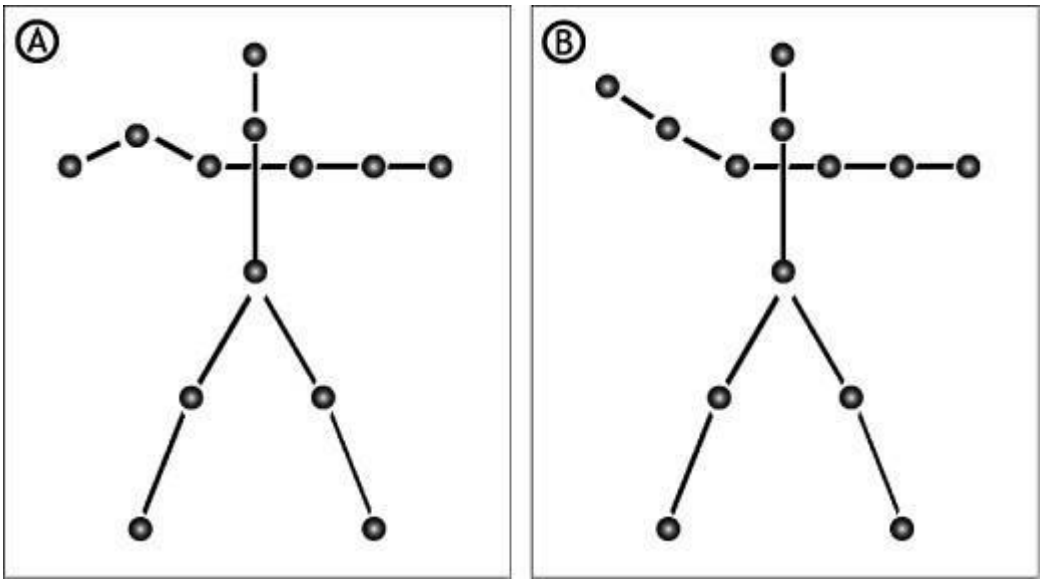
Basic rules of creating human movement

When you characterize a character, the character engine automatically defines a group of settings that govern human movement, so you do not need to create your own. Still, it is important to have a good idea of what is involved in creating realistic human movement.

Making a series of points act like a human skeleton involves setting up rules to determine the ways in which the points can interact with each other. For proper interaction, it is necessary to create relationships to govern how each point influences the other. Otherwise, when the series of joints moves, it does not behave like a human skeleton.

Refer again to which uses dots to represent joints. With no relationships established, each joint moves independently.

For example, if you move the dot representing an elbow, only the elbow moves and the wrist and hand do not move. On a human skeleton, if you move your elbow, the wrist and hand move as well.



A. If no relationships are established to define the movement of the right arm on the simple skeleton, it moves in an unnatural fashion. B. A rule is created to make the wrist and hand move when the elbow moves, resulting in natural movement.

To create natural movement, you need a rule that says ―when the right elbow moves, the right wrist and hand must also move‖.

You also need a rule to govern the rotation of joints. Through studying human movement, it is obvious that when, for example, the right elbow moves, the wrist and hand also move, and the right shoulder rotates. So you create a rule that says ―when the right elbow moves, the right shoulder rotates.‖

15.4 [1]D animation:

armature that can be used to control the mesh by weighting the vertices. This process is called rigging and can be used in conjunction with keyframes to create movement.

Other techniques can be applied, such as mathematical functions (e.g., gravity, particle simulations), simulated fur or hair, and effects such as fire and water simulations. These techniques fall under the category of 3D dynamics.

[1] D animation is digitally modeled and manipulated by an animator. The animator usually starts by creating a 3D polygon mesh to manipulate. A mesh typically includes many vertices that are connected by edges and faces, to give the visual appearance of form to a 3D object or 3D environment. Sometimes, the mesh is given an internal digital skeletal structure called an

3DTerms

- Cel-shaded animation is used to mimic traditional animation using CG software. Shading looks stark, with less blending of colors. Examples include, Skyland (2007, France), Appleseed Ex Machina (2007, Japan), The Legend of Zelda: Wind Waker (2002, Japan)
- Machinima – Films created by screen capturing in video games and virtual worlds.
- Motion capture is used when live-action actors wear special suits that allow computers to copy their movements into CG characters. Examples include Polar Express (2004, US), Beowulf (2007, US),A Christmas Carol (2009, US), The Adventures of Tintin (2011, US)
- Photo-realistic animation is used primarily for animation that attempts to resemble real life, using advanced rendering that mimics in detail skin, plants, water, fire, clouds, etc. Examples include Up (2009, US), Kung-Fu Panda (2008, US), Ice Age (2002, US).

15.5 Media-rich 2D:

The term "computer animation" itself broadly covers a wide variety of genres and applications, though the simplest way to break it down is into the categories of 2D and 3D animation. "2D", short for "two-dimensional", is sometimes also called "vector animation", and is typically done in programs like Macromedia Flash and Macromedia Director. The most familiar form of 2D animation can be found just by turning on your TV on a Saturday morning: traditional cartoons, which are progressing more and more into the digital realm. You probably see simpler animations every day just while surfing the web, in the form of advertisements, Ecards, and cartoon shorts. Vector animation is also useful in designing interactive interfaces for the web.

2D animation, true to its name, is rendered in a two-dimensional space. 3D animation, however, is rendered in a virtual three-dimensional space, using polygons captured by various virtual "cameras" to "film" the animation. 3D animation has a variety of applications, from video games to animated films; most commonly, 3D animation is used to render many of the special effects seen in live-action films, removing the need for scale model sets or staged stunts.

While both types of computer animation can be accomplished by either frame-by-frame animation or by mathematical interpolation between key frames, the initial steps prior to animating are drastically different; the two separate processes also require different software

packages. With that in mind, the tutorials provided here have been grouped into the categories of 2D and 3D animation, before being subdivided by skill level to walk, step-by-step, through the basics of creating your own animations. The 2D animation tutorials cover animation in Flash and Director/Shockwave, while the 3D animation tutorials work in 3D Studio Max.

15.6 3D applications - pictures, sound, video, and special effects:

As the development of technology is progressing day after day, computer is introduced into the animation industry, the traditional animation technology (hand-draw animation) is a very large scale project and the amount of time, labour and equipment involved is very costly. While

for the computer animation, the process is much less labor intensive, and generally much cheaper than traditional animation, there is lesser chance of getting error as from stage to stage as computer can have a backup file. Also by using computer it can shorten the production life cycle.

From the business point of view, animation can also help business to promote and advertise their product, as flash animation design can attract customer more than a whole bunch of words or text given out in brochure or plain text website. It can help customers to have a better

memory of the products.

From the education point of view, animation cartoons was one of the method to educate children outside the classroom through the media

3D models:

3D modelling on computers is used by Product Designers, these people create virtually anything within programmes like TrueSpace or Solidworks CAD. Instead of a physical product, the designers create a graphic 3D image on the software package they are using; this method is ultimately cheaper than physical creation as it allows any shape or form to be created without having to pay for materials in building a physical example.

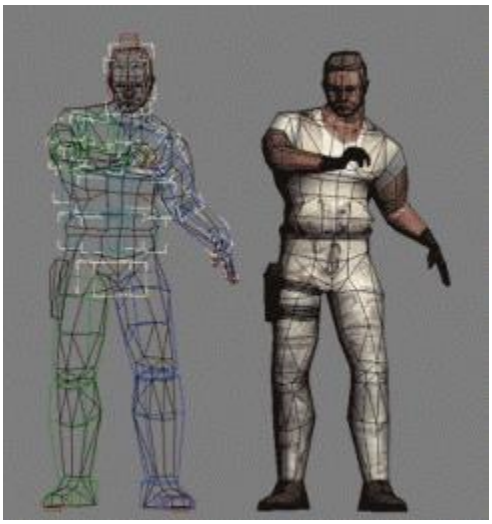
3D models are split in two seperate categorys, Solid and Shell/boundarys. Solid models are the most realistic and define the volume/weight of an object they are supposed to represent. They‘re more difficult to develop within a software package than its counterpart, shell/boundary,

therefore they are normally used in non visualised media. An example of solid modelling would be first aid kits and ornaments.

Here is an example of Solid 3D modelling. It is a First Aid Kit taken from a design forum, the user created this object from scratch and used other designs as a reference point.

Product design: Product design involves the designers of a product gathering and suggesting designs that would suit the intended audience of the product, for example they may have a meeting and brainstorm to develop new or flesh out existing ideas on the subject. Eventually the designers will start to create the product, several designers may be working on the same product or there may be several designers creating different products to better decide on which is the first choice to be sent to market.

There are several ways a designer will create a object, the most popular being autodesk and adobe softwares, these include autodesk‘s own CAD package and many adobe packages like Bridges and Photoshop. They may also use other programmes like 3ds max, which is another software package from autodesk. This editing software allows the user to create and develop 3d objects and make them anything they can think; this is very useful to designers.



The developer would also talk to the designers about how the product is specific to a certain age range; for example a 3+ videogame would not work with massive amounts of horrific monsters stained in blood, the designers would have to develop child friendly characters.

Animation, TV, Film and Videogames: These are the main users of 3d technology, animation is the main user of 3d as modern animations like cartoons use the tools nesscessary to develop a piece of 3d entertainment; it is the same with videogames and some TV shows that are animated and not shot in real 3d like Eastenders. Films are different, the producer developing a catoon film like Toy Story will use 3d development software heavily whereas someone who is developing a film that uses CGI characters as well as real people will use a mix of real filming and software, possibly using green screen as well. A prime example of a CGI film is Transformers, some sections are real fil whereas other parts use CGI to mix the giant robots with the real actors.

Web and Architecture: 3d design software can be used to develop ideas for new buildings or additional extensions to a building. The same can be said for web based design, a website can be built from scratch just like a designer would build a building design in a programme.

Building design- In architectural design the producer must be accurate with his development, and also has to take in consideration of the laws of physics etc etc. For example the user could create a building with a sloped roof, the designer would have to show how the roof conforms to real life laws by showing within the programme that it won‘t fall or break off.

Web based design- Within web based design the common laws that govern our world do not apply apart from laws set by the Governmant obviously; the site for example could have crazy designs of future cities that have been created on certain programmes and implemented as the background for the site. Decent web design is required to sell a site effectively. For example Google is a plain professional looking site, the colours work and the site is used daily by millions of people who are always happy with the way it looks. The same can be said with Facebook, although people complain about the changes that are made they eventually endure and get used to the new design. However there are bad web designers out there, terrible site design does not sell the viewers, they will probably leave the site after viewing the front page. Below is a prime example of awful website design, viewers would want to leave instantly.

7. Unit Summary:

This unit provides brief details about the 3D animation and its applications, media rich 2D

8. Keywords:

Video Content, Applications, Anatomy of human being, Media-rich 2D, 3D applications.

9. Exercises:

1) Explain Video content.

2) Explain 3D animation.

3) Expalin media-rich 2D.

4) Explain 3D animation applications in pictures, sound, video, and special effects.

15.10 References:

1.Prabat K Andleigh and Kiran Thakrar, ―Multimedia Systems and Design‖, PHI, 2003. (UNIT 3 to 5)

2.Donald Hearn and M.Pauline Baker, ―Computer Graphics C Version‖, Pearson Education, 2003.

3. Designing Interfaces by Jenifer Tidwell

4. Digital Multimedia by Nigel Chapman

5.Information Architecture for the World Wide Web: Designing Large-Scale Web Sites by Peter Morville, Louis Rosenfeld

6. Information Visualization, Second Edition: Perception for Design by Colin Ware

7. Letting Go of the Words: Writing Web Content that Works by Janice Redish
8. Multimedia Applications (X.media.publishing) by Ralf Steinmetz
9. Now You See It: Simple Visualization Techniques for Quantitative Analysis by Stephen Few
10. Visual Thinking: for Design by Colin Ware
11. Writing for New Media, Third Edition: Content Development for Bloggers and Professionals by Timothy Paul Garrand
12. Real-Time Video Compression: Techniques and Algorithms By Raymond Westwater Publisher: Springer 1997

Unit 16: Animation Files and Internet

Structure:

1. Learning Objectives
2. Animation files and internet
3. Movie types and its uses in supportability for the web.
4. Unit Summary
5. Keywords
6. Exercises
7. References

16.0 Learning Objectives:

After studying this unit, you will be able to

- Understand about animation files and internet
- Discuss about movie types and its uses in supportability for the web

16.1 Animation files and internet:

The animation on the Internet is constantly changing. It wasn't too long ago when the simple Gif89a, known as the animated GIF, was the only game in town. However, with first the use of plug-ins for browsers, and then the invention of cascading style sheets and DHTML, the world of animation on the Internet is exploding.This look at the current status of animation tools for creating animation on the Internet is based on different levels of complexity for the audience browsers. Animated GIFs are the most universally viewed by the Internet audience. Most of the client browsers can view this type of animation without any additional software.

Animations that require plug-ins, or additional software add-ons for browsers, allow for greater diversity and more interaction, but require the audience to download additional software

to be able to view them. This was more troublesome in the recent past, but luckily this process is getting more automated in the more recent browser versions.

Most recently, the latest browsers allow for animations without plug-ins, but these browser versions haven't been fully populated among the Internet audience. Known as Dynamic HTML (DHTML), this style of animations is viewable with the 4.0 or greater browsers. Unfortunately, browser companies have implemented slightly different versions of this technology, so sometimes one has to choose which browser type to support or create two different versions of the animations.

What Type Of Animation Do You Want To Create?

In determining which tools to use you first have to decide what type of animation you want to create. Below I've listed the different types of Internet animations, and what they are generally used for, plus, I have included the benefits and drawbacks for each tool:

Gif89A

Benefits: Both repeating and single animation; Plays on almost all browsers with no plug-ins needed.

Drawbacks: No interaction; Entire file has to be downloaded before starting; Complex animations create very large file size; No audio Plug-ins.

Benefits: Greater interaction; Audio can be used; Ability to start animation before entire file is downloaded; Can transfer existing animations into some formats. Drawbacks: Most browsers need additional software to show animations; Requires some level of audience expertise to load software DHTML

Benefits: No plug-ins needed; Allows for animations and scripting of animations Drawbacks: Need the most recent browsers to use; Competing standards require different versions of content.

The FirstAnimations

The GIF89a technique allows for multiple images to reside in one file, and then be played back in sequence after being downloaded to a browser. Once a series of images is created using such image editing tools as Adobe Photoshop, Adobe Illustrator, Macromedia Freehand, Corel

Draw, etc. or video authoring tools like Adobe Premiere, they are imported into a GIF89a tool for the animation composition.

Old Favorites: GIF89a Tools

Windows: GIF Construction Set (Alchemy Mindworks).

This shareware tool allows for both transparent and interlaced GIF creation, as well as video-to-GIF animation conversion. This program also has an Animation Wizard to simplify the process for the newcomer. For Windows 95, there are more deluxe features such as greater pallet choices, transitions and special effects.

Macintosh: GifBuilder

This freeware utility from Yves Piquet has a scriptable utility for creating Animated GIFs from PICT, GIF, TIFF, and Photoshop images, as well as Quicktime and FilmStrip file conversion.

Cross-Platform Tools WebPainter

From Totally Hip comes an inexpensive tool to create animations that combine standard design elements (spray can, various brushes, smudging, etc.) with animation features such as onion skinning (layering). You can download a free sample of this product from the Totally Hip web site.

New Products on the Block

Both Adobe and Macromedia are introducing new tools to aid in the creation of web graphics and animated GIFs. Both Adobe's Image Ready and Macromedia Fireworks promise to combine image creation, animation and image compression tools in one. While both of these tools cost more than the freeware or shareware tools shown above, they promise to become one stop shopping for web image creation. These tools will be available for both the Windows and Macintosh platforms.

Plug-ins: The Next Evolution

These groups of tools allow for much more interactive capabilities or the ability to take your existing animations directly to the web. They require a plug-in or additional piece

of software to be included in the viewer's browser. Sometimes these plug-ins ship with the browser, such as Apple's Quicktime, or the viewer must download it onto their computer. Luckily, in the more recent versions of both Netscape and Microsoft's browser, this has become much easier to do.

For your existing animations or animations created on current animation software, exporting the animation to Quicktime or AVI will allow you to place this into a web page. There are many places on the web that will show you how to format your web page to insert this media type into the HTML. For the viewer to play this file, they will have to download the entire file before playback, so keep an eye on file size. One way to address this in a large animation piece is to break the animation up into smaller pieces, and allow the user to download a segment at a time.

To reduce the delay to the viewer of having to download the entire piece, there are streaming video technologies that will allow you to encode the animation in a format that requires a special plug-in to view. The two most prevalent technologies of this type is Real Network's Real Media and Microsoft‘s Netshow.

Real Networks Real Media allows for an animation piece to be encoded in a specific file type, and then streamed or downloaded and played in real time across the net. Adobe's Premiere and other digital editors will export into this format. For multiple viewers to see this animation at the same time, Real Media sells their server software which is available in various flavors of Unix and NT. Ask your Internet service provider or hosting company about this service.

Netshow, from Microsoft, is a similar product, but the server is free for NT servers. The early versions of this technology were not at the same quality level as Real Media, but Microsoft is introducing Netshow 3.0, which will be a great improvement.

While Internet video, either downloaded or streaming, can be used for your animation, it is highly compressed and appears in a small (160x240 pixels) window in the browser or helper window. For full screen animations, or greater interaction, you need to create your animation in the tools designed for Internet Animation.

Let's review the current tools and their pros and cons for original Internet plug-in animations.

Old Favorites: Plug-in Animation Tools Shockwave (Macromedia PC and Mac)

This technology twist from Macromedia allows for Director files to be compressed and played back in your viewers' browsers. Bundled with both Microsoft and Netscape browsers, this allows for the interaction and animation capabilities of Director to be available to your Internet audience. It will allow you to convert existing Director files into a Shockwave version, and in recent versions of Director, allows you to determine if you want to have the entire file downloaded before playing or streaming to the browser. This is the most prevalent interactive animation tool on the Internet, allowing for bitmap graphics, audio files, Quicktime embedding in files and many other advanced features, but file sizes are large and it requires more viewer memory for acceptable playback.

Flash (Macromedia PC and Mac)

This is a simple vector-based animation program that keeps getting better with age. By using vector based graphics, this format allows for smaller downloads and scalable graphic sizes (the same graphic will automatically scale for the browser size window). The tool is less complex than Director, and doesn't include the variety of tools found in higher end traditional animation tools. However Flash allows for simple animations that create fast and easy roll-over buttons and animations with simple sound use.

Real Flash (Real Networks, Macromedia)

Recognizing the small and fast file transfers of this technology, Real Networks and Macromedia have worked together to create Real Flash, a streaming animation program that combines the streaming audio capabilities of the Real Network's Real Media server with the vector graphics capabilities of Flash. This allows for streaming vector-based animations, and is pushing the Internet closer to real-time animation. Many content companies are creating original animations for the web using this technology. It's fast, easy to author in, and syncs higher quality sound with streaming animations. Look to Macromedia or Real Network's web sites for examples of this style of Internet animations.

Coming Down The Pike Based upon the success of the products listed above, many newcomers are looking at plug-in based animation tools for the Internet. One new one is Liquid Motion (Microsoft), which allows for both ActiveX utilization (on IE 4.0 and greater browsers) and a Java implementation on other browsers. This tool set appears to be competing with

Macromedia's Flash, and we should wait to see what the adaptability of this new tool will be. The beta is available from the Microsoft web site.

7th Level is presenting Agent 7 technology in both Netscape and Microsoft, but only for Windows clients. This is an interesting streaming 2-D animation program, but the tool set for creating this style of animation is not yet announced. Watch the 7th Level site for ore details.

For an alternative way of delivering animations to the desktop, Togglethis allows for delivery of animations via the Internet's first killer application, e-mail. After downloading a player from their site, an animation can be delivered via e-mail. This allows for a much smaller file size (10-25 kilobytes), but the animations play back on your desktop, not within the browser. Warner Bros. Online and Togglethis have created Bozlo the Beaver, a new Warner Bros. cartoon character. Look for this creature at either one of their sites.

DHTML:The new kid is Dynamic HTML, which allows for animation, style sheets and other ways to better format HTML. There is a growing set of Internet tools that will allow you to create DHTML. One benefit is that it works across browser types and needs no plug-ins for playback. One problem is that Microsoft and Netscape have different implementations of design specifications, so the HTML plays back differently on each browser.

DHTML is still in its early stages. At some point the standards board will finalize the specifications for this format, and both Microsoft and Netscape have agreed to conform to this specification. It may not be able to do the complexity of animations and interactions that the plug-in tools will allow, but it is the easiest format for complex animations for the user since animated GIFs.

Windows: Dynamite (Astound Windows):

This standalone DHTML tool allows for the creation of DHTML for Windows based platforms. This tool allows for simple DHTML creation, but doesn't allow the customizing of some of the other tools. However, for most people however trying to learn this style of animation creation, this tool offers good training and an on-line tutorial.

Macintosh:

CyberStudio 3.0 (GoLive):

This update to the top ranked Macintosh web site creation tool has added the capability to author DHTML pages. It allows for many pre-authored effects, along with the ability to author and store your own animation techniques. It is only available for the Macintosh, and is not yet shipping. A beta is available from the GoLive web site. Cyberstudio appeals to authors from a page layout, less programming level of comfort.

Cross-Platform Tools:Dreamweaver (Macromedia; Both platforms)

The first industrial strength web site creation tool, Dreamweaver excels at DHTML. It allows for both pre-defined and author created animations, and can author out to either browser type. It also keeps cleaner code than other site creation tools, but can be a little imposing to those authors unfamiliar to HTML authoring.

Supporting animation format suitable for internet

- **.gif images -** Fully supported, but limited use with frames. .png files are supposed to have support for animation, but you don't see it in the wild very often, if at all.
- **Flash, Silverlight, and other plugins -** Flash has the most ubiquity, but all plugins need to be installed in the browsers and can't necessarily be counted on to be installed on a user's computer. May not work for mobile platforms.
- **CSS3 -** Limited transforms or animations, limited browser support.
- **HTML5 video -** The future, but limited browser support.
- **JavaScript -** Needs to be enabled, but should have near universal browser-support. Limited types of animation. See Raphael.js.
- **Canvas tag -** Modern browser support, but supports rich animations.

---

2. Movie types and its uses in supportability for the web:

---

2 broad categories of video formats. Some formats are meant for your finalized video and are called **sharing formats**. Other formats are used more in the beginning stages of a video

project. They are good as **raw video master** clips. These formats will have higher resolution and larger file sizes than the sharing formats.

Considerations for Internet Video File Formats

If you are making and uploading video to the web, you need to be aware of video formats in any one of these three stages of the process:

1. **What kind of video comes out of your camera?**
   This is often referred to as the source video or raw video format.
2. **What kind of video does your video editing program accept?**
   You want to make sure that whatever comes out of your camera is supported by the video editing program you want to use.
3. **What kind of sharing format is best for the** online video platform **you've selected ?**
   When you are evaluating whether a format is the "right" one or not, the three main evaluating criteria are:

- Size of the file
- Resolution and overall appearance of the file
- Compatibility of the file

Depending on how the video clip will function, any one of a dozen or so popular formats will be among the "best" one to choose.

For the Internet, you want smaller file sizes. Even though the web's capacity for video delivery keeps growing, and most people have broadband, it is still best to keep file sizes as small as possible. A large video file is more likely to encounter buffering issues and drive the viewer crazy. The smaller the file size while still maintaining high quality, the easier it will stream, which may very well make the difference between whether someone bothers to watch or not.

There can be huge differences in size from format to format depending on the compression used. I've experimented around with this and I can take the same video, convert it to QuickTime and get an 80 megabyte file or convert it to an mpeg 4 with h.264 compression and get a 10 megabyte file

Common Video File Formats:

AVCHD (Advanced Video Codec High Definition):

AVCHD (.mts) is a high end, high-definition (HD) format originally developed by Sony and Panasonic for HD home theater systems. It's not a sharing format for the web because it is so huge, but it has become very common as a lot of newer HD camcorders record in this format. Video in this format would be for the beginning of your video project and serves as a master clip you would use to edit with.

AVCHD is in its infancy as a format and since it's still fairly new and compatibility with certain video editing programs may be an issue. Some video editing software applications have begun to support this format but many of can not handle it well yet. Additionally, playback of AVCHD files requires speedy CPUs and a sufficient amount of RAM. That makes this format a little difficult to work with but it maintains high quality. As time goes by, it will no doubt become easier to use.

.AVI (Audio Video Interlaced):

This is a long-time standard developed by Microsoft and has been around as long as digital video has. .AVI files (particularly when uncompressed) tend to be HUGE, way too big for the internet. AVI is more for the beginning of a video project, not the end. In that sense, it is not really a sharing format. They'll slide into just about any video editing program and the quality is still high enough to be a master clip.

AVI is windows-based and is virtually universal. Problem is, not all AVIs are created equally and you can still run into compatibility issues. AVI is what's known as a container format, which means it contains multiple streams of different type data, including a control track and separate video and audio streams. Now, what streams inside the container is not necessarily the same from one avi video to the next as the codecs used for compression can vary.

.FLV (Flash Video Format):

Next, we'll talk about the .flv format. Flash video is the single most common sharing format on the web today. You'll see the .FLV file extension on videos encoded by Adobe Flash software to play within the Adobe Flash Player. Virtually everyone (99%) has the adobe player installed in their browser and so this has fast become the most common online video viewing platform.

Almost all the video sharing sites stream video in flash. You can upload formats other than flash, and those sites will convert it into flash for streaming to the end user. Notable users of

the Flash Video format include YouTube, Yahoo! Video, MySpace, and many others. Many television news operations are also using Flash Video on their websites. Most of those sites accept uploads in a handful of formats like QuickTime, mpeg4, or wmv, and then they convert it to flash or MP4 before actually putting it out on the net.

In addition to the nearly universal flash video player, FLV is popular because it gives one of the smallest file sizes after compression yet it retains fairly good quality.

If you self-host your own videos, you should convert them to flash for greatest compatibility with the highest number of Internet viewers.

Although FLV's are the most common format found on the web today, the standard is moving towards the use of using MP4 H.264 files within flash players as it is compatible with both online and mobile, not to mention some HTML5 browser support (Safari, Chrome).

.MPEG (Motion Picture Experts Group):

MPEG was developed by the Motion Picture Experts Group. This international group was established in 1988 to develop standards for digital audio and video formats but they're not the only group doing so as anyone who studies digital video files formats knows.

MPEG-4 Part 14 (.MP4):

MPEG-4 Part 14 is a great sharing format for the internet. It's small but looks fairly clean. It's the video format employed by a growing number of camcorders and cameras and it is highly recommended.

In fact, YouTube recommends using MP4 format. YouTube accepts multiple formats, and then converts them all to .flv or .mp4 for distribution.

As mentioned earlier, more and more online video publishers are moving to MP4(with H.264 as the video compression codec) as the standard internet sharing format with use within both Flash players as wellas HTML5. This is the **format that we recommend** for online delivery.

.WMV (Windows Media Video)

A .WMV file indicates a windows media video file. Windows Media Video is used for both streaming and downloading content via the Internet. Microsoft's Windows Media Player, an application bundled with Windows operating systems, is built for WMV files. WMV files are

tiny. WMV will give you one of the smallest final file sizes. As you might expect, this means they are compressed so much they really do not look very good. In fact, I'd say the resolution is pretty crummy. But a tiny file size can be a real advantage for some things. If you get an email with an actual video attached instead of just a link to a video, it is probably a wmv file. They are the only ones small enough to attach to an email.

.MOV:

.MOV is the file extension used to identify an Apple Quick Time Movie. .MOV is an extremely common sharing format. It is considered one of the best looking and it does look great but the files sizes are big.

---

3. Unit Summary:

This unit provides brief details about animation files and internet, movie types and its supportability for internet.

---

4. Keywords:

Animation Files, Internet, Movie types and its Uses.

---

5. Exercises:

---

1) Explain the various Animation files for internet?

2) List the Movie types and its uses in supportability for the web.

16.6 References:

---

1. Prabat K Andleigh and Kiran Thakrar, ―Multimedia Systems and Design‖, PHI, 2003. 2.

Donald Hearn and M.Pauline Baker, ―Computer Graphics C Version‖, Pearson Education, 2003.

3. Designing Interfaces by Jenifer Tidwell
4. Digital Multimedia by Nigel Chapman

5.Information Architecture for the World Wide Web: Designing Large-Scale Web Sites by Peter

Morville, Louis Rosenfeld

6. Information Visualization, Second Edition: Perception for Design by Colin Ware
7. Letting Go of the Words: Writing Web Content that Works by Janice Redish
8. Multimedia Applications (X.media.publishing) by Ralf Steinmetz
9. Now You See It: Simple Visualization Techniques for Quantitative Analysis by Stephen Few
10. Visual Thinking: for Design by Colin Ware
11. Writing for New Media, Third Edition: Content Development for Bloggers and Professionals by Timothy Paul Garrand
12. Real-Time Video Compression: Techniques and Algorithms By Raymond Westwater Publisher: Springer 1997

THANKS

BEST OF LUCK for Your Study to all dear Students