

Multimedia System

Course Code: CACS457 | Year/Semester: IV / VIII

Unit 4: Video and Animation

(6 Hours)

Video and animation bring together everything from earlier units — images, timing, and motion — into one continuously moving experience. This unit covers how digital video is represented and stored, how computer-based animation is created and controlled, and how both are eventually displayed and transmitted to viewers.

1. Digital Video

Digital video is a sequence of digital images (called frames) displayed one after another fast enough that the human eye perceives continuous motion rather than separate still pictures. This illusion relies on a property of human vision called persistence of vision, where the eye/brain briefly retains an image after it disappears.

Digital Video = A Rapid Sequence of Still Images (Frames)

Playing frames quickly (e.g. 24-30 frames/second) creates the illusion of motion

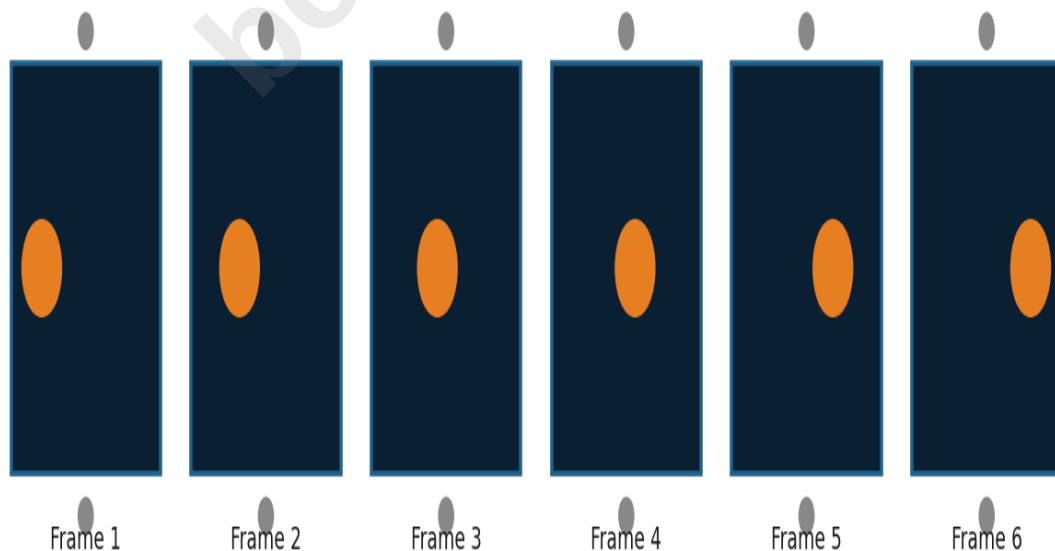


Figure 1: Digital video is simply many still image frames played back in rapid succession.

Key Properties of Digital Video

Property	Plain-English Meaning
Frame rate	The number of frames shown per second (fps); common values are 24, 25, 30, and 60 fps
Frame resolution	The pixel width × height of each individual frame (e.g. 1920 × 1080)
Bit depth	The number of bits used to represent the color of each pixel in a frame
Aspect ratio	The ratio of frame width to height, e.g. 16:9 (widescreen) or 4:3 (standard)
Bit rate	The amount of video data processed per second, usually in Mbps (megabits per second)

Note: A higher frame rate gives smoother motion, but also produces more data — this trade-off between smoothness/quality and file size/bandwidth runs throughout this entire unit.

Example 1: Calculating Total Frames and Raw Data Rate

Question: A video is recorded at 30 frames per second (fps), with each frame having a resolution of 640 × 480 pixels and 24-bit color depth. For a 10-second clip, calculate (a) the total number of frames, and (b) the uncompressed data rate in Mbps.

Solution / Explanation:

(a) Total frames = Frame rate × Duration = $30 \times 10 = 300$ frames.

(b) Data per frame (bits) = Width × Height × Bit depth = $640 \times 480 \times 24 = 7,372,800$ bits.

Data rate (bits/second) = Data per frame × Frame rate = $7,372,800 \times 30 = 221,184,000$ bits/second.

Convert to Mbps: $221,184,000 \div 1,000,000 \approx 221.2$ Mbps.

This huge raw data rate is exactly why video compression (Section 3) is essential for practical storage and streaming.

2. Video Signal Representation

Before discussing file formats, it helps to understand how the video signal itself — the way frames are scanned and colors encoded — is represented, both historically (analog) and digitally.

Scanning: Interlaced vs Progressive

Traditional analog and some digital broadcast systems use interlaced scanning, where each frame is split into two 'fields' — one containing only the odd-numbered horizontal lines, the other only the even-numbered lines — drawn alternately to save bandwidth. Modern digital displays mostly use progressive scanning, where every line of a frame is drawn in a single pass, giving a sharper, more stable image, especially for fast motion.

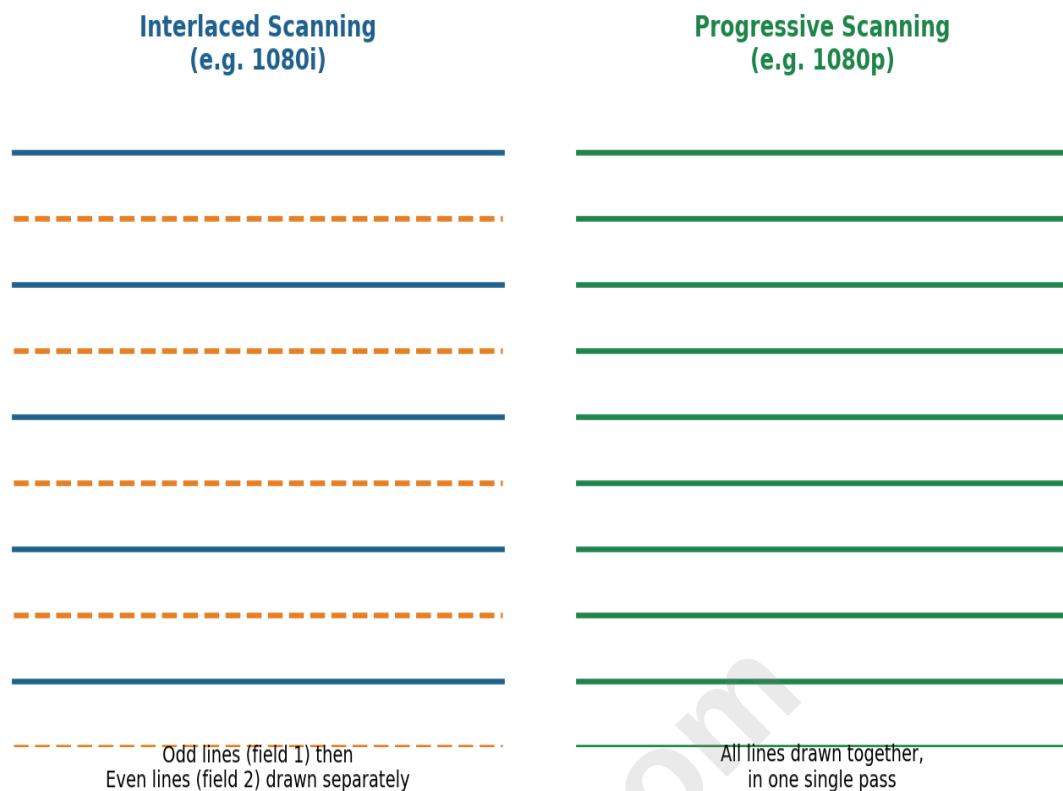


Figure 2: Interlaced scanning draws odd and even lines separately; progressive scanning draws every line in one pass.

Color Representation: YUV / YCbCr

While images are often stored using RGB (Unit 3), video signals are commonly represented using a luminance-chrominance model such as YUV or YCbCr: 'Y' carries brightness (luminance), while 'U/Cb' and 'V/Cr' carry color information (chrominance). This separation is useful because the human eye is far more sensitive to changes in brightness than to changes in color, so chrominance data can be sampled at a lower resolution than luminance without a noticeable quality loss — a technique called chroma subsampling (e.g. the common '4:2:0' scheme).

Term	Meaning
Y (Luma)	Brightness/intensity information of the image — essentially the grayscale version
U / Cb	Blue-difference chrominance component (color information)
V / Cr	Red-difference chrominance component (color information)
Chroma subsampling (e.g. 4:2:0)	Storing color information at lower resolution than brightness, since the eye is less sensitive to color detail

Note: Chroma subsampling alone can reduce raw video data by roughly half before any further compression is applied, simply by exploiting a limitation of human color perception.

3. Computer Video Format

A digital video file is generally made of two separate things working together: a codec (coder-decoder) that compresses and decompresses the actual video/audio data, and a container format that packages the compressed video stream, audio stream, subtitles, and metadata together into one file.

Typical Digital Video Pipeline: Capture to Display

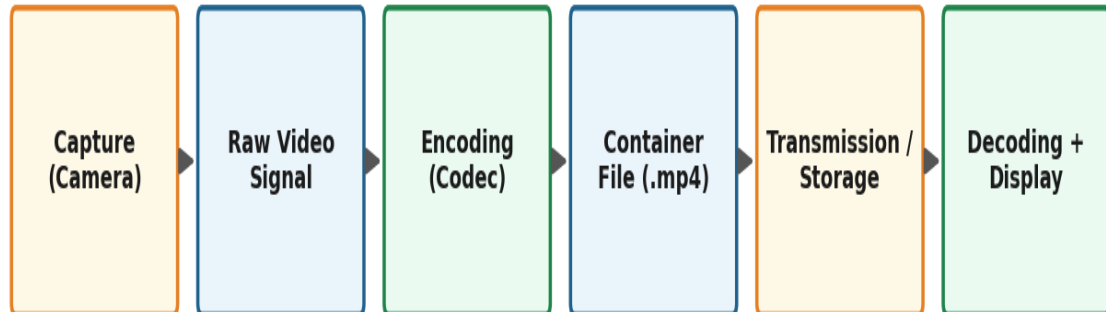


Figure 3: A typical digital video pipeline — from capture, through encoding and packaging, to final display.

Common Container Formats

Container	Typical Use
MP4 (.mp4)	Most widely used container — streaming, mobile devices, general distribution
AVI (.avi)	Older Windows container, less efficient but still widely supported
MOV (.mov)	Apple QuickTime container, common in professional video editing
MKV (.mkv)	Open-source container supporting many tracks (multiple audio/subtitles)
WebM (.webm)	Open, royalty-free container designed for efficient web streaming

Common Video Codecs

Codec	Plain Description
H.264 / AVC	Extremely widely used, very good compression efficiency, broad compatibility
H.265 / HEVC	Successor to H.264, roughly double the compression efficiency, used for 4K video
VP9 / AV1	Open, royalty-free codecs developed for efficient web video streaming
MPEG-2	Older codec, used historically for DVDs and digital broadcast television

Note: A common point of confusion: '.mp4' is a container, not a codec. The same .mp4 file could internally use H.264, H.265, or another codec — the container just holds the data.

4. Computer-Based Animation

Computer-based animation is the process of using a computer to create the illusion of movement by generating and displaying a sequence of images, rather than filming real-world motion as in video. It ranges from simple 2D cartoon-style animation to complex, physically realistic 3D animation used in films and games.

Unlike traditional hand-drawn ('cel') animation, where artists draw every single frame by hand, computer-based animation can automatically generate many of the in-between frames, dramatically reducing the manual work required (see Sections 6-7 on tweening).

Type	Description
2D animation	Flat, two-dimensional artwork animated using frames, paths, or tweening (e.g. cartoons, UI motion graphics)
3D animation	Objects modeled in three dimensions and animated by manipulating a virtual skeleton/rig, then rendered into 2D frames
Motion graphics	Animated text, shapes, and graphic design elements, often used in titles, advertisements, and explainer videos
Stop-motion (digital)	Physical objects photographed frame-by-frame and assembled digitally into an animated sequence

5. Animation Language

An animation language is a specialized programming or scripting language (or a structured description format) used to define how objects move, change, and behave over time, instead of (or in addition to) manually positioning every keyframe by hand in a graphical tool.

Animation languages typically let a designer or programmer describe: which objects exist on the scene, what properties they have (position, rotation, scale, color, opacity), how those properties should change over time (timing, easing/interpolation curves), and rules for interaction or triggering (e.g. 'start this animation when the button is clicked').

Example / Type	Plain Description
Declarative animation languages (e.g. CSS animations, SMIL)	Describe the start state, end state, and timing; the system computes the motion automatically
Scripting-based animation (e.g. JavaScript-driven, ActionScript)	Use program code to directly calculate and update object properties frame by frame
Scene description / rigging languages (used in 3D tools)	Describe object hierarchies, skeletons, and constraints that drive complex character animation

Note: Animation languages matter because they make animation reusable, parametric, and programmable — for example, the same script can animate a hundred different objects without an artist manually keyframing each one.

6. Timeline and Frame-Based Animation

Many animation tools organize work using a timeline — a horizontal track divided into discrete frames, similar to a film strip, where the animator can see and control exactly what happens at each moment in time.

In frame-based animation, the animator (or artist) defines the content of every single frame individually along the timeline — much like traditional hand-drawn animation, except now done inside software. This gives complete, precise control over every frame but is the most labor-intensive approach, since nothing is calculated automatically.

Aspect	Description
Control	Maximum — every frame is manually defined
Effort required	High — large numbers of frames must each be created by hand
Best suited for	Highly expressive or stylized animation where automatic interpolation would look wrong (e.g. traditional cartoon-style movement)
Typical tools	Traditional 2D animation software with frame-by-frame drawing support

7. Timeline and Tween-Based Animation

Tween-based animation ('tweening' is short for 'in-betweening') is a far more efficient approach: the animator only defines the important keyframes — the starting and ending state of an object's properties — and the software automatically calculates all the frames in between by interpolating (smoothly transitioning) between those keyframe values.

Tween-Based Animation: Keyframes + Computed In-Between Frames

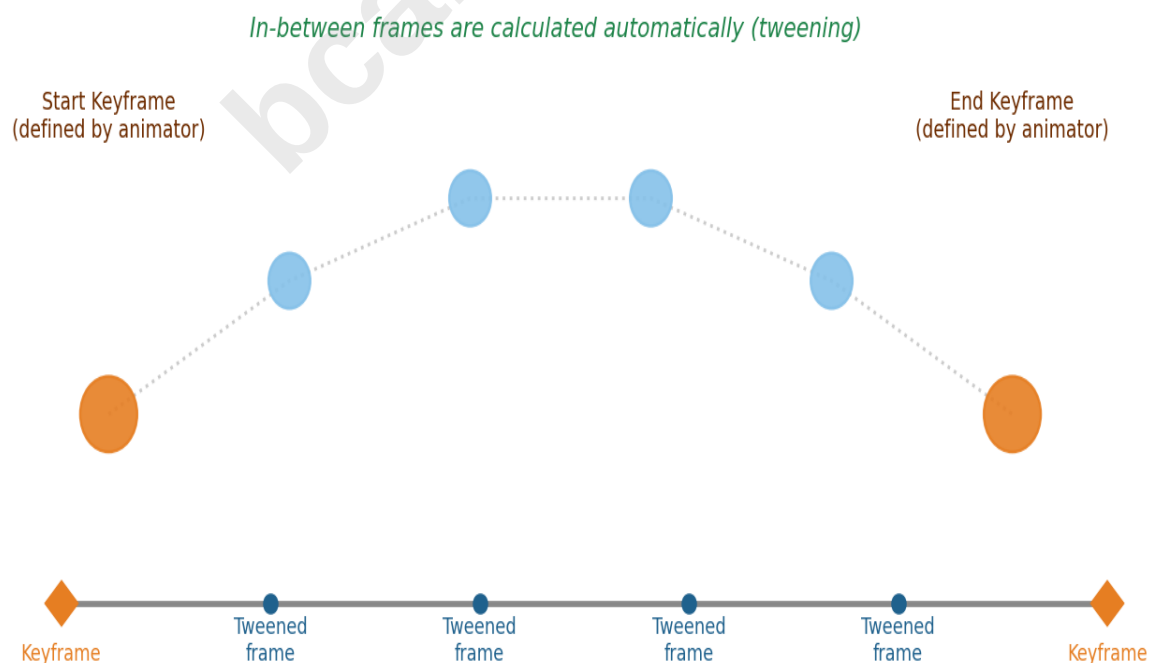


Figure 4: In tween-based animation, only the keyframes are set manually; every in-between frame is computed automatically.

Two common types of tweening are motion tweening (interpolating an object's position, rotation, or scale over time) and shape tweening (smoothly morphing one shape into a different shape over time). The mathematical function controlling how values change between keyframes (e.g. constant speed versus gradually speeding up/slowing down) is called an easing function.

Example 2: Calculating a Tweened (Interpolated) Position

Question: An object is animated moving in a straight line. At keyframe 1 (time = 0s) its x-position is 50 pixels; at keyframe 2 (time = 4s) its x-position is 250 pixels, using simple linear interpolation. What is the object's x-position at time = 1.5 seconds?

Solution / Explanation:

Formula (linear interpolation): $\text{Position} = \text{Start} + (\text{End} - \text{Start}) \times (\text{elapsed time} \div \text{total time})$.

$\text{Position} = 50 + (250 - 50) \times (1.5 \div 4)$.

$\text{Position} = 50 + 200 \times 0.375 = 50 + 75 = 125$ pixels.

At 1.5 seconds into the 4-second motion, the object has moved 37.5% of the way from its start to end position, landing at $x = 125$ pixels — this calculation is exactly what 'tweening' performs automatically for every frame in between.

Aspect	Frame-Based Animation	Tween-Based Animation
Who creates in-between frames	The animator, manually	The software, automatically (interpolation)
Effort required	High	Low to moderate
File size impact	Larger (every frame is unique data)	Smaller (only keyframe data needs to be stored)
Editability	Must redraw frames to make changes	Easy — just adjust keyframe values or easing

8. Methods of Controlling Animation

Beyond simple keyframe tweening, several other methods exist for controlling how animated objects move and behave, depending on how much realism, automation, or interactivity is needed.

Method	Plain Description
Keyframe animation	Animator sets key states; software interpolates between them (see Section 7)
Procedural animation	Motion is generated by algorithms/rules/physics simulation rather than hand-set keyframes (e.g. cloth, water, particle effects)
Kinematics (forward/inverse)	Forward kinematics moves a hierarchy (e.g. skeleton) from the root outward joint by joint; inverse kinematics lets the animator move an end point (e.g. a hand) and automatically calculates the joint angles needed
Motion capture (mocap)	Real human or object movement is recorded using sensors/cameras and mapped onto a digital character

Method	Plain Description
Behavioral / AI-driven animation	Characters react and animate dynamically based on rules or learned behavior, rather than a fixed predetermined sequence
Path-based animation	An object's motion is constrained to follow a predefined path or curve in the scene

Note: Modern productions often combine several of these methods — for example, a game character might use motion capture for walking, inverse kinematics so feet stay planted on uneven ground, and procedural animation for hair or cloth movement.

9. Display of Animation

Once an animation has been created, it must be displayed (rendered to the screen) smoothly. This involves the same underlying display concepts used for video playback, with a few animation-specific considerations.

Concept	Plain-English Meaning
Refresh rate	How many times per second the display itself redraws the screen (e.g. 60 Hz, 120 Hz)
Frame buffer	Memory area holding the pixel data of the frame currently being prepared/shown
Double buffering	Drawing the next frame in a hidden buffer while the current frame is shown, then swapping — avoids visible flicker/tearing
Real-time rendering	Calculating and displaying each animation frame on the fly (e.g. in games, interactive apps) rather than playing back pre-rendered frames
Pre-rendered playback	Displaying a sequence of frames that were already fully computed and saved in advance (e.g. animated films)

Note: If an animation's frame rate cannot keep up with the display's refresh rate — for example because real-time rendering is too slow — the result is visible stuttering or 'dropped frames', even if the underlying animation data itself is perfectly smooth.

10. Transmission of Animation

Sending animation data from one system to another — for example streaming an animated video, or loading an interactive animated web page — requires balancing visual quality against available bandwidth, similar to general video transmission.

Two broad transmission strategies are used, depending on what is actually being sent:

Strategy	What Is Transmitted	Trade-off
Pre-rendered video transmission	Final compressed video frames (the animation has already been fully rendered into a video, then compressed like any other video)	Looks identical everywhere, but file size depends on resolution/duration like normal video
Vector/instruction-based transmission	The animation's underlying instructions/keyframes/vector shapes themselves (similar in spirit to how MIDI sends instructions, not audio, from Unit 2)	Extremely small file size and scalable quality, but playback device must compute the animation itself

For streaming pre-rendered animation/video, techniques such as adaptive bitrate streaming (automatically switching between higher and lower quality versions depending on the viewer's network speed) and buffering (downloading a little ahead of playback to absorb network slowdowns) are commonly used to keep playback smooth.

Note: This mirrors the MIDI-versus-digital-audio trade-off from Unit 2: instruction-based animation transmission is tiny but playback-dependent, while pre-rendered video transmission is large but guarantees identical playback everywhere.

Key Takeaways — Unit 4

- Digital video is a rapid sequence of frames; frame rate, resolution, bit depth, and bit rate together determine quality and file size.
- Video signals are commonly represented using interlaced or progressive scanning, and luminance-chrominance models (YUV/YCbCr) with chroma subsampling to save data.
- A video file combines a codec (e.g. H.264, H.265) for compression with a container format (e.g. MP4, MKV) that packages video, audio, and metadata together.
- Computer-based animation generates motion using software rather than filming real life, spanning 2D, 3D, motion graphics, and digital stop-motion.
- Animation languages let designers/programmers describe object properties and motion over time, enabling reusable and programmable animation.
- Frame-based animation requires manually defining every frame, while tween-based animation only needs keyframes, with in-between frames calculated automatically.
- Animation can be controlled through keyframes, procedural rules/physics, forward/inverse kinematics, motion capture, or AI-driven behavior.
- Displaying animation smoothly depends on refresh rate, frame buffering, and whether rendering happens in real time or is pre-rendered.
- Animation can be transmitted as pre-rendered compressed video (large, consistent playback) or as instructions/vector data (small, but requires playback-side computation).

— End of Unit 4 —