# Video Communications and Video Streaming

John G. Apostolopoulos
Streaming Media Systems Group
Hewlett-Packard Laboratories

# Video Communication Applications

- Video storage, e.g. DVD or Video CD
- Videophone over PSTN
- Videoconferencing over ISDN
- Digital TV
- Video streaming over the Internet
- Wireless video
  - Videophone over cellular (Dick Tracy's watch)
  - Video over 3G and 4G networks: Interactive games, etc.

# Outline of Today's Lecture

- Properties of Video Communication Applications
- Brief case studies:
  - Video storage, e.g. DVD
  - Digital television
- Video streaming over the Internet
  - Bandwidth problem → Rate control
  - Delay jitter → Playout buffer
  - Loss → Error control

# Properties of Video Communication Applications

Wide range of different video communication applications with different operating conditions or different properties:

- Broadcast
- Multicast
- Point-to-point
- Pre-encoded (stored) video
- Interactive / real-time or non-real-time
- Dynamic or static channels
- Packet-switched or circuit-switched network
- Quality of Service (QoS) support
- Constant or variable bit rate channel

*The specific properties of a video communication application strongly influence its design*

Video Streaming

# Properties of Video Communication Applications (cont.)

- *Broadcast*
  - One-to-many (basically one-to-all)
  - Typically different channels characteristics for each recipient
  - Sometimes, system is designed for worst case-channel
  - Example: Broadcast television
- *Multicast*
  - One-to-many (but not everyone)
  - Example: IP-Multicast over the Internet
  - More efficient than multiple unicasts

Video Streaming

# Properties of Video Communication Applications (cont.)

- *Point-to-point*
  - One-to-one
  - Properties depend on *available back channel*:
    - With back channel: Receiver can provide feedback to sender → sender can adapt processing
    - Without back channel: Sender has limited knowledge about the channel
  - Examples: Videophone, unicast over the Internet

# Properties of Video Communication Applications (cont.)

- *Pre-encoded (stored) video*
  - Decoder retrieves a previously compressed video that is stored (locally or remotely)
  - Limited flexibility, e.g. often preencoded video can not be significantly adapted to current situation
  - Examples of locally stored: DVD or Video CD
  - Examples of remotely stored: Video-On-Demand (VOD), RealNetworks & Microsoft coded content

Video Streaming

# Properties of Video Communication Applications (cont.)

- *Real-time (or interactive) vs non-real-time*
  - Real-time: Information has *time-bounded usefulness*, e.g. if the info arrives, but is late, it is useless
  - Equivalent to maximum acceptable latency on transmitted information
  - Non-real-time: Loose latency constraint (many secs)
  - Examples of real-time: Videophone or videoconferencing, interactive games

# Properties of Video Communication Applications (cont.)

- *Dynamic (time-varying) vs static channels:*
  - Most communication involve channels whose characteristics vary with time, e.g. capacity, error rate, delay
  - Video communication over a dynamic channel is much more difficult than for a static channel
  - Examples of dynamic channels: Internet, wireless
  - Examples of largely static channel: DVD, ISDN

# Properties of Video Communication Applications (cont.)

- *Packet-switched vs circuit-switched network*
  - Packet-switched: Packets may exhibit variable delay, may arrive out of order, or may be lost completely
  - Circuit-switched: Data arrives in order, however may be corrupted by bit errors or burst errors
  - Example of packet-switched: LAN, Internet
  - Example of circuit-switched: PSTN, ISDN
- *Quality of Service (QoS) support*
  - Types of service: Guarantees on bandwidth, maximum loss rates or delay
  - Network QoS support can greatly facilitate video communication
  - Networks that support QoS: PSTN, ISDN
  - Networks w/o QoS support: Current Internet (best effort, e.g. no guaranteed support)

# Properties of Video Communication Applications (cont.)

- *Constant bit rate (CBR)* or *variable bit rate (VBR)* coding
  - Constant bit rate leads to variable quality
  - Variable bit rate can enable constant quality
  - Example of CBR: Digital TV, videoconferencing over ISDN
  - Example of VBR: DVD

# Basic Video Coding Question: VBR vs CBR coding

- Question: How many bits should we allocate to code each frame?

# How to Allocation Bits Among Frames?

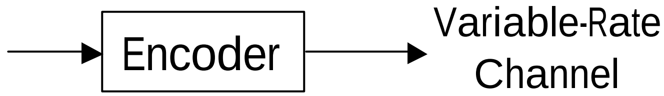
- Digitized (uncompressed) video has a constant rate

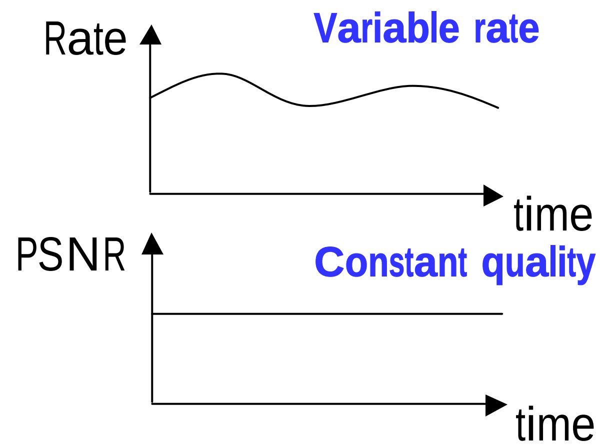
$$\frac{480 \times 720\, pixels}{frame} \times \frac{30\, frames}{\sec} \times \frac{24 bits}{pixel} = 250M\, {}^{bits}\!/_{\sec}$$

- Question: Compress at a constant bit rate? Variable rate?
- Observations:
  - Some frames are more complex than others, or are less predictable than others, and therefore require more bits
  - E.g., to achieve constant quality for every frame, a high complexity frame would require more bits than a low complexity frame

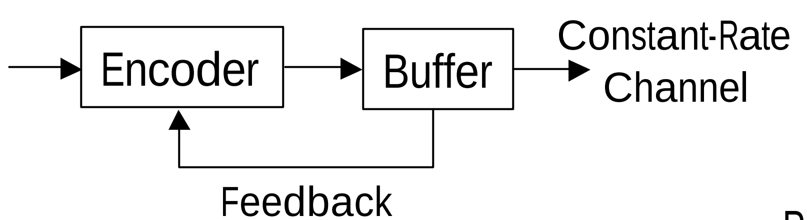
# VBR vs CBR Coding

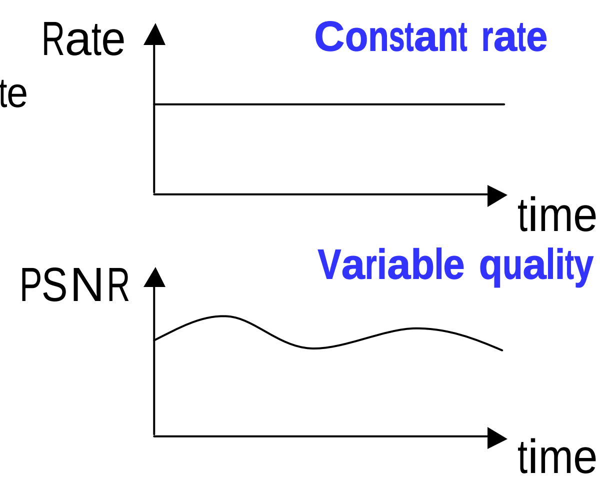
- Variable Bit Rate (VBR)




- Constant Bit Rate (CBR)



# VBR vs CBR Coding (cont.)

- *Tradeoff between quality and bit rate:*
  - Constant quality → variable bit rate
  - Constant bit rate → variable quality
- Constant quality corresponds to approximately the same distortion per frame:
  - Can be achieved by constant quantization stepsize for all frames
- Constant bit rate corresponds to approximately the same bit rate per frame (or other unit of time):
  - Can be achieved by using a buffer and feedback to direct the encoding

# Outline of Today's Lecture

- Properties of Video Communication Applications
- Brief case studies:
  - Video storage, e.g. DVD
  - Digital television
- Video streaming over the Internet
  - Bandwidth problem → Rate control
  - Delay jitter → Playout buffer
  - Loss → Error control

# Video Coding for Storage

- Goal: *Store a video in storage with $R_{Total}$ bits*
  - Example: DVD, 2 hour movie in 4.7 GB
- Problem: *How do we encode the video for this storage constraint?*
- Possible approach # 1:
  - Allocate equal number of bits to each frame,

    For N frames:

$$R_i = \frac{R_{Total}}{N} \quad \text{where } R_i \text{ is bits for frame } i$$

  - Problem:
    - Some frames are more complex than others
    - Some frames are more predictable than others
    - $\rightarrow$ Some frames should be allocated more bits than others

Video Streaming

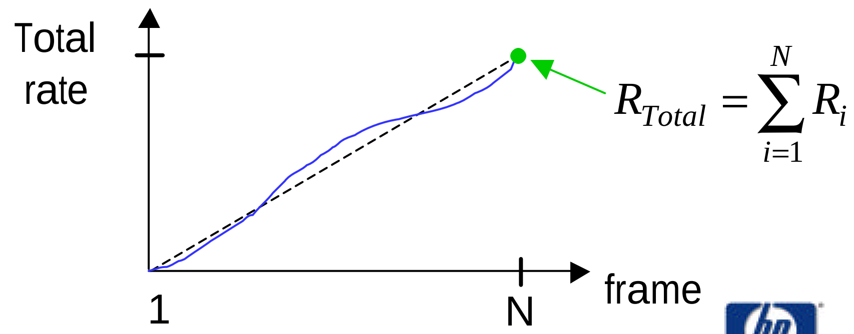
# Video Coding for Storage (cont.)

- Basic video coding problem for storage:

$$\text{minimize } D_{Total} = \sum_{i=1}^{N} D_i \text{ such that } R_{Total} = \sum_{i=1}^{N} R_i$$

$$D_i = \text{distortion for frame } i$$

- Possible approach # 2:
  - Allocate bits per frame so that on *average:* $R_i \approx \frac{R_{Total}}{N}$
  - Allow some variation
  - Ensure storage constraint is satisfied when Nth frame is coded



# Video Coding for Storage (cont.)

- Proposed approach # 2 (cont.):
  - Better than approach #1
  - Problem: Future frames are unknown
    - How many bits to allocate for them?
    - Can over estimate (too conservative)
      - Waste bits at end of sequence
    - Can under estimate
      - Not enough bits at end of sequence
    - → Either way sub-optimal quality
  - Basic Problem: *Future frames are unknown, have to guess how many bits to allocate for them*

# Video Coding for Storage (cont.)

- Idea: Video coding for storage doesn't require causal processing
  - Can *examine all frame* before encoding
  - *Perform global bit allocation* (we have a global constraint)
- Proposed approach #3:
  1. Code entire video sequence
  2. Gather and analyze statistics
  3. Identify complex areas of video sequence
  4. Re-estimate bit allocation for each frame
  5. Re-encode entire video sequence

  Repeat (steps 2–5)

  - *Multi-pass algorithm*: Process entire video multiple times
- Multi-pass coding can provide much better performance then single-pass coding

# Video Coding for Storage (cont.)

- Example of DVD:
  - MPEG-2 Main-profile @ main-level video
  - Storage constraint: 4.7 GB
  - VBR coding
  - Can use multi-pass encoding to optimize quality given global storage constraint

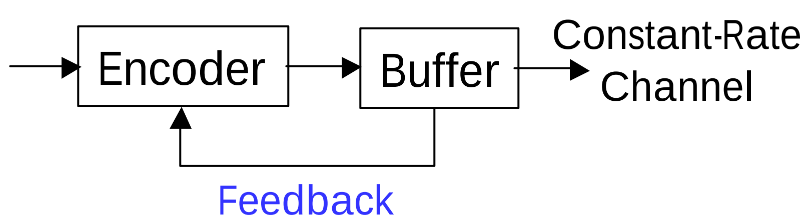
# Outline of Today's Lecture

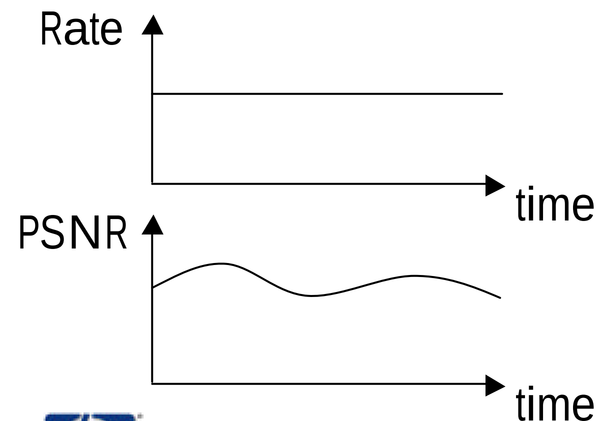
- Properties of Video Communication Applications
- Brief case studies:
  - Video storage, e.g. DVD
  - Digital television
- Video streaming over the Internet
  - Bandwidth problem → Rate control
  - Delay jitter → Playout buffer
  - Loss → Error control

# Video Coding for Digital Television

- Terrestrial (over-the-air) broadcast television
- Constraint: Constant bandwidth channel (20 Mb/s)

  → Requires *CBR coding*
- *Must regulate video bit rate*
  - Buffer to smooth instantaneous bit rate
  - Buffer control mechanism to control average bit rate
- Buffer feedback intuitively:
  - Quantizes *coarsely* if bit rate is too high
  - Quantizes *finely* if bit rate is too low



# Video Coding for Digital TV (cont.)

- Requirement:
  - Fast initialization and channel acquisition (turning on TV and changing channels)
  - Requires random access into video (½ sec OK)
- Solution: *Periodic I-frames*, MPEG GOP structure, one I-frame every ½ sec
- Remarks:
  - Simple solution, works well
  - Also used to provide random access for DVD
  - However, requires lots of bits for each I-frame
  - Impractical for many low-bit-rate applications

# Video Coding for Digital TV (cont.)

- Example of Digital TV:
  - MPEG-2 Main-profile @ high-level
  - Channel constraint: 20 Mb/s
  - CBR coding
  - Receiver initialization/channel acquisition: Random access via periodic I-frames (MPEG GOP structure)

- Prof Lim will discuss Digital TV in detail next Tuesday

# Outline of Today's Lecture

- Properties of Video Communication Applications
- Brief case studies:
  - Video storage, e.g. DVD
  - Digital television
- Video streaming over the Internet
  - Bandwidth problem $\rightarrow$ Rate control
  - Delay jitter $\rightarrow$ Playout buffer
  - Loss $\rightarrow$ Error control

Video Streaming

# Video Delivery over the Internet: File Download

*Download video:*

- Same as *file download*, but a LARGE file
- Allows simple delivery mechanisms, e.g. TCP
- Disadvantages:
  - Usually requires LONG download time and large storage space (practical constraints)
  - Download before viewing (requires patience)

# Video Delivery over the Internet: Streaming Video

*Streaming video:*

- Partition video into packets
- Start delivery, begin playback while video is still being downloaded (5-10 sec delay)
- *Simultaneous delivery and playback* (with short delay)
- Advantages:
  - Low delay before viewing
  - Minimum storage requirements

# Streaming Video: Sequence of Constraints

- Problem of streaming video can be expressed as a *sequence of constraints:*
  - Frame N must be delivered & decoded by time $T_N$
  - Frame N+1 must be delivered & decoded by time $T_N+\Delta$
  - Frame N+2 must be delivered & decoded by time $T_N+2\Delta$
- Any data that is lost is useless
- Any data that arrives late is useless
- Goal: Design system to satisfy this sequence of constraints

# Streaming Video over the Internet

- Problem: Internet only offers best-effort service
- No guarantees on:
  - Bandwidth
  - Loss rates
  - Delay jitter
- Specifically, these characteristics are *unknown* and *dynamic*
- Goal: Design a system to reliably delivery high-quality video over the Internet

Video Streaming

# Problems in Video Streaming over the Internet

Problems to be addressed include *unknown* and *dynamic*:

- Bandwidth
- Delay jitter
- Loss

- Many other problems also exist for streaming, but in the brief time available we focus on these three key video problems

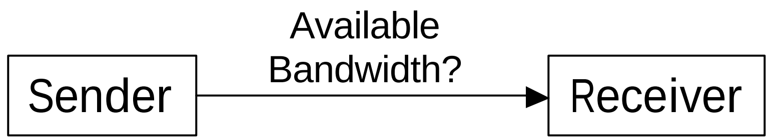
Video Streaming

# Problems in Video Streaming over the Internet

Problems to be addressed include unknown and dynamic :

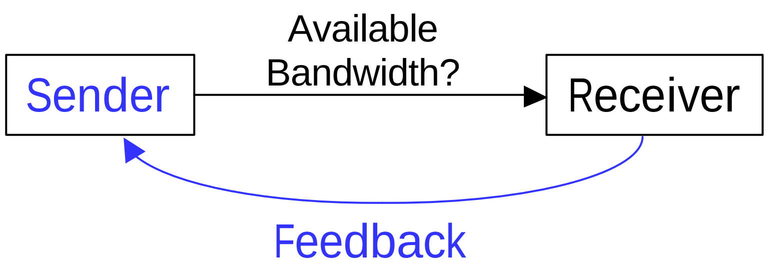
- *Bandwidth*
  - Can not reserve bandwidth in Internet today
  - Available bandwidth is dynamic
  - If transmit faster than available bandwidth
    - → Congestion occurs, packet loss, and severe drop in video quality
  - If transmit slower than available bandwidth
    - → Sub-optimal video quality
  - Goal: Match video bit rate with available bandwidth
- Delay
- Loss

# Overcoming the Bandwidth Problem: Rate Control




- *Rate control:*
  1. *Estimate* the available bandwidth
  2. *Match video rate* to available bandwidth
- Rate control may be performed at:
  - Sender
  - Receiver
- Available bandwidth may be estimated by:
  - Probe-based methods
  - Model-based (equation-based) methods

# Overcoming the Bandwidth Problem: Source-Based Rate Control




- *Source-based rate control:*
  - Source *explicitly adapts* the video rate
  - *Feedback* from the receiver is used to estimate the available bandwidth
  - Feedback information includes packet loss rate
- Methods for estimating available bandwidth based on packet loss rate:
  - Probe-based methods
  - Model-based methods

# Estimating Available Bandwidth: Probe-Based Methods

- *Probe-based methods:*
  - Basic idea: *Use probing experiments to estimate the available bandwidth*
  - Example: Adapt sending rate to keep packet loss rate ρ less then a threshold $P_{th}$
    - If ($\rho < P_{th}$) then increase transmission rate
    - If ($\rho > P_{th}$) then decrease transmission rate
  - Different strategies exist for adapting transmission rate
  - Simple, ad-hoc

# Estimating Available Bandwidth: Model-Based Methods

- *Model-based (equation-based) methods*
  - Goal: *Ensure fair competition* with concurrent TCP flows on the network, e.g. fair sharing of bandwidth
  - Basic idea:
    - *Model the average throughput* of a TCP flow
    - Transmit video with the *same throughput* as if is was a TCP flow

$$l = \frac{1.22 \times MTU}{RTT \times \sqrt{r}}$$

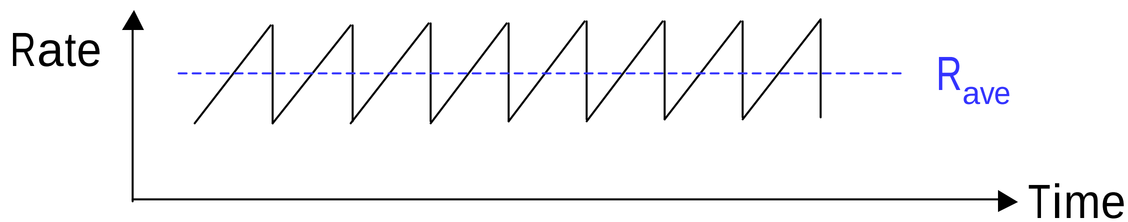
$l$ = Throughput of TCP
MTU = Maximum Transmit Unit (max packet size)
RTT = Round Trip Time
$r$ = Packet loss ratio

  - Similar characteristics to TCP flow on macroscopic scale (not microscopic)
  - Behaves macroscopically like a TCP flow, "fair" to other TCP flows, referred to as *"TCP-friendly"*

[Floyd, et.al.; Mathis et.al.; Tan, Zakhor]

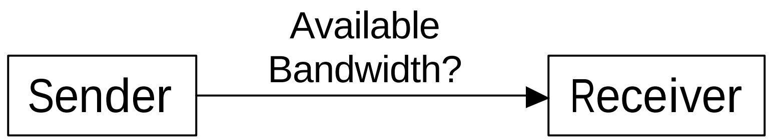
# Why not use TCP for Rate Control?

- *TCP*:
  - Guarantees delivery via retransmission, leading to *time-varying throughput and delay*
  - Additive-increase multiplicative-decrease (AIMD) rate control




- Problem: Oscillations are detrimental for streaming
- Therefore, exactly matching TCP traffic pattern is bad
- Instead, match TCP traffic pattern on a coarser (macroscopic) scale, e.g. same average throughput over a time-window
- Summary:
  - Exactly emulating TCP rate control (AIMD) is bad
  - TCP-friendly approaches attempt to share bandwidth fairly on a macroscopic scale

# Overcoming the Bandwidth Problem: Rate Control



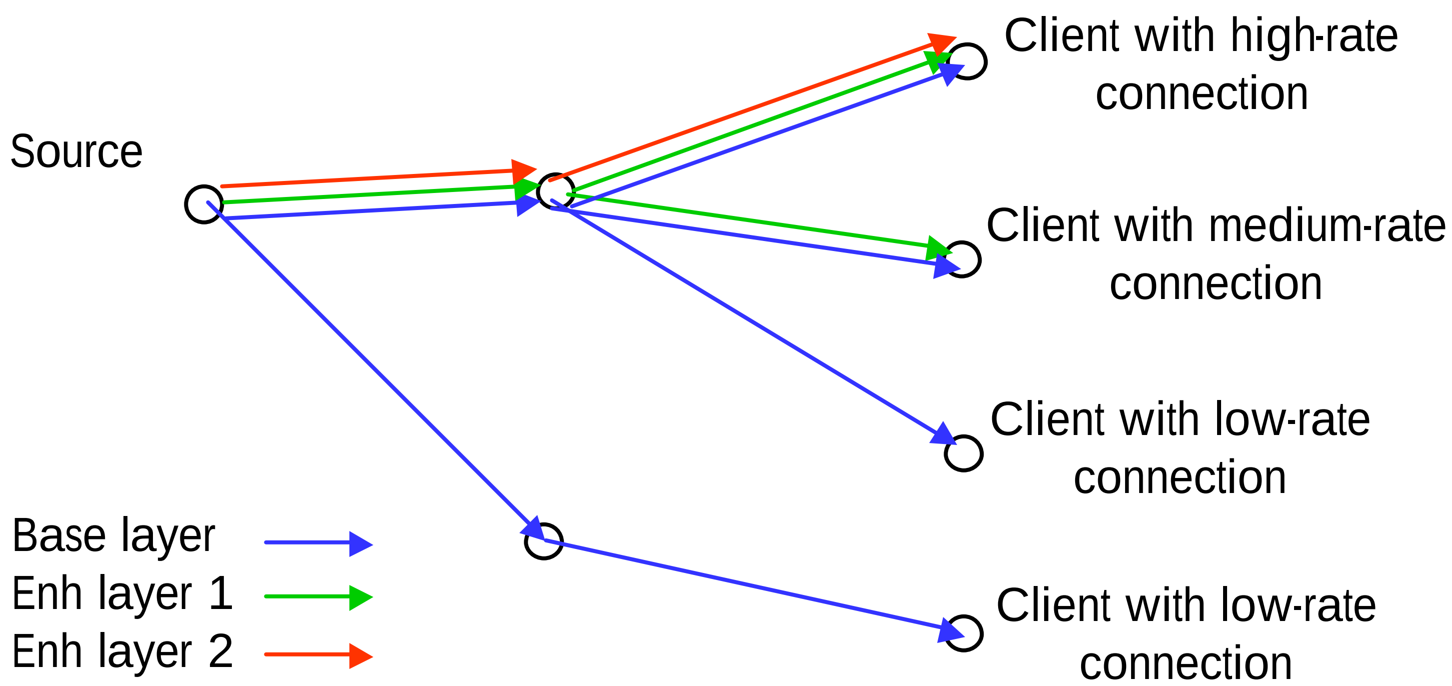

- Rate control:
  1. Estimate the available bandwidth
  2. Match video rate to available bandwidth
- Rate control may be performed at:
  - Sender
  - *Receiver*
- Available bandwidth may be estimated by:
  - Probe-based methods
  - Model-based (equation-based) methods

# Receiver-Based Rate Control

- *Receiver explicitly selects* the video rate from a number of possible rates
- Key example: Receiver-driven Layered Multicast
  - Sender codes video with scalable or layered coder
  - Sends different layers over different multicast groups
  - Each receiver estimates its bandwidth and joins an appropriate number of multicast groups
    - Receives an appropriate number of layers up to its available bandwidth

# Receiver-Based Rate Control (cont.)

- Example of *Receiver-Driven Layered Multicast* [McCanne, Jacobson, Vetterli]
  - Each client can join/drop layers



# Rate Control: Adapting the Video Bit Rate

- Source must match video bit rate with available bandwidth
- Video bit rate may be *adapted* by:
  - Varying the quantization
  - Varying the frame rate
  - Varying the spatial resolution
  - Adding / dropping layers (for scalable coding)
- Options depend on real-time encoding or pre-encoded content:
  - *Real-time encoding:* Adapting is straightforward
  - *Pre-encoded content:* Limited options, e.g. drop B-frames, drop layers in scalable coding, or perform transcoding

# Problems in Video Streaming over the Internet

Problems to be addressed include unknown and dynamic:

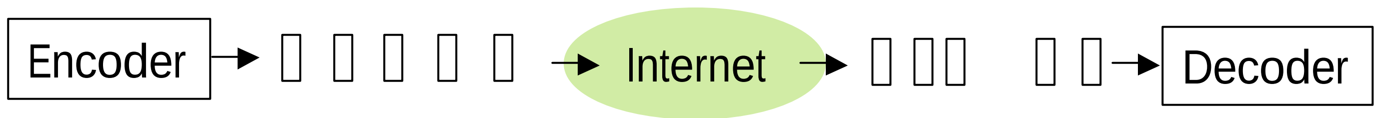
- Bandwidth
- *Delay jitter*
  - *Variable end-to-end packet delay*
  - *Compensate via playout buffer*
- Loss

# Why is Delay Jitter an Issue?

Example:

- Video encoder captures/sends video at a certain rate, e.g. 10 frames/sec or one frame every 100 ms
- Receiver should decode and display frames at the *same rate*
  - *Each frame has its own specific playout time*
  - *Playout time:* Deadline by which it must be received/displayed
- If a frame arrives after its playout time it is useless
- If subsequent frames depend on the late frame, then effects can propagate

# Delay Jitter




- End-to-end delay in Internet: Depends on router processing and queuing delays, propagation delays, and end system processing delays
- *Delay jitter:*
  - End-to-end delay may fluctuate from packet to packet
  - Jitter: Variation in the end-to-end delay
- Example: Video coded at 10 frames/sec
  - Each frame sent in one packet every 100 ms
  - Received packets may not be spaced apart by 100 ms
    - Some may be closer together
    - Some may be farther apart

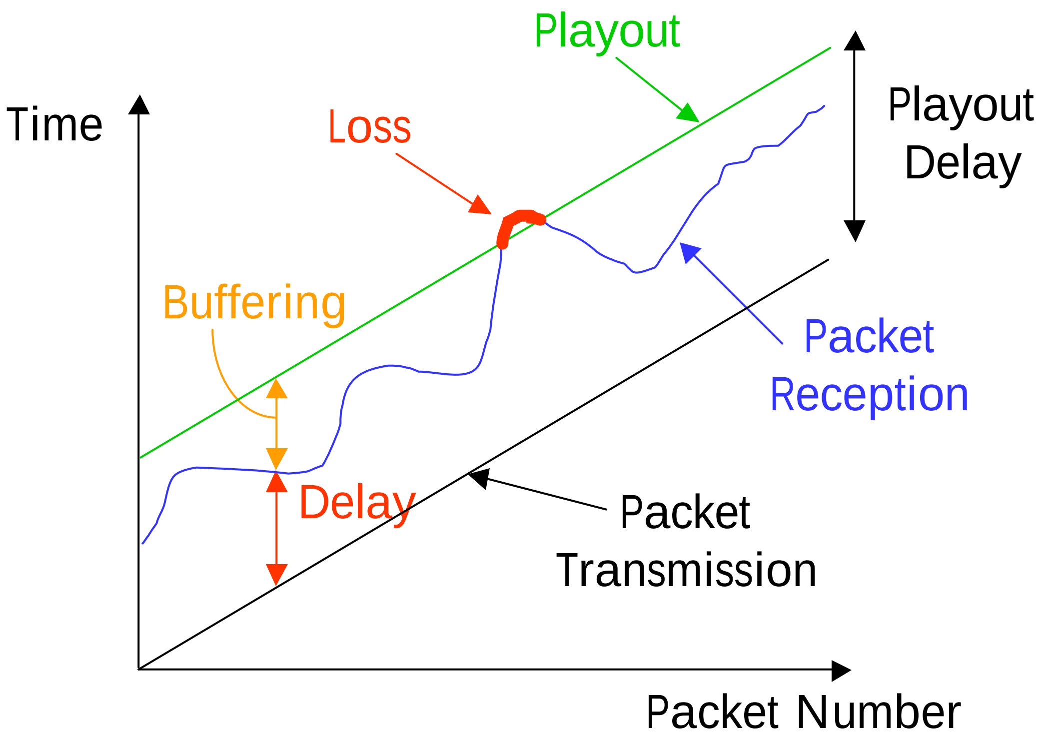
# Overcoming Delay Jitter: Playout buffer

- Goal: *Overcome delay jitter*
- Approach: Add *buffer at decoder to compensate for jitter*
- Corresponds to adding an offset to the playout time of each packet
  - If (packet delay < offset) then OK
    - Buffer packet until its playout time
  - If (packet delay > offset) then problem

Video Streaming

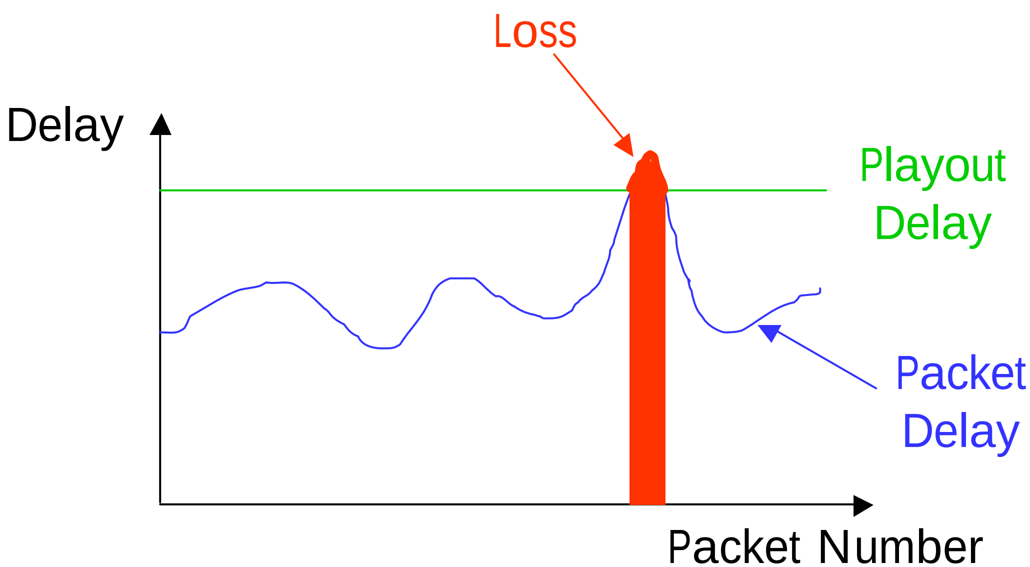
# Overcoming Delay Jitter: Playout Buffer (cont.)

- Packet delivery, time-varying delay (jitter), and playout delay:



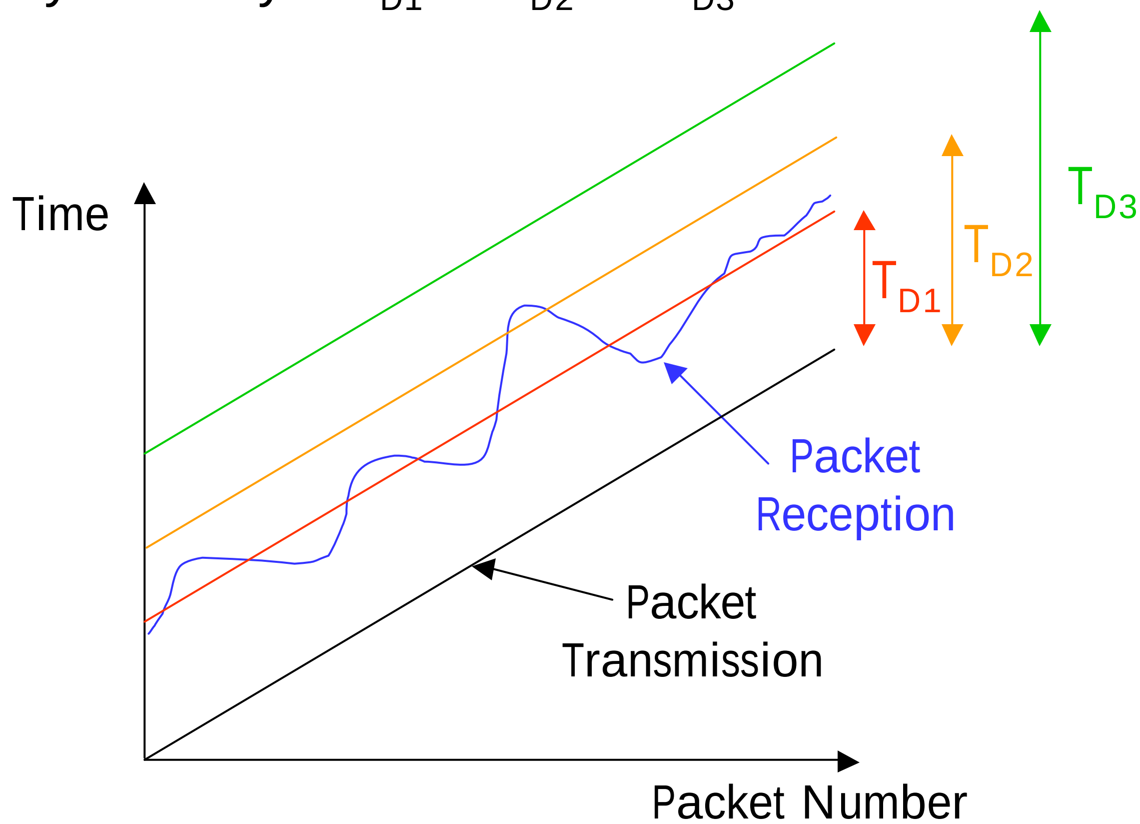
# Overcoming Delay Jitter: Playout Buffer (cont.)

- Delay per packet and effect of playout delay:



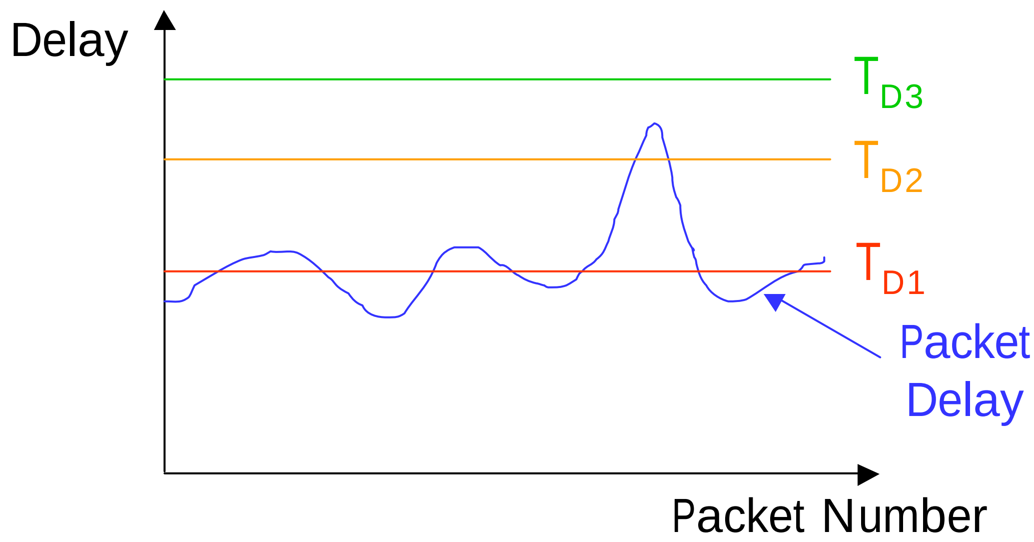
# Effect of Different Playout Delays

- Playout delays: $T_{D1} < T_{D2} < T_{D3}$



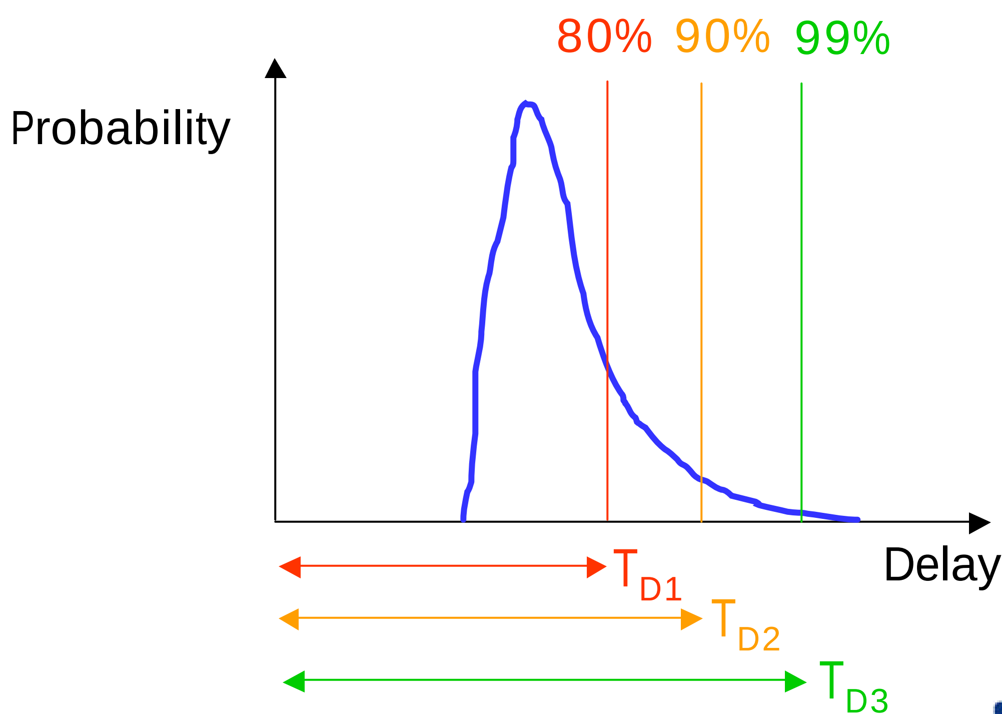
# Effect of Different Playout Delays (cont.)

- Playout delays: $T_{D1} < T_{D2} < T_{D3}$



# Effect of Different Playout Delays (cont.)

- As the playout delay is increased, the cumulative distribution of in-time packets is increased
- Note: (1) minimum transmit time, (2) long tail in the distribution



# Comments on Playout Delay

- Designing appropriate *playout strategy* is very important
- *Tradeoff between playout delay and loss*
  - Longer delay leads to lower loss rates
  - Shorter delay has higher loss rates
- Streaming of stored video *can tolerate* long delays (e.g. Real uses 5-10 secs)
- Real-time interactive video *can not tolerate* long delays (maybe 400 ms)
- Delay jitter is dynamic (time-varying)
  - Fixed playout delay is sub-optimal
  - *Adaptive playout* delay is better
    - Estimate variance of jitter and adapt playout delay

# Problems in Video Streaming over the Internet

Problems to be addressed include unknown and dynamic:

- Bandwidth
- Delay jitter
- *Loss*
  - *Overcome losses via error control:*
    - *Forward Error Correction (FEC)*
    - *Retransmission*
    - *Error concealment*
    - *Error-resilient video coding*

# Error Control

- Goal of error control:
  - To overcome the effect of errors such as packet loss on a packet network or bit or burst errors on a wireless link
- Types of error control:

**Channel Coding**
  - Forward Error Correction (FEC)
  - Retransmission

**Source Coding**
  - Error concealment
  - Error-resilient video coding

# Error Control

- Goal of error control:
  - To overcome the effect of errors such as packet loss on a packet network or bit or burst errors on a wireless link
- Types of error control:
  - *Forward Error Correction (FEC)*
  - Retransmission
  - Error concealment
  - Error-resilient video coding

# Forward Error Correction (FEC)

- Goal of FEC or channel coding: Add specialized redundancy that can be used to recover from errors
- Example: Overcoming losses in a packet network
  - Losses correspond to packet erasures
  - Block codes are typically used
  - K data packets, (N-K) redundant packets, total of N packets
  - Overhead N / K
  - Example:
    - 5 data packets, 2 redundant packets (K,N) = (5,7)
    - 7 / 5 = 1.40 or 40 % overhead

# FEC (cont.)

- Error correcting capability:
  - If no errors, then K data packets provide data
  - As long as any K of the N packets are correctly received the original data can be recovered

    (Assuming maximum distance separable (MDS) code)
  - Simplest case:
    - N = K+1
    - Redundant packet is parity packet, simplest form of erasure code
    - OK as long as no more than 1 out of N packets are lost
  - Example: 5 data packets, 2 redundant packets (5,7)
    - Can compensate for up to 2 lost packets
    - OK as long as any 5 out of 7 are received

# FEC and Interleaving

- Problem: *Burst errors* may produce more than N-K consecutive lost packets
- Possible solution: FEC combined with *interleaving* to spread out the lost packets
- FEC and interleaving often effective
- Potential problem:
  - To overcome long burst errors need large interleaving depth → Leads to large delay

# Summary of FEC

- Advantages:
  - Low delay (as compared to retransmits)
  - Doesn't require feedback channel
  - Works well (if appropriately matched to channel)
- Disadvantages:
  - Overhead
  - Channel loss characteristics are often unknown and time-varying
    - FEC may be poorly matched to channel
    - Therefore often ineffective (too little FEC) or inefficient (too much FEC)

# Error Control

- Goal of error control:
  - To overcome the effect of errors such as packet loss on a packet network or bit or burst errors on a wireless link
- Types of error control:
  - Forward Error Correction (FEC)
  - *Retransmission*
  - Error concealment
  - Error-resilient video coding

# Retransmissions

- Assumption: *Back-channel exists* between receiver and sender
- Approach: Receiver tells sender which packets were received / lost and *sender resends lost packets*
- Advantages:
  - Only resends lost packets, efficiently uses bandwidth
  - Easily adapts to changing channel conditions
- Disadvantages:
  - Latency (round-trip-time (RTT))
  - Requires a back-channel (not applicable in broadcast, multicast, or point-to-point w/o back-channel)
  - Effectiveness decreases with increasing RTT

# Retransmission (cont.)

Variations on retransmission-based schemes:

- Video streaming with time-sensitive data
  - *Delay-constrained retransmission*
    - Only retransmit packets that can arrive in time
  - *Priority-based retransmission*
    - Retransmit important packets before unimportant packets
  - Leads to interesting *scheduling problems*, e.g. which packet should be transmitted next?

# Joint Source-Channel Coding

- Data communication:
  - All data bits must be reliably delivered
- Video communication:
  - *Some bits are more important than other bits*
  - *It is not necessary for all bits to be reliably delivered*
- Idea: *Exploit the differing importance* in the video data
- Joint source-channel coding: Designing the source and channel coders to exploit the difference in importance

# Joint Source-Channel Coding (cont.)

Examples of coded video data with *different importance:*

- *Different frame types* have different importance (depending on dependencies between frames)
  - I-frame: Most important
  - P-frame: Medium importance
  - B-frame: Minimum importance (can be discarded)
- *Different layers in a scalable coder* have different importance
  - Base layer: Most important
  - Enhancement layer 1: Medium importance
  - Enhancement layer 2: Minimum importance

# Joint Source-Channel Coding (cont.)

- *Adapt error-control* based on importance of video data
  - FEC: *Unequal error protection*
  - Retransmit: *Unequal (prioritized) retransmit strategies*
- Example for I, P, and B frames:

| | I-frame | P-frame | B-frame |
|---|---|---|---|
| FEC | Maximum | Medium | Minimum (or none) |
| Retransmit | Maximum | Medium | Can discard |

Video Streaming

# Joint Source-Channel Coding (cont.)

- Example for scalable video coding:

| | Base Layer | Enhancement Layer #1 | Enhancement Layer # 2 |
|---|---|---|---|
| FEC | Maximum | Medium | Minimum (or none) |
| Retransmit | Maximum | Medium | Can discard |

# Review of Today's Lecture

- Properties of Video Communication Applications
- Brief case studies:
  - Video storage, e.g. DVD
  - Digital television
- Video streaming over the Internet
  - Bandwidth problem → Rate control
  - Delay jitter → Playout buffer
  - Loss → Error control
    - Forward Error Correction (FEC)
    - Retransmission
    - *Error concealment*
    - *Error-resilient video coding*

Next lecture { (Error concealment, Error-resilient video coding)