Conferencing
Conferencing and Computer-Supported Collaborative Work

Characteristics:

**synchronous:** simultaneous interaction; “video conferencing”

**asynchronous:** time-shifted: email, news groups, web, shared document editing, workflow computing (edit, sign, process), …

➡️ may want to use both

Can integrate audio and video into the Internet or use separate network (ISDN, plain old telephony services (POTS)).
Network impairments

duplicates: not uncommon for multicast; generally harmless

packet loss: up to several percent ➔ audible clicks, loss of encoding state; not as bad for conferencing video

delay: due to transmission on slow links, propagation (5 µs/km), switching

delay jitter: arrival distortion:

caused by queueing (resource contention) in nodes
Playout delay

The diagram illustrates the concept of playout delay, showing the comparison between packets generated or received at the sender and received by the receiver over time. The steps represent the timing of packet generation and receipt, highlighting the delay between the generation and receipt points.
A | B | C | D
---|---|---|---
play A | play B | play C (missed) | play D

sender

time

receiver

too late
Playout delay buffer

packet arrival

playout delay

loudspeaker
Playout delay

- only compensate for variable of delay $\delta$
- packets may be lost, reordered, gaps $\Rightarrow$ need timestamp $t$
- low loss (recording, seminar) $\leftrightarrow$ low delay (telephony, discussion)
- achieve minimum possible playout delay $\Rightarrow$ adaptation
- can adjust delay $D$ only at beginning of talkspurt

**relative timing:** play first packet after $D$ $\Rightarrow$ what if first few packets bunched?
   - Example: $D = 200$, $t = 0$, $\delta = 50$ $\Rightarrow$ playout at 250

**absolute timing:** maintain fixed relationship
   - Example: playout at 200 regardless of arrival of first packet

complications: clocks are not synchronized (drift and offset); “reboot”
Playout delay

- time of departure $t_i$, estimated network delay $\hat{d}_i$
- assume synchronized clocks for simplicity, but works without $\rightarrow$ clock offset = long network delay
- for first packet in talkspurt: $p_0 = t_0 + \hat{d}_0 + \mu \hat{v}$, where $\hat{v}$ is estimated delay variation
- mechanisms differ in computation of $\hat{d}$ and $\mu$
- for other packets in talkspurt: $p_j = p_i + t_j - t_i$
Playout delay

- \( \hat{d}_i = \alpha \hat{d}_{i-1} + (1 - \alpha) n_i \) where \( n_i \) is network delay, \( \alpha \approx 0.998 \)
- \( \hat{v} = \alpha \hat{v}_{i-1} + (1 - \alpha) |\hat{d}_i - n_i| \)
- \( \mu \) can be tuned to achieve a desired loss rate:

\[
\text{if } (p_C < p_L - \theta) \land (\mu \leq \mu_{\text{max}} - \delta_{\text{inc}}; \\
\quad \mu \leftarrow \mu + \delta_{\text{inc}}; \\
\text{else if } (p_C > p_L + \theta) \land (\mu \geq \mu_{\text{min}} + \delta_{\text{dec}}; \\
\quad \mu \leftarrow \mu - \delta_{\text{dec}}; \\
\text{else} \\
\quad \mu \leftarrow \mu
\]

- typical: \( \mu_{\text{max}} = 8, \theta = 0.05, \delta_{\text{inc}} = 0.4, \delta_{\text{dec}} = 0.2 \)
Playout delay performance

Trace 1

Application Loss Probability vs. Target Loss Probability for different methods:
- Exp-avg Ext
- Spk-det Ext
- Window Ext
- Prev-opt (Bin)
- Analytical
- Optimal
Handling packet loss

discover: gap in packet sequence (account for reordering)

retransmit: if enough time, ask for retransmission ➤ multicast dangerous!

forward error correction: like RAID ➤ transmit XOR of block of packets

redundancy: transmit low-fidelity version with delay

cover up: fill in waveform at receiver, e.g., based on prior and next block
Forward error correction

FEC over 1,2,3 yields 2 FEC blocks

- $(n, k) = (5, 3) = \text{transmit block of 5, need to receive } any \ 3 \text{ packets}$
- increases delay, network load $\rightarrow$ modest losses
Forward error correction

![Graph showing network loss probability vs. application loss probability for Exp-avg, Exp-avg (add (N-1)*pkt-length), and Exp-avg Ext.](image)
MBone Conferencing

- traditional (POTS, ISDN, ATM) conferences:
  - central server tracks participants
  - multipoint control unit replicates data
- minor disadvantages:
  - doesn’t scale
  - complicated when dealing with failures
  - single point of failure; network partitions!
- usually don’t need to know *immediately* when somebody enters or leaves the room (there are exceptions. . .)
- ideal for soft-state and IP multicast
- multicast convenient for large conferences: no need to inform all others or central server
- multicast ➠ anybody within ttl radius can get data ➠ need encryption for privacy
- but: unless you trust every provider, need it anyway
- –: multicast + encryption ➠ can still easily get some information on participants
- conference is visible to network ➠ re-use network facilities, avoid strange failure modes
- conferencing tools are much simpler

light-weight session model
mostly used for seminar-style conferences, but can also be extended to small, interactive groups
Conferencing architectures

- traditional: single application
- Internet (Mbone) tools: individual, standalone applications
  - separation of “engine” and GUI
  - possibly distribution across office-scale network
  - control much easier to make cross-platform than media
What if media agents could talk to each other?

- “start-and-forget” ➔ continuous involvement
- audio follows video: stereo placement according to window location
- video follows audio: enlarge image of speaker
- audio enables video: send at higher rate when talking
- auxiliary applications: recorders, talk timers, …
- floor controller controls audio, video
- SNMP agent retrieves statistics, controls media agent

must be easy to add, without access to source code
independent of conference control mechanism
Architecture

control flow in multimedia conferences:

**horizontal:** conference control (CC), between participants

**vertical:** local, one participant, between applications

components:

**controller:** conference controller, floor controller, . . .: \(\Rightarrow 1\!\!1\)

**media agents:** audio, video, whiteboard, . . .

reuse same media agents \(\leftrightarrow\) different CC protocols, styles
Architecture

classic controllers

icc
sd
invitation agent
floor controller

message replicator

pmm

media agents

NeVoT

audio system
tkaudio

Internet

UDP

ivs, vic, nv, ....

vat, RTP
Local coordination

- message reach
  - **all**: e.g., membership, floor control, conference state, ...
  - **application**: all audio tools
  - **specific**: configure one video tool

  → unicast, local multicast, local broadcast

- message reliability
  - **national**: ephemeral, refreshed (VU meter)
  - **response**: configuration
Examples

- vat/vic message bus (video-follows-audio)
- pmm (NeVoT, vic)
- message bus (mbus)