Video

Event-based programs

- `read()` is blocking ➠ server only works with single socket ↔ audio, network input
- need I/O multiplexing ➠ event-based programming
- also need to handle time-outs, connection requests
- all events (mouse clicks, windows, etc.) handled by event loop
- **do**
  wait for event(s)
  handle event (hopefully short)
  forever
- harder to maintain state, recursion
- alternative 1: “interrupts” (signals) ➠ called at any time
- alternaitve 2: threads (separate scheduling, same address space)

Multiplexing with `select()`

```
int select(int nfds, fd_set *readfds, fd_set *writefds,
  fd_set *exceptfds, struct timeval *timeout)
```

- block until $>=1$ file descriptors have something to be read, written, or an exception, or timeout
- set bit mask for descriptors to watch using `FD_SET`
- returns with bits for ready descriptors set ➠ check with `FD_ISSET`
- cannot specify amount of data ready

Audio timing

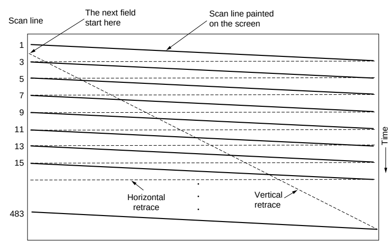
Need to write block of audio to speaker every t ms ($t = 20 \ldots 100$ ms) ➠

1. timer ➠
   - OS overhead
   - may not be accurate
   - error accumulation (time between timers)
   - clock may differ from audio sampling clock
2. use audio input: for every block read, write one audio block ➠ stay in sync
3. but: doesn’t work for half-duplex audio cards

Analog video

- black & white (RS170) + color burst subcarrier
- 15,750 Hz horizontal scanning

system	where	lines (disp.)	size	f/s	bw
NTSC	US, Japan	525 (483)	4:3	30	6 MHz
PAL	Europe	625 (576)	4:3	25	8 MHz



reduce flicker ➠ interlace scanning: 2 fields/frame

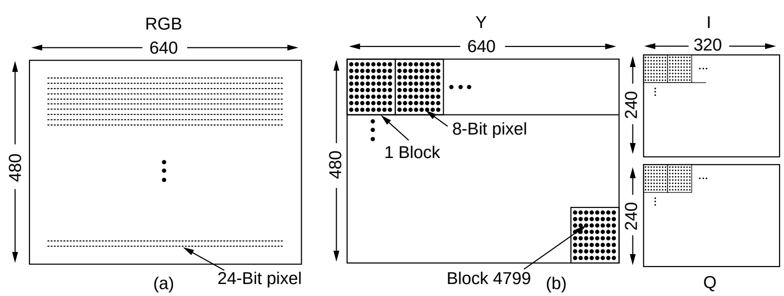
Digital video

format	lines	pixels
SCIF	480	704
CIF	288	352
QCIF	144	176

- motion video: 5 to 30 f/s
- camera ➠ Red, Green, Blue ➠ chrominance, luminance
- YUV: Y = luminance, UV: color difference (red, blue − Y)

Digital video

eye more sensitive to luminance ➠ subsample chrominance: 2:1:1, 4:1:1, 4:2:2



treat chrominance and luminance differently

Video coding

lossless (entropy): (X ray images!) run-length, statistical encoding (Huffman coding, . . .)

lossy: ➠ exploit spatial redundancy

transform: → frequency domain, higher quantization steps for higher frequencies

vector quantization: map NxN block into N^2-dimensional space and find closest in *codebook*

model-based: geometric description ➠ very low bit-rate

Images

X-ray digitization: 4000 pixels x 4000 lines, at 12 bits/pixel

slide: 120-150 dpi for 6.75” x 10.25”

70 mm movie: 2210 lines

35 mm film: 1753 lines

slide film: 100 lines/mm (2500 lines/inch)

HDTV: 1125 lines

fax: 200 lines/inch

Video coding: JPEG (Joint Photographic Experts Group)

- individual (still) pictures
- lossless or lossy
- typically about 2 Mb/s for video stream
- discrete cosine transform (DCT): samples ➠ blocks (16x16 Y, 8x8 UV) ➠ frequency domain 2D matrix, (0,0) = “DC”

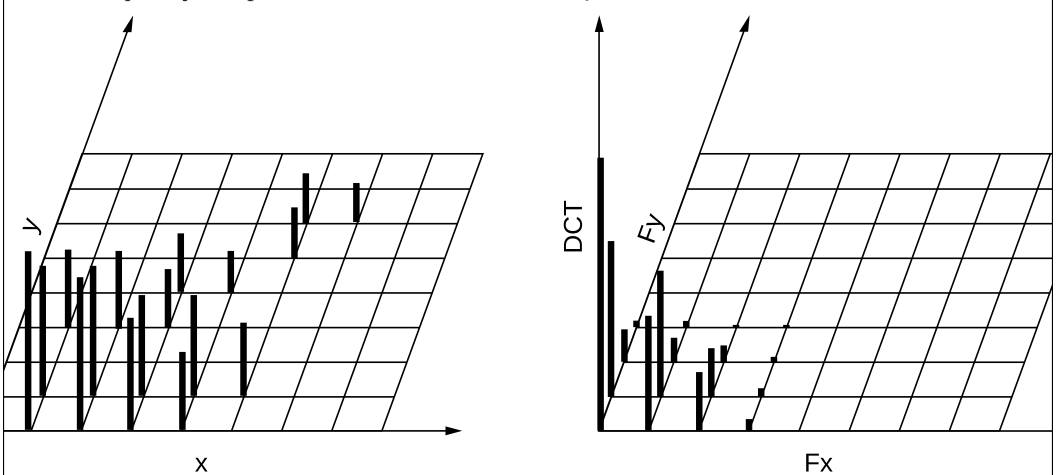
$$B_{u,v}(i,j) = \cos\left(\frac{(2i+1)u\pi}{16}\right)\cos\left(\frac{(2j+1)v\pi}{16}\right)$$

with the transformed image

$$F(u,v) = \frac{2}{N}C(u)C(v)\sum_{i=0}^{7}\sum_{j=0}^{7} f(i,j)B_{u,v}(i,j)$$

Video coding: JPEG

code frequency sampled with different resolution Q



Video coding: JPEG (Joint Photographic Experts Group)

DC or average (0,0) value: difference to previous block

DCT Coefficients

150	80	40	14	4	2	1	0
92	75	36	10	6	1	0	0
52	38	26	8	7	4	0	0
12	8	6	4	2	1	0	0
4	3	2	0	0	0	0	0
2	2	1	1	0	0	0	0
1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Quantized coefficients

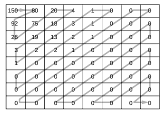
150	80	20	4	1	0	0	0
92	75	18	3	1	0	0	0
26	19	13	2	1	0	0	0
3	2	2	1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Quantization table

1	1	2	4	8	16	32	64
1	1	2	4	8	16	32	64
2	2	2	4	8	16	32	64
4	4	4	4	8	16	32	64
8	8	8	8	8	16	32	64
16	16	16	16	16	16	32	64
32	32	32	32	32	32	32	64
64	64	64	64	64	64	64	64

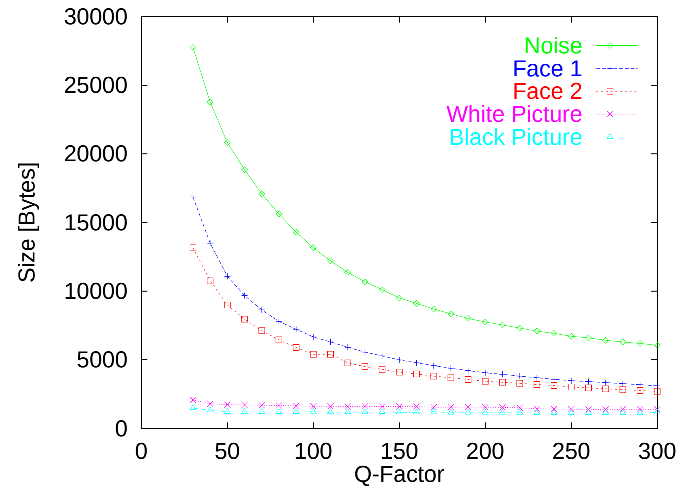
JPEG

- zig-zag scan:



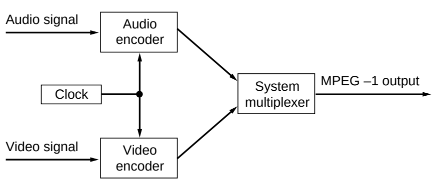
- run-length coding: group all zeros
- Huffmann coding
- See also `http://www.cs.columbia.edu/~hgs/video`

JPEG bandwidth



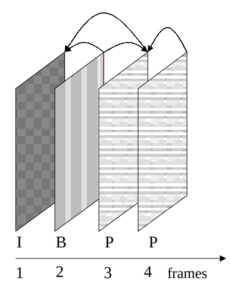
MPEG (Motion Picture Experts Group)

Audio and video:



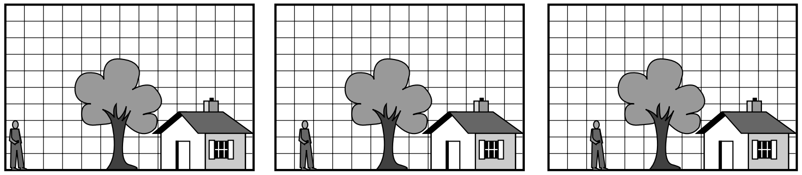
- ➠ JPEG + motion compensation ≈ H.261, MPEG video
- MPEG-1: 1.2 Mb/s fixed rate (CD ROM)
- MPEG-2: higher resolutions (HDTV), scaling
- image prediction: intra (I), forward (P), bidirectional (B) ➠ IBBPBBPBBPBBIBBPBBPB
- need I frames for error resiliency, joining movie in the middle

MPEG I, P, B



B frames need to wait for next frame.

MPEG: motion compensation



- transmit “motion vectors” to account for panning and zooming ➠ hard to find (must try lots), easy to decode

H.261 video codec

- ISDN ($n\times$ 64 kb/s) conferencing $\longrightarrow$ lower delay
- conditional replenishment: only transmit blocks that are different
- motion vectors for each 16x16 macroblock: ± 15 integer pixels
- GOB: 11 macroblocks H, 3 macroblocks V, marked by *start code*

H.263 video codec

- sub-QCIF (128×96), 4CIF (704x576), 16CIF (1408x1152)
- motion prediction outside frame
- advanced prediction mode: 4 vectors for each 8x8 block
- advanced intra prediction – within picture
- slice-structured mode: non-overlapping rectangles
- scalability: temporal (B frames), SNR, spatial

HDTV

- subset of MPEG-2 video compression, Dolby AC-3 audio compression
- vestigial sideband modulation (8-VSB) of 19 Mb/s or 16-VSB for two channels in CATV
- formats:

1280 x 720	24, 30, 60 Hz progressive scan
1920 x 1080	24, 30 progressive, 60 Hz interlaced

- MPEG-2 transport stream: fixed-length 188-byte packets (4x47 ATM cells)
- one channel = one or more programs

Multiplexing

Pack multiple streams into a single lower layer

- IETF: MIME, RTP (later)
- ITU: H.221 (synchronous, 80x8)
- MPEG: elementary, transport, program
- file formats: AVI, QuickTime

Characteristics of digital audio and video

	audio	video
rate	5.3…64…1500 kb/s	0.2…1.5…19 Mb/s
loss tolerance	$\leq 5\%$	10^{-5}…10%
packet size	small	large
traffic	constant + silences	variable bit rate

Audio traffic models

talkspurt: constant bit rate: one packet every 20…100 ms ➠ mean: 1.67 s

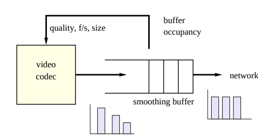
silence period: usually none (maybe transmit background noise value) ➠ 1.34 s

➠ for telephone conversation, both roughly exponentially distributed

- double talk for “hand-off”
- may vary between conversations… ➠ only in aggregate

Video traffic models

- easy case: fit into constant bit rate



- alternative: variable rate ➠ mux gain of $\approx 2 \ldots 6$
- short time scales: packets within slice or frame
- medium time scales: I, B, P packet pattern
- longer time scales: scene changes (every few seconds) ➠ higher rate
- looks similar at all time scales, long-term correlation, *heavy-tailed distribution* ➠ *self-similar*
- but: for short queues, long-term correlations don’t matter