Video in the Interface
Video: the BEST* modality

- As passive or active as needed
- Simple directional localization
- Line-of-sight supports see/be seen paradigm (within visible spectrum)

- Rich sensor

* According to a vision person.
Video Details

- Analog vs Digital
  - TV monitors
- Frame rates
  - 10fps for smoothness, 30fps common
- Resolution
  - Broadcast standards (NTSC, PAL, SECAM)
- Interlacing
Video Details (cont’d)

- Color scheme
  - RGB (familiar 3-byte rep)
  - YCC/YUV (Y=luminance, CC=chrominance/hue)

- Formats
  - Analog: composite, S-Video, component
DV standard

- 720 x 480
- 24-bit
- 29.97 fps
- .9 pixel aspect ratio (not square!)
- 44.1 kHz stereo audio
- 4:1:1 YCC/YUV
Storing Digital Video

- Single frame of uncompressed video
  - $720 \times 486 \times 3 = 1049760$ bytes ~ 1MB!
  - 1 second = 30MB
  - 1 Minute = 1.5GB

- Must compress
  - Normal Digital Video (DV) is 5:1
Compression

- Reduce resolution
- Reduce frame rate
- Reduce color information
  - Humans more sensitive to luminance than color
- Spatial (intra-frame) vs. Temporal (inter-frame)
- Codec handles compression and decompression of video
Wrapper vs. CODEC

- **Wrappers:**
  - tif, mov, qt, avi

- **CODECS:**
  - Sorenson, DV, Cinepak, MPEG II

- **CAUTION:** Lossy vs. Lossless
Using Video

- Much like other natural data types:

- As data
  - Playback/reminder: useful for humans

- Image understanding
  - Extracting features: interpreted by machines
Motivating Automated Capture

- Weiser’s vision: ubiquitous computing

  technology seamlessly integrated in the environment

  provides useful services to humans in their everyday activities

  Video (and other natural data types) are a part of the “seamless integration”
  component of this: make the machine adapt to the person, rather than the other way around
Motivation

- Scenarios in Weiser's Scientific America article:

  Sal doesn't remember Mary, but she does vaguely remember the meeting. She quickly starts a search for meetings in the past two weeks with more than 6 people not previously in meetings with her, and finds the one.

  Sal looks out her windows at her neighborhood. Sunlight and a fence are visible through one, and through others she sees electronic trails that have been kept for her of neighbors coming and going during the early morning.
Defining Capture & Access

- Recording of the many streams of information in a live experience and the development of interfaces to effectively integrate those streams for later review.

- Most often: in video-as-data mode
Capture & Access Applications

- Automated *capture* of live experiences for future *access*.

natural input

indexing

ubiquitous access
Capture & Access Applications

- Augmenting devices & environments with a memory
Capture & Access Design Space

• Benefits of automated capture and access have been explored in a number of domains, such as:
  • Classrooms: Lecture Browser,
  •        Authoring on the Fly
  • Meetings: Tivoli, Dynomite, NoteLook
  • Generalized experiences: Audio Notebook, Xcapture

• Application design space defined by:
  • Who: Users & roles
  • What: Experience & captured representation
  • When: Time scale
  • Where: Physical environments
  • How: Augmented devices & methods
Non-video example: PAL

- Personal Audio Loop
  - Instant review of buffered audio
  - relative temporal index
  - even/ReplayTV like jumps
  - cue/marker annotations
  - rapid skimming/playback
Example: eClass

Formerly known as Classroom 2000

electronic whiteboard
microphones

cameras
web surfing machine
extended whiteboard
Example: eClass

- synchronize streams
- web access
- Video segmented, indexed based on inputs from other modalities (voice comments, whiteboard marks) without need for video recognition
Example: eClass

- Segue to ubicomp infrastructures (next lecture)
- Separation of concerns made eClass evolvable for ~6 years
  - pre-production
  - capture
  - post-production
  - access
Building Capture & Access Applications

- What are requirements for any infrastructure to facilitate these kinds of applications?
Infrastructure aimed to facilitate the development of capture and access applications.
INCA architecture
INCA architecture

Capturer

tool for generating artifact(s) documenting history of what happened.
INCA architecture

Accessor

tool for reviewing captured artifact(s)
INCA architecture

Storer

tool for storing captured artifact(s)
INCA architecture

Transducer

tool for transforming captured artifact(s) between formats & types
INCA architecture

Basic functions are translated into executable forms

Modules can transparently communicate with each other
• same process for building applications whether it is a distributed system or a self-contained device
eClass in INCA
Information integration & synchronization

• Support for the integration of information is wrapped into access

• Supports a data-centric model of building capture & access applications:
  - captured data = content + attributes
  - Simple way to specify data being captured
  - Simple way to specify data to access
  - Simple way to associate different pieces of captured data
Additional features

- Additional support is provided to protect privacy concerns
  - Various stakeholders can observe and control the run-time state of an application
Interfaces based on Video Recognition

- Generally, has the same problems as using other forms of natural input for recognition
  - Errors, error correction
- Other problems, more-or-less unique to video
  - Unintended input
  - How to give feedback in the same modality?
    - Speech input, speech feedback
    - Video input.... what is equivalent of video feedback?
Image Analysis

- Thresholds
- Statistics
- Pyramids
- Morphology
- Distance transform
- Flood fill
- Feature detection
- Contours retrieving
Recognizing Video Input: AKA computer vision

- Often leads to feature-based recognizers
  - Similar to those described for handwriting analysis

- Some work recently on how to make these easier to construct:
  - “Image Processing with Crayons” -- Fails & Olsen. CHI 2003
  - Take sample video images
  - “Color” over them to indicate positive and negative samples
  - System generates classifier based on most salient features
  - Easy to integrate into other applications without ML programming
“Image processing with crayons”

- Addresses problems with how developers create machine learning-centric interfaces without having to know much about machine learning
  - Wide range of sample features that are useful for vision built into the system
  - Features: computed characteristics of input that can be compared to some template using a distance function
  - Most ML algorithms assume that substantial time will go into feature selection during development/testing
  - Implementing features can be difficult (complex convolution kernels, for example)
- “Coloring” successive samples tells the system how to pick features that provide the highest level of disambiguation
  - Generates classifier automatically based on training data
  - You can run classifier on additional data to see how it performs, mark up areas that it gets right/wrong
Other Recognition-Based Interfaces

- DigitalDesk
  - Recall from movie day
  - Two desks:
    - Paper-pushing
    - Pixel-pushing
  - Applications:
    - Calculator
    - Paper Paint
    - etc.
- Very simple “recognition”
  - Background subtraction from projected image to find occluded areas
  - Assume finger shape, tap sound used to trigger action
  - Other approaches: offset camera, look at when object and shadow collide
Other Recognition-Based Interfaces

- **ScanScribe**
  - Saund, UIST'03. Interaction with images drawn on a whiteboard through video
  - Available for free download from PARC:
  - (Not just useful for video... perceptually-supported image recognition)

- **ZombieBoard**
  - Saund
  - Combines video image as data with video image as command
  - Whiteboard capture tool
Support for Video in the UI

- Java Media Framework (JMF)
  - Handling raw video
- Vision
  - VIPER Toolkit (Maryland)
  - http://viper-toolkit.sourceforge.net/
  - Intel's OpenCV
  - http://sourceforge.net/projects/opencvlibrary/