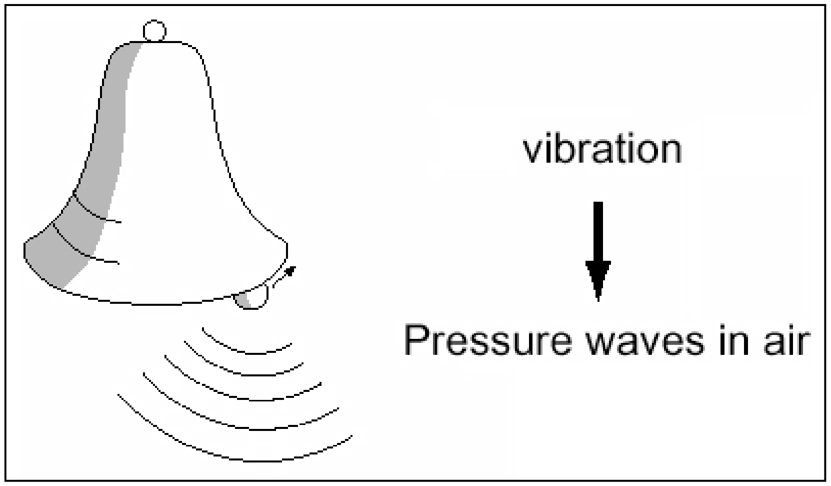
Sound and Non-Speech Interfaces: Going Beyond Conventional GUIs

Audio Basics

How sound is created

- Sound is created when air is disturbed (usually by vibrating objects) causing ripples of varying air pressure propagated by the collision of air molecules

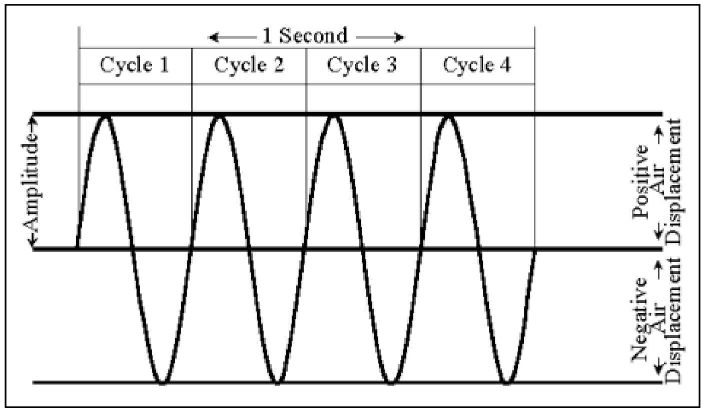


Why Use Audio?

- Good support for off-the-desktop interaction
  - Hands-free (potentially)
  - Display not necessary
  - Effective at a (short) distance
- Can add another information channel over visual presentation

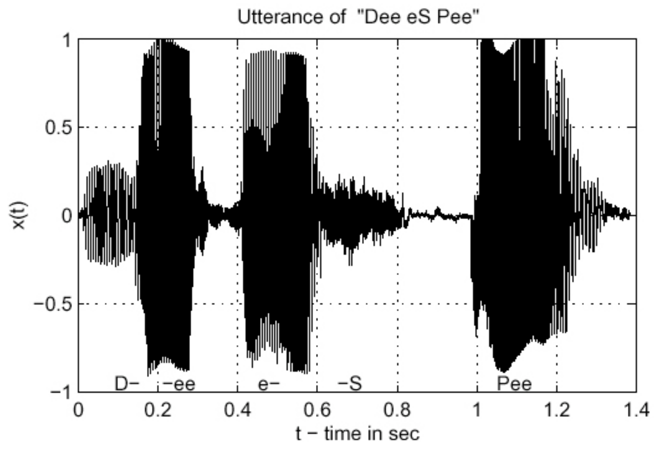
How Sound is Perceived

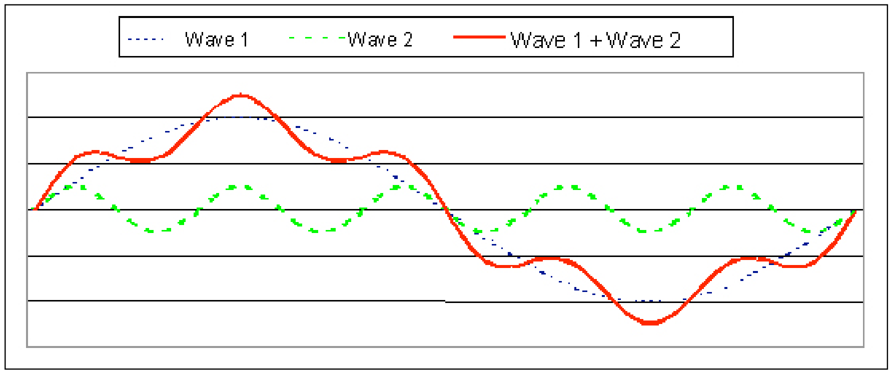
- Characteristics of physical phenomenon (the sound wave):
  - Amplitude
  - Frequency
- How we perceive those:
  - Volume
  - Pitch



Complex Sounds

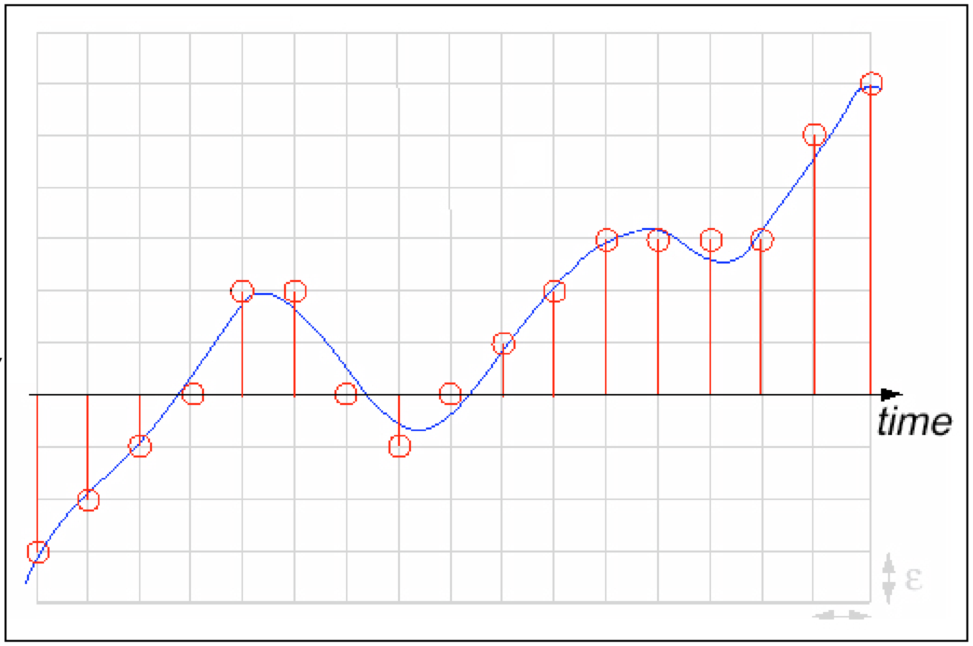
- Most natural sounds are more complex than simple sine waves
    - Can be modeled as sums of more simple waveforms; or, put another way:
    - More simple waveforms mix together to form complex sounds





Sampling Audio

- Sampling rate affects accurate representation of sound wave
- Nyquist sampling theorem
- Must sample at 2x the maximum possible frequency to accurately record it
- E.g., 44,100 Hz sampling rate (CD quality) can capture frequencies up to 22,050 Hz



Additional Properties of Audio that can be Exploited to Good Effect

- Sound localization
- Auditory illusions

Sound Localization

- We perceive the location of where a sound originates from by using a number of cues
  - Inter-aural time delay: the difference between when the sound strikes left versus right ears
  - Perhaps most important: *head-related transfer function*: how the sound is modified as it enters our ear canals
- We can take a normal sound and *process it* to recreate these effects
  - Calculate and add precise delay between left and right channels
  - Apply a filter in realtime to simulate HRTF
  - Requires ability to pipe different channels to left and right ears
- Problematic: each person's HRTF is slightly different
  - Because of external ear shape
- Still, can do a reasonably good job
- Generally need head tracking to keep apparent position fixed as head moves

Auditory Illusions

- Example: Shepard Tone
  - Sound that appears to move continuously up or down in pitch, yet which ultimately grows no higher or lower
  - Identified by Roger Shepard at Bell Labs (1960's)
- Useful for feedback where you have no bounded valuator?

Speech versus non-speech audio

- Speech *is* just audio; why consider them separately?
    - Uses in interfaces are actually vastly different (more on this later)
    - Actually processed by different parts of the brain
- Understanding the physical properties of audio, you can create new interaction techniques
    - Example: "cocktail party effect" -- being able to selectively attend to one speaker in a crowded room
    - Requires good localization in order to work
- In this lecture, we're focusing largely on non-speech audio

Using Audio in Interfaces

- That's all fine...
- ... but what special opportunities/challenges does audio present in an interface?

Changing the assumptions

- What happens when we step outside the conventional GUI / desktop / widgets framework?
  - Topic of lots of current research
  - Lots of open issues
- But, a lot of what we have seen is implicitly tied to GUI concepts

Example: "Interactive TV"

- WebTV and friends
- Idea is now mostly dead, but was attempt to add a return channel on cable and allow the user to provide some input
- Basic interaction, though, is similar for Tivo and other "living room interfaces"
- Is this "just another GUI?" Why or why not?

Not just another GUI because...

- Why?

Not just another GUI because...

- Remote control is the input device
  - Not a (decent) pointing device!
  - (Despite having many dimensions of input--potentially one for each button)
- Context (& content) is different
  - "Couch potato" mode
    - only a few alternatives at a time
    - simple actions
    - the "ten foot" interface -- no fine detail (not that you have the resolution anyway)

  ➡ Convenient to move in big chunks

Preview: Leads to a navigational approach

Have a current object

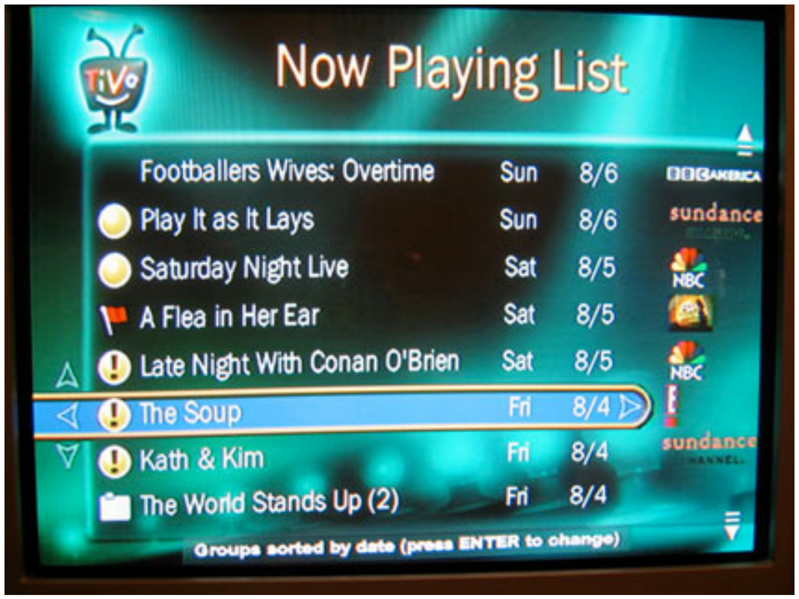
Act only at current object

- Typically small number of things that can be done at the object
  - Often just one

Move between current objects

Example: Tivo

- UP/DOWN
  - Moves between programs
- LEFT/RIGHT
  - Moves to menus/submenus
- At each item, there are a small, fixed set of things you can do:
  - SELECT it
  - DELETE it
  - ... maybe a few others depending on context



Now Playing List

Footballers Wives: Overtime	Sun	8/6
Play It as It Lays	Sun	8/6
Saturday Night Live	Sat	8/5
A Flea in Her Ear	Sat	8/5
Late Night With Conan O'Brien	Sat	8/5
The Soup	Fri	8/4
Kath & Kim	Fri	8/4
The World Stands Up (2)	Fri	8/4

Groups sorted by date (press ENTER to change)

Generalizing: Non-pointing input

- In general a lot of techniques from GUIs rely on pointing
  - Example: a lot of input delivery
- What happens when we don't have a pointing device, or we don't have anything to point to?
  - Extreme example: Audio only

The Mercator System

http://www.acm.org/pubs/citations/proceedings/uist/142621/p61-mynatt/

- Designed to support blind users of GUIs
  - GUIs have been big advance for most
  - Disaster for blind users
- Same techniques useful for e.g., cell phone access to desktop
  - Converting GUI to audio

Challenge: Translate from visual into audio

- Overall a very difficult task
- Need translation on both input and output

Output translation

- Need to portray information in audio instead of graphics (hard)
  - Not a persistent medium
    - Much higher memory load
  - Sequential medium
    - Can’t randomly access
  - Not as rich (high bandwidth) as visual
  - Can only portray 2-3 things at once
    - One at a time much better

Mercator solution

- Go to navigational strategy
  - only "at" one place at a time
  - only portray one thing at a time
- But how to portray things?
  - Extract and speak any text
  - Audio icons to represent object types

Audio icons

- Sound that identifies object
  - e.g. buttons have characteristic identifying sound
- Modified to portray additional information
  - “Filtears” manipulate the base sound

Filtear examples

- Animation
  - Accentuate frequency variations
  - Makes sound "livelier"
  - Used for "selected"
- Muffled
  - Low pass filter
  - Produces "duller" sound
  - Used for "disabled"

Filtear examples

- Inflection
  - Raise pitch at end
  - Suggests "more" -- like questions in English
  - Used for "has sub-menus"
- Frequency
  - map relative location (e.g., in menu) to change in pitch (high at top, etc.)

Filtear examples

- Frequency + Reverberation
  - Map size (e.g., of container) to pitch (big = low) and reverb (big = lots)

- These are all applied "over the top of" the base audio icon
- Can't apply many at same time

Mapping visual output to audio

- Audio icon design is not easy
- But once designed, translation from graphical is relatively straight forward
  - e.g. at button:
    play button icon, speak textual label
- Mercator uses rules to control
  - “when you see this, do that”

Also need to translate input

- Not explicit, but input domain also limited
  - Nothing to point at (can't see it)!
  - Pointing device makes no sense
- Again, pushes towards navigation approach
  - limited actions (move, act on current)
  - easily mapped to buttons

Navigation

- What are we navigating?
  - Don't want to navigate the screen
    - very hard (useless?) w/o seeing it
  - Navigate the conceptual structure of the interface
    - How is it structured (at UI level)
    - What it is (at interactor level)

Navigation

- But, don't have a representation of the conceptual structure to navigate
  - Closest thing: interactor tree
  - Needs a little "tweaking"
- ➡ Navigate transformed version of interactor tree

Transformed tree

- Remove purely visual elements
  - separators and "decoration"
- Compress some aggregates into one object
  - e.g. message box with OK button
- Expand some objects into parts
  - e.g. menu into individual items that can be traversed

Traversing transformed tree

- Don’t need to actually build transformed tree
  - Keep cursor in real interactor tree
  - Translate items (skip, etc.) on-the-fly during traversal
- Traversal controlled with keys
  - up, first-child, next-sibling, prev-sibling, top

Traversing transformed tree

- Current object tells what output to create & where to send input
  - upon arrival: play audio icon + text
  - can do special purpose rules
- Have key for "do action"
  - action specific to kind of interactor
  - for scrollbar (only) need two keys

Other interface details

- Also have keys for things like
  - “repeat current”
  - “play the path from the root”
- Special mechanisms for handling dialog box
  - have to go to another point in tree and return
  - provide special feedback

Mercator actually has to work a bit harder than I have described

- X-windows toolkits don't give access to the interactor tree!
- Only have a few query functions + listening to the "wire protocol"
  - protocol is low level
    - drawing, events, window actions

Mercator actually has to work a bit harder than I have described

- Interpose between client and server
  - query functions get most of structure of interactor tree
  - reconstruct details from drawing commands
  - catch (& modify) events

Why not just use a toolkit that gives access to the tree?

- To be really useful Mercator needed to work on existing "off the shelf" applications without modification (not even recompile or relink)
  - critical property for blind users

Audio Input

- Most audio input has focused on *speech input*
- However, some work on non-speech input

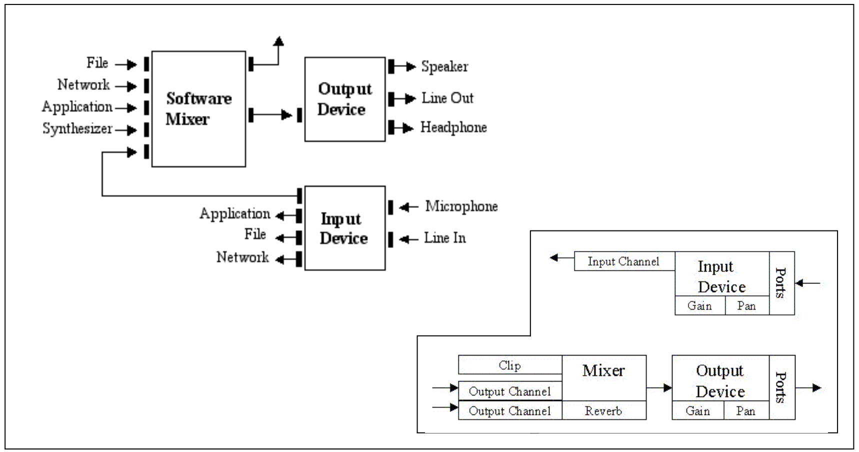
- Example:
  - "Voice as Sound: Using Non-verbal Voice Input for Interactive Control," Igarashi and Hughes, UIST 2001

Question for the class

- How would you do drag and drop in audio?

Resources

- JavaSound architecture
  - Insanely good (and under-utilized) architecture for doing powerful audio processing
  - Build on technology licensed from Beatnik, Inc.
  - Company started by Thomas Dolby (the "blinded me with science" guy)
- http://java.sun.com/products/java-media/sound/
- http://www.jsresources.org/



File
Network
Application
Synthesizer
Software Mixer
Output Device
Speaker
Line Out
Headphone
Application
File
Network
Input Device
Microphone
Line In
Input Channel
Input Device
Gain
Pan
Ports
Clip
Output Channel
Output Channel
Mixer
Reverb
Output Device
Gain
Pan
Ports