Touchless Interfaces

Managing sensor data

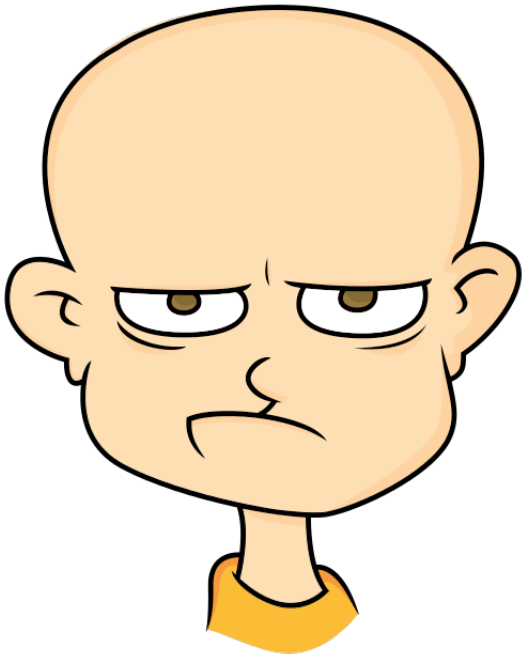
Design considerations

In-air and speech interfaces

Sources

- “Sensor and Recognition Based Input for Interaction”, Human-Computer Interaction Handbook (Wilson, 2012)
- Designing SpeechActs: Issues in Speech User Interfaces (Yankelovich et al., CHI 1995)
- “Touch versus In-Air Gestures”, Brave NUI World: Designing Natural User Interfaces for Touch and Gesture (Wigdor, 2011)

Today, we have a wide array of sensors available. We can use these to build novel applications, with innovative interaction.



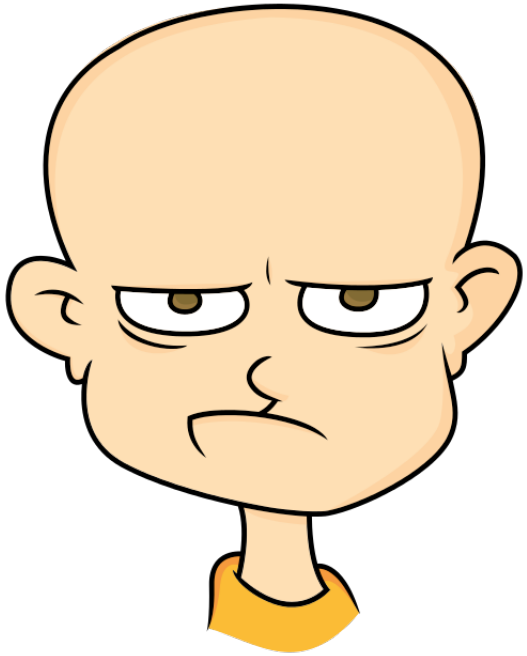
OpenClipArt.org

"Frustration Meter"

- microphone to detect users muttering and yelling at the computer
- pressure sensor to detect whether user is typing hard or squeezing the mouse
- webcam to detect frowning
- chair sensors to detect user agitation

Challenges with Sensors

The frustration meter is not easy to build:



OpenClipArt.org

- need to learn the mapping between the output of the sensors and the presumed state of frustration in the user – varies by user
- the way users expresses frustration is task and context dependent
- sensor output is noisy
- may need a combination of sensors
- our preconceived notions of frustration may not be what the sensors observed
- need to balance the cost of our system making mistakes and the benefit that the system provides

What might we want to sense?

Occupancy & Motion: senses motion or a person's presence

- infrared motion detectors, pressure mats, computer vision w. cameras

Range: calculate distance to a given object

- stereo computer vision system (triangulation), infrared cameras (time-of-flight), Polaroid ultrasound (time-delay)

Position: geo-location of the device

- GPS, motion capture system (with corresponding body model)

Movement and Orientation: sense spatial motion (e.g., translation, rotation)

- inertial sensors like gyroscopes, accelerometers

Gaze: determining where a person is looking (e.g. target on-screen)

- camera, eye-tracking systems

Speech: detect and process voice input or commands (e.g. Siri)

- array microphone combines audio from multiple sources for filtering

Brain-wave activity: low-fidelity input, potential for accessibility

- EEG (requires extensive training).

What can we do with this data?

Support in-air gestures: hand pose, spatial trajectory of hands or stylus

- tracking and movement devices (e.g. smartphone), hand-tracking systems (cameras, using markers etc.)

Support speech commands: specific words or phrases

- command or phrase recognition, data entry (e.g. Google Now, Siri)

Identify objects or people: object recognition, person recognition (face recognition)

- lots of options: on-body devices/wearables (RFID, BT), fingerprints, retina-scanning, heart-rate monitoring, EEG

Determine context: figure out the context of the user (e.g. in-car)

- environmental sensor that detect air temperature, lighting quality, air pressure; cameras that detect location or environment

Determine Affect: detect an emotion or subjective feeling in a person

- respiration, heart-rate monitors, blood-pressure sensors, EMG (muscle contractions)

Design Considerations

Computation cost

- Computer vision algorithms are still computationally intensive, and some analysis cannot easily be performed in real-time (e.g. head-tracking on a phone)
- Approaches may require aggregating data from multiple sensors.
- High-volume of continuous data!
- Sensor data is really, really noisy and requires work to cleanup

Traditional or non-traditional interfaces

- How do we integrate these sensors and this type of data into common, real-world applications?
- Sensors: suggest that apps are more data-driven than task-driven (recent approaches of speech and facial recognition)

Step 1: Pre-processing
- compress
- smooth
- thresholding (discretize continuous quantity)
- downsample
- handle latency

Step 2: Feature Selection
- e.g., face detection
  - distances between eye, nose and mouth
  - eye-blinking patterns

Step 3: Classification
- determining which class a given observation belongs to
- e.g., recognize which gesture is performed

Example: "Computer, Turn on the Lights"

(Brumitt et al., 2000) Users try controlling the light using these options:

- a traditional GUI list box
- graphical touch screen display depicting a plan view of the room with lights
- two speech only-based systems
- a speech and gesture-based system

Users prefer to use speech to control the lights, but the vocabulary used to indicate which light to control is highly unpredictable.

Example: “Computer, Turn on the Lights”

Insight: Users look at the light that they are controlling while speaking.

XWand system is a hand-held device that can be used to select objets in the room by pointing, and a speech recognition system for simple command and control grammar. (turn on, turn off, roll to adjust volume)

http://www.youtube.com/watch?v=bf3mQRmzA4k#t=146

Wilson, 2007

Key Challenge: Balancing Usage Patterns

Explicit Interaction
- user takes action and expects a timely responsive from the system
- e.g. Siri, Kinect-driven games

Implicit Interaction
- based on user's existing pattern of behaviour
  - e.g., frustration meter
  - e.g., think "personalized google search"
  - e.g., a smart home that observe an inhabitants daily pattern of coming and going to determine an optimal thermostat schedule

We can use sensor data to drive interaction in both scenarios.

Key Challenge: Errors is a Balancing Act

False Positive Error

- "positive" == system recognizes a gesture
  - i.e. user does not intend to perform a gesture but the system recognizes one
- makes the system seem erratic or overly sensitive.
- triggered by high sensitivity

False Negative Error

- "negative" == system fails to recognize
  - i.e. user believes he/she has performed a gesture but the system does not recognize it
- feels as-if doing something wrong
- makes system seem unresponsive
- triggered by low sensitivity

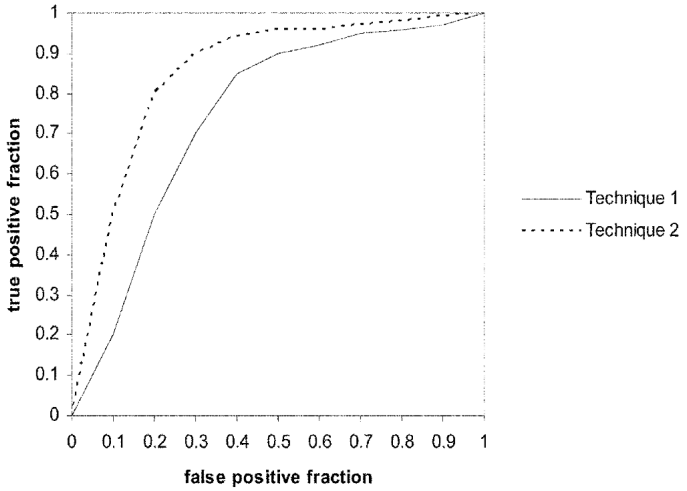


FIGURE 10.1. ROC curves illustrate the trade-off between the rate of true positives and false positives, and can be useful in comparing recognition techniques. Here we see that for a given tolerable rate of false positives, Technique 2 yields better recognition performance than Technique 1.

Wilson, 2007

Key Challenge: Errors is a Balancing Act

Users need to feel a sense of control

- may feel unable to correct a mistake made by the system, or unable to exert more explicit control in an exception
- DWIM (i.e. users want a system to “do what I mean”)

Users may be very intolerant of errors

- in speech recognition, only users that are unable to use a regular keyboard may accept a dictation system that fails 3 times out of 100 words

Strategies

- graceful degradation, i.e., return a results similar to desired results
- avoid false positive by seeking deliberate confirmation from users
- give control: allow users to query the system for why it took a given action, fix errors, and revert to manual mode.

Example 1: In-Air Gestures

Minority Report

The 2002 film Minority Report shows:

- Video conferencing (before Skype and similar)
- Gesture-based User Interfaces
- Predictive crime fighting (pre-crime)
  - beyond the scope of this course!

https://www.youtube.com/watch?v=PJqbivkm0Ms

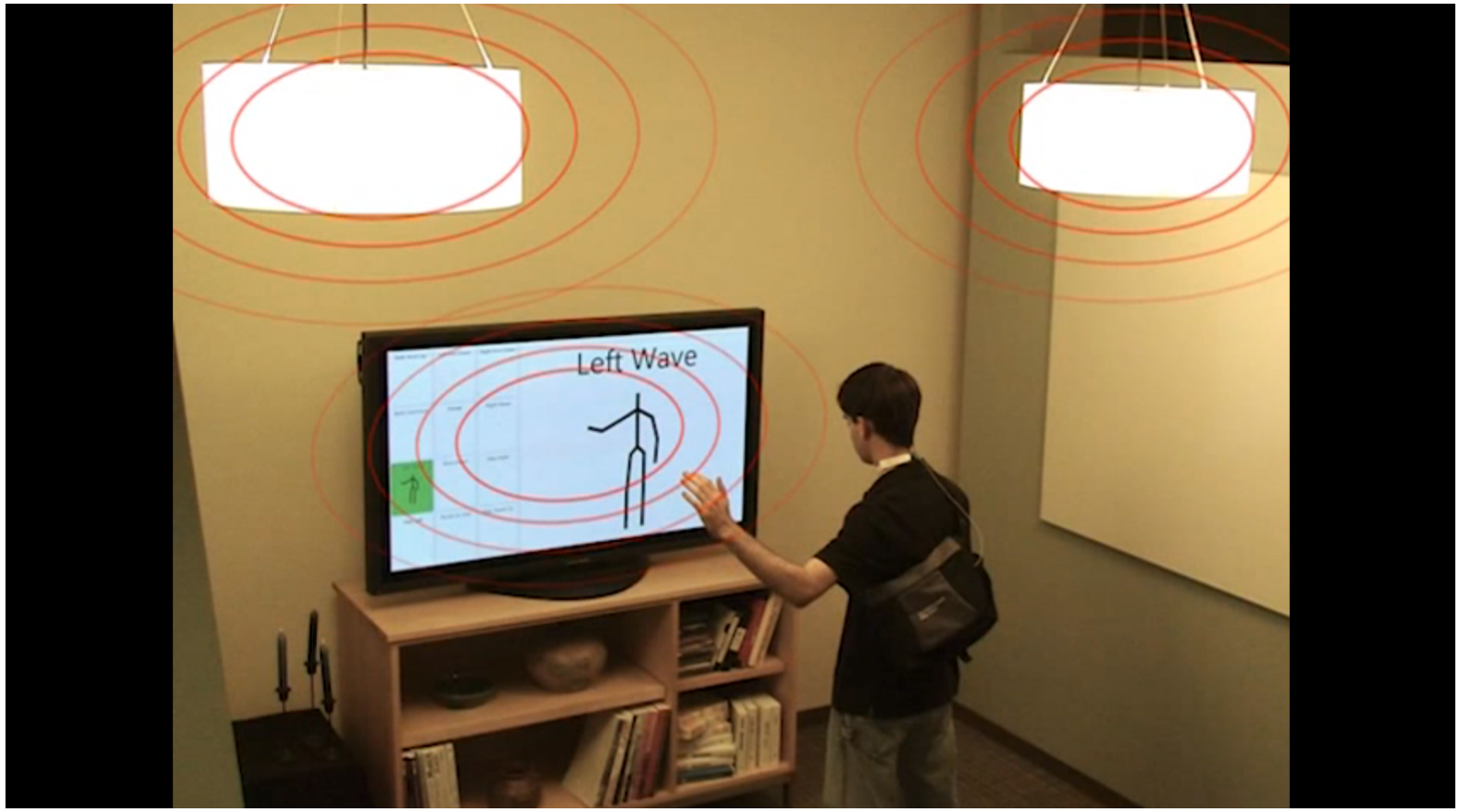
Example: Pointing to the future of UI

The real-life technology that inspired the movie

https://vimeo.com/49216050

http://www.ted.com/talks/john_underkoffler_drive_3d_data_with_a_gesture
(5:30-9:15)

Example: Humantenna



http://www.youtube.com/watch?v=em-nvzxzC68

http://www.youtube.com/watch?v=7IRnm2oFGdc

Benefits

In-air gestures offer benefits over “traditional” interaction:

1. No need to physically touch anything
   - Ideal for situations where it would be impractical to have a physical input device
   - e.g. during surgery, or public spaces where you want to prohibit interaction
2. More expressive than traditional keyboard + mouse
   - Potential for “Direct Manipulation in 3D space”.
   - Pairs with VR, AR.

Problem: "Live Mic"

In-Air Gestures suffer from the "Live Mic" problem.

https://en.wikipedia.org/wiki/We_begin_bombing_in_five_minutes

> "My fellow Americans, I'm pleased to tell you I just signed legislation which outlaws Russia forever. The bombing begins in five minutes."
>
> — Ronald Reagan, 1984

Problem: Limited Input States

These mechanisms have limited input channels.

- Mouse is a three-state input device.
- Touch Interface is a two-state input device.
- Touchless Interface is a one-state input device.
  - i.e. an in-air gesture system is always on, and listening for input.

Consider a user who needs to sneeze, scratch her head, stretch, gesture to another person in the room - what would this mean for the three input devices?

Solutions

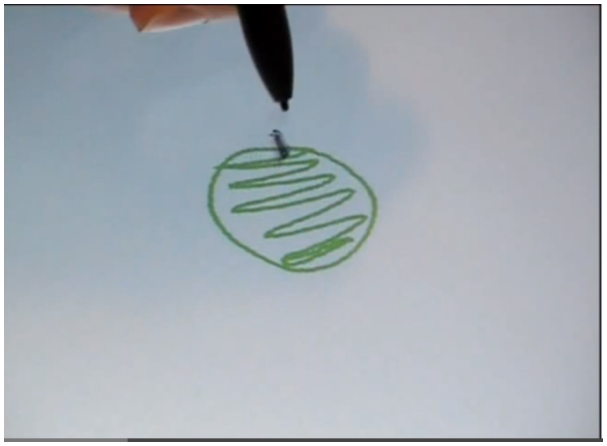
1. Reserved actions
2. Delimiters
3. Multi-modal input

Solution 1: Reserved Actions

Take a particular set of gestural actions and reserve them so that those actions are always used for navigation or commands.

Wigdor, 2011

pigtail gesture



http://www.youtube.com/watch?v=WPbiPn1b1zQ

hover widget

Question: for hover widget, which type of error is more common (false negative or false positive) and why?

- user might accidentally lift their pen and move it
- user might exit the zone without realizing it *

Solution 2: Delimiters

A *delimiter* is an indicator that you want to start or stop the recognition of a gesture

- e.g. Ruiz and Li, “DoubleFlip: A Motion Gesture Delimiter for Mobile Interaction (used in Moto X to activate the camera)
- e.g. Widgor: to engage, user pushes past an invisible plane in front of them

Advantages?

- provides an a priori indicator to the recognizer that the action to follow is intended to be treated as a gesture (or not).
- enables the complete gestural space to be used

Disadvantages?

- where should this invisible plane be? This may be different for different users, or different for the same user over time

Solution 2: Delimiter

Alternative: to engage, user pinches finger and thumb together

Advantages:

- less likely to be subject to false positive errors (Why?)
  - –the action is unlikely to occur outside the context of the app
- less likely to be subject to false negative errors (Why?)
  - –the action is easy to recognize and differentiate from other actions

Disadvantages:

- cannot use pinching in your gestural set

Solution 3: Multi-Modal Input

iPhone is a touch input device, so … Why does have a button?

The key problem is that users need to be able to exit their application and return to the home screen in a reliable way.

What's the alternative?

- a reserved action for exit

The multi-modal solution enables touch input to be *always* be sent to the application.

Hardware buttons control system functions: universal parameters (e.g., volume, mute) and navigation (e.g., Home, use Siri).

Advantage over Reserved Action or Clutch:

- does not reduce the vocabulary of the primary modality

Some other examples

- CTRL-drag becomes copy instead of move
- speech + gesture (e.g., the “put that there” system)

Put That There (MIT, 1980)
http://www.youtube.com/watch?v=-bFBr11Vq2s

Feedback is Useful

Provide feedback on all recognition results and the nature of failure so that users can modify the behaviour to meet the system's expectation.

Wilson, 2007

EyeToy

- image of player on screen
- by watching themselves on screen, players are able interact with the onscreen elements without relying on the sophisticated (but more failure prone and computationally intensive) hand-tracking algorithms
- this feedback also keeps player staying in camera's field of view

http://blog.us.playstation.com/2010/11/03/eyetoy-innovation-and-beyond/

Feedback is Useful

Provide feedback on all recognition results and the nature of failure so that users can modify the behaviour to meet the system's expectation.

Wilson, 2007

Jump system (Michael Terry)

- image of hands on screen
- by watching themselves on screen, users monitor system.
- Overlay indicates successful recognition of fiducials, clearly demonstrates that successful recognition occurs.
- this feedback also keeps player staying in camera's field of view

Figure 3. The output display. The reflection window can be seen in the lower-left hand corner

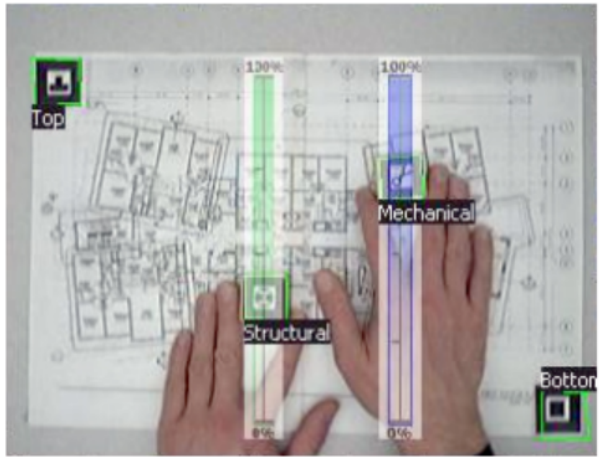
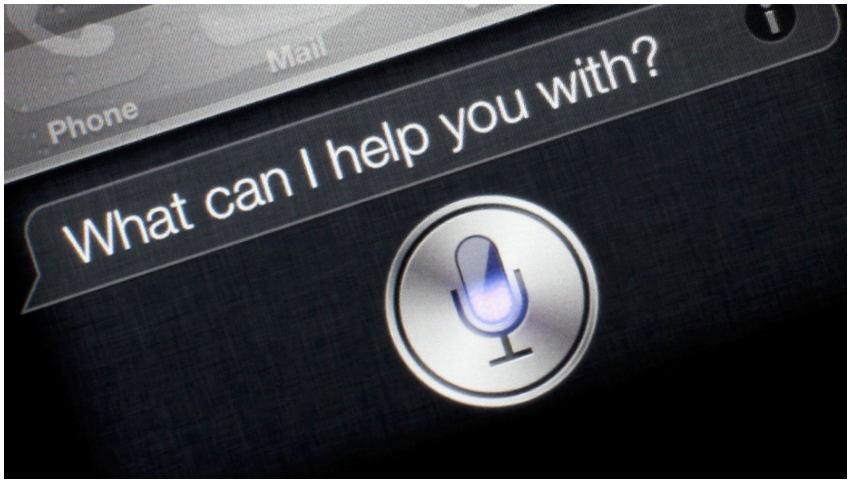
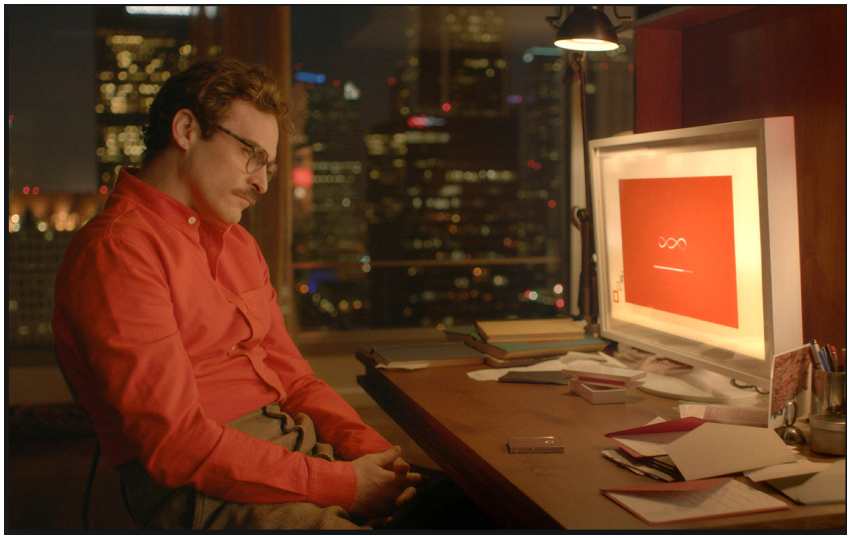


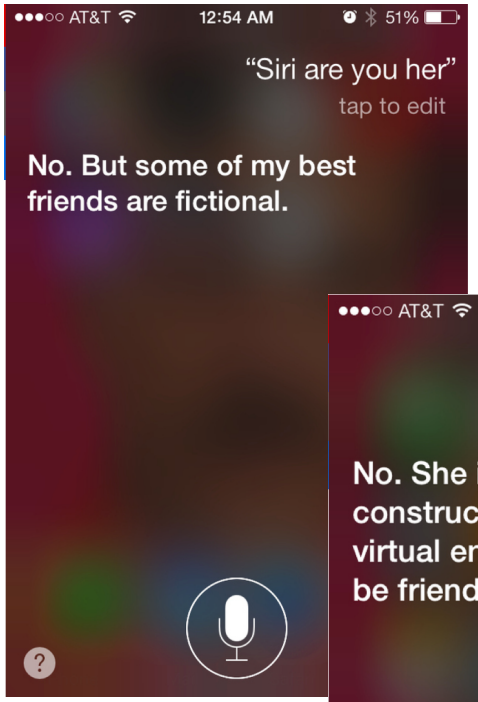
Figure 4. Close-up of the reflection window showing virtual sliders superimposed over two filter tokens

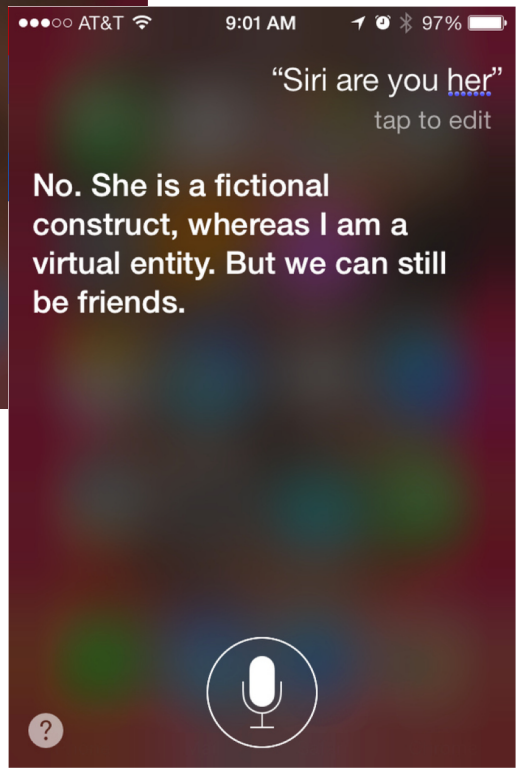
Example 2: Speech Interfaces

Speech Interfaces (SUI)









SpeechActs - early example

SpeechActs provides interfaces to a number of applications, including email, calendar, weather and stock quotes.

https://www.youtube.com/watch?v=OzNRsGaXyUA

- To make interaction feel conversational, the system avoids explicitly prompting users for input whenever possible
  - e.g., use a prompt tone
- Found that recognition rates were a poor indicator of satisfaction

> "Users bring many and varying expectations to a conversation, and their satisfaction will depend on how well the system fulfills those expectations"
>
> Yankelovich, 1995

Discourse segment pop-up / navigation / context

- Challenges in navigating through sequence of system commands
  - –e.g. How do you complete a task, or signal that you're done dictating commands?
  - –e.g. How do you correct a mistake?
  - –e.g. After replying to a message, how do you navigate back and return to reading messages?

Users want to use their "human" language

- Manager: Next Monday — Can you get into John's calendar?
- Assistant: Gosh, I don't think I can get into his calendar.
  - –no mention of "browse", "user ID"; use of "relative date"

System prompting

- When should the system ask for confirmation?
  - –What if the user does not confirm properly?
  - –How do you "pop up a dialog" in speech?

GUI to SUI: Recognition Errors

- Recognition errors are frequent (e.g., user speaks before system is ready, background noise, words spoken by passerby, out-of-vocabulary utterances)
  - –Users often say something once and have it recognized, then say it again and have it mis-recognized
  - –Lack of predictability means that users cannot have a useful conceptual model of how the system works

Types of Errors

- Rejection error: system has no hypothesis about what user has said (e.g. “brick wall effect”: keep saying “I don’t understand”)
  - –progressive assistance
    - What did you say? -> Sorry? -> Sorry. Please rephrase.
- Substitution error: mistaking utterances (e.g., “send a message” interpreted as “seventh message")
- Insertion error: recognize noise as a legal utterance

GUI to SUI: Other Challenges

Lack of Visual Feedback

- long pauses in conversations are perceived as embarrassing, so users feel a need respond quickly
- lack of think time can lead to false starts or “ums” and “ahs”
- less information is transmitted to users at a time

Speech is easy to produce, hard to absorb

- we have short term memory
- eliminate repetitive word

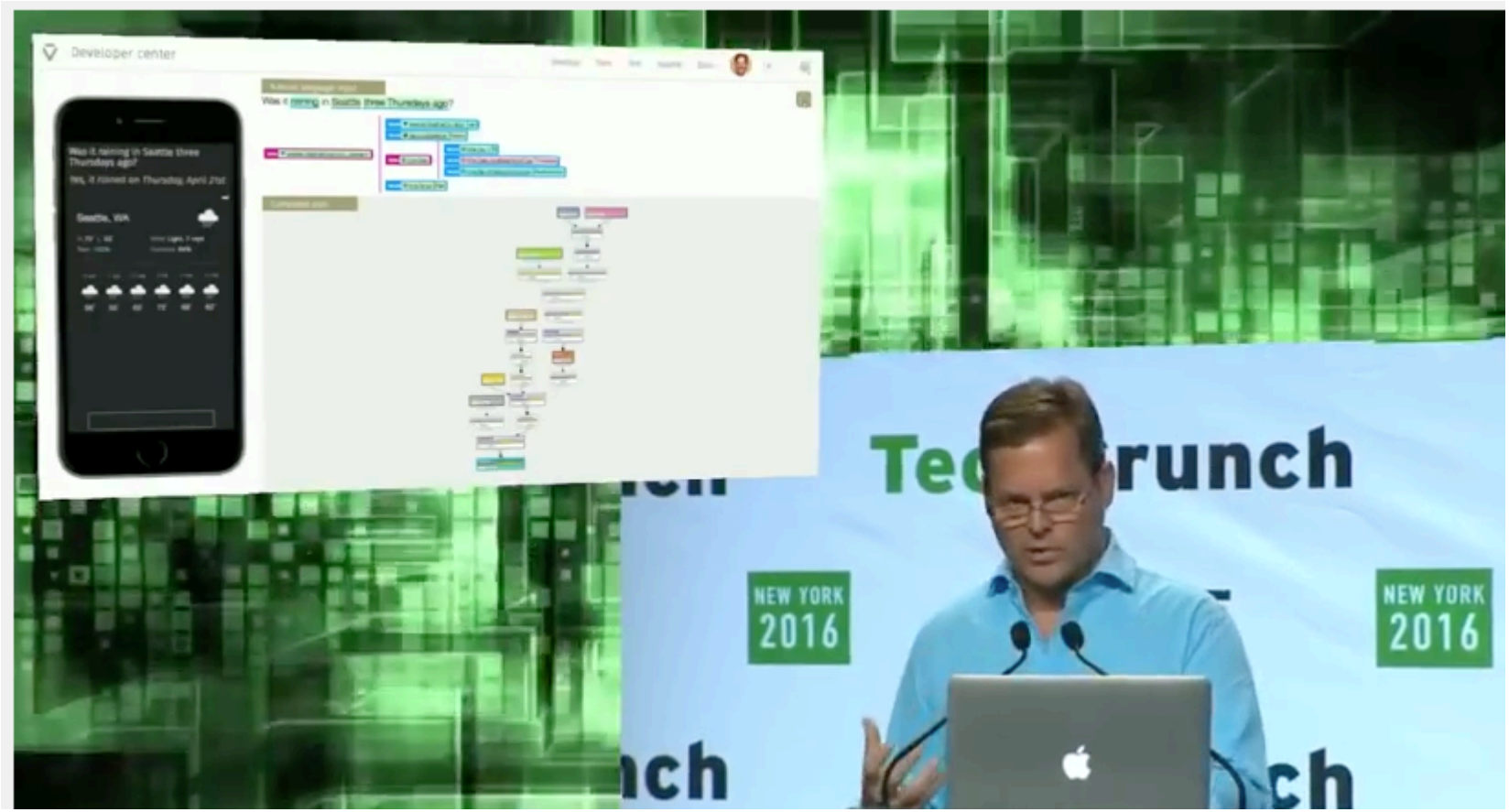
```
Currently, Sun is trading at 32, up 1/2 since yesterday
           SGI is                 23, down 1/4
           IBM is                 69, up 1/8
```

Ambiguous Silence

- is speech recognizer working on something? did it not hear?
- users need immediate feedback if response is slow

All of these issues make it challenging to use speech as our “one and only” input mechanism for complex interactions.

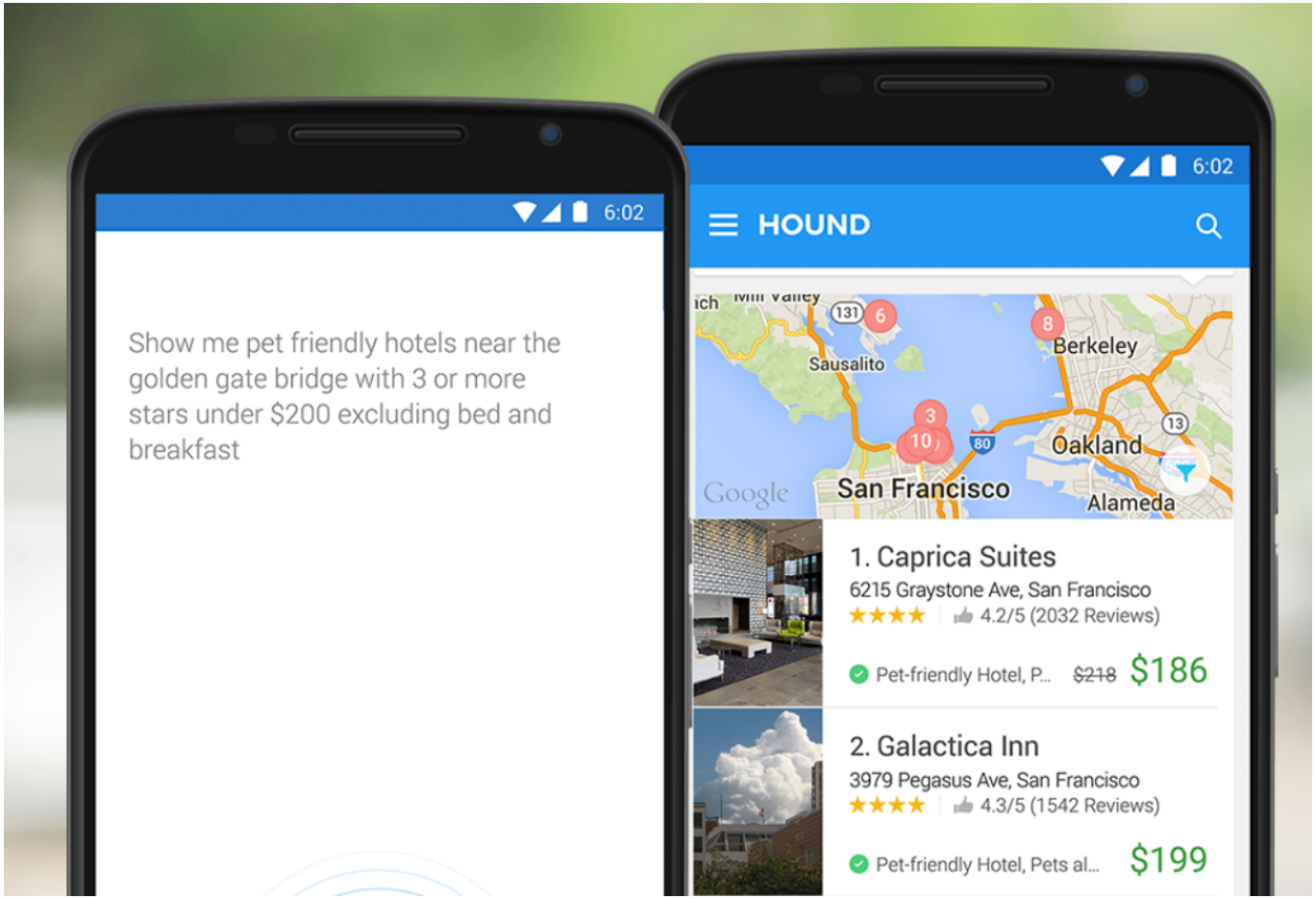
Viv – modern example



Speech recognition system, from creators of Siri

https://www.youtube.com/watch?v=Ml07aeZqeco

Hound - modern example



Hound - available on iOS and Android.

http://www.soundhound.com/hound
https://www.youtube.com/watch?v=M1ONXea0mXg

Summary

- Touchless interfaces are one-state input devices!
- The mapping of sensor information to commands is computationally expensive, ambiguous and error prone.
- In designing touchless interfaces, a key objective is to ensure that the system fails gracefully, and errors are managed properly.
- Consider touchless as one modality, that can complement other input modalities.