UUIs – Ubiquitous User Interfaces

Alexander Nelson
April 16th, 2018

University of Arkansas - Department of Computer Science and Computer Engineering
“As more and more computation is woven into the fabric of our lives, our interaction with such ubicomp systems cannot be the focus of our attention. As noted previously, if every physical object in our home demanded attention in terms of alerts, updates, and configurations in the way our current personal computers do, we would become quickly overwhelmed. Indeed, it has been noted that the benefits of inexpensive Ubiquitous Computing may be overwhelmed by its personal cost (Heiner et al., 1999). Instead, the computational and activity support offered by ubicomp systems must reside in the periphery of our attention and should remain unnoticed until required.”

Aaron Quigley – Ubiquitous Computing Fundamentals Chapter 6
Traditional User Interfaces

Traditional Computing Interface

- Monitor (Visual Output)
- Keyboard (Text Input)
- Mouse & Cursor (View Context)
- Speaker (Audio Output)
Two user-in-the-loop interaction styles:

- Notification – Ubicomp system obtains user’s attention
- Modification – User modifies context of the ubicomp system

User out-of-the-loop interaction?
System makes intercessions for user – Does not necessitate UI or UI may be undesirable
(e.g. Nest changing temperature)
Modification
Modification – Interaction pattern where user intentionally modifies state of ubicomp system

Traditional GUI analog – Mouse/Keyboard Input
Input technologies for ubicomp systems are many and varied. Example inputs:

- Infrared Remote Controller
- Wall-mounted switch panels
- Web-based dashboards
- Touchscreen
- Game Console Controllers

Can cause confusion in how to interact with systems.
Usability Overlap

Figure 6.3 – “Controller Hell”

This lecture is developed from text and images provided in our optional textbook – Ubiquitous Computing Fundamentals, Chapter 6, by Aaron Quigley
Ubiquitous Interfaces often make use of sensed contexts to facilitate/aid interaction

Common sensor-based inputs:

- Physiological Measurement
- Environmental Measurement
- Location
- Identity
- Audio/Video
- Gesture
- Touch
How does the system designer make use of these sensed values for input?

So far we covered sensors in terms of:

- Context-Awareness
- WBANs/MBANs

These interaction patterns are typically more stochastic and inferred/implicit rather than explicit and intentional.
Gesture – Movement of part of the body to express meaning
Human-to-Human Gesture – Aid/Facilitate ideas in lieu/support of speech (e.g. Sign Language)
Gesture Recognition – Interpreting human gestures through computation

Gestures can be interpreted as intentional ambient human-computer interaction
Example: Handshake

Haddock et al. 2009 – Handshake detection for social network development
Pen/Stylus Gestures

Mobile computers (e.g. Palm Pilot and others) gave rise to pen/stylus based gestures
Inertial Sensors (e.g. Accelerometer/Gyroscope) can establish 6 degrees of freedom for gestures
Move Up/Down, Left/Right, In/Out
Rotate Yaw, Pitch, Roll
Smartphone Gestures

Touchscreen + Inertial Sensor Input
Vision-based Gestures

Kinect – Vision + Depth
Physiological Sensor-based Gestures

Physiological Sensors as gesture input:

- Electromyography (EMG) – Myo Band
- Electrooculography (EOG) – Eye Motions
- Electroencephalography (EEG) – Brain-Computer Interface
Interpreting Gestures

Sensors provide a time-series set of data
Types and frequency depend on set of sensors

Example: Wii-Mote w/ Remote Plus
Sensors – Accelerometer, Gyroscope, Optical Sensor (for triangulation)
Single point in time defined by the 8-dimensional tuple:
\[ S_i = \langle A_x, A_y, A_z, \omega_x, \omega_y, \omega_z, P_x, P_y \rangle \]
Where \( A \) is acceleration, \( \omega \) is angular velocity, and \( P \) is the position of the pointer (can be null)
Interpreting Gestures

Gesture defined as the set of time-series points:
\[ G = \{S_0, S_1, S_2, \ldots, S_n\} \]

Gesture Classification can be continuous or discrete

Discrete – Pre-segmented gestures (i.e. Touch-based Smartphone gesture)
Continuous – Must be recognized during continuous sensor sampling

Discrete gestures are typically segmented through distinct end-points
“Pen-down” and “pen-up” – Distinct segmenting time points (from stylus being touched until removed)
For continuous Gesture Recognition, time series is often segmented for classification.

Example: Sliding Window

Some on-line recognition continuously update probabilities and do not need segmentation.
Common segmentation strategies:

- Sliding Window
- On-line gesture likelihood classifier
  Detect probability of any gesture occurring, based on feature extraction (reduces number of times classifier runs)
- Start-Stop gesture actions
Interpreting Gestures

Gestures match against an “alphabet” – The available gestures

Alphabets need to be accurate (both precision & recall)

*Alphabets need to be memorable*

---

*Figure 1: The Graffiti (top) and Unistrokes (bottom) gesture alphabets.*
How do you classify gestures to the “alphabet?”
Classifier – Determine gesture based on features of input data
Classifier can be programmer defined or from Machine Learning
Usability
Usability often defined by five quality components:\(^3\):

- Learnability – Training Time
- Efficiency – Execution Time
- Memorability – Retraining Time
- Errors – Precision/Recall
- Satisfaction – Pleasant to use (Qualitative)

\(^3\)https://www.nngroup.com/articles/usability-101-introduction-to-usability/
Usability for UUIs is slightly different
Quigley defines usability for UUIs in terms of:

- Conciseness – Few actions in brief time to achieve task
- Expressiveness – Consistency of use
- Ease – Learning curve for UUI
- Transparency – How well UUI conveys state information
- Discoverability – Can user make mental model of interface
- Invisibility – How well does UUI stay in the periphery
- Programmability – Can UUI be extended for other applications
Notification
Notification – Ubicomp system obtains user’s attention
Primarily in calm ubiquitous interfaces
i.e. A system that is in the periphery of attention that wants to provide information
Ambient Interfaces are intended to be “ignorable” / “glanceable”
Information that typically resides in the periphery of attention.
Periphery – based on visual analog where certain forms can be
detected outside line-of-sight.
Criteria for Ambient UIs

Quigley defines five characteristics of a Ubiquitous User Interface to be a tangible interface:

1. Provide a clear coupling of physical artifact to relevant and related computation, state, and information
2. Ensure contextually appropriate physical affordances for computational interaction
3. Ensure contextually sensitive coupling of physical artifact to intangible representation (audio/graphics)
4. Support “invisibility in action” (not literal invisibility) and natural behavior
5. Ensure a grounding of the UI interaction design in the fields of ethnography, industrial design, kinesthesiology, and architecture
Peripheral information can be conveyed through:

- Static Visual
- Dynamic Visual
- Sounds
- Touch/Tactile/Haptic
- Smell
- Temperature Changes

Based on the five traditional senses (i.e. Sight, Hearing, Smell, Touch, Taste(?))
Ambient Interface – Sight

Power Aware Cord – Gustafsson
Ambient Interface – Sight

Information Percolator – Heiner 1999
Ambient Interface – Sight

Ambient Umbrella

Top quality gust-buster umbrella canopy design

The magic is in the handle – it glows to indicate when rain or snow is forecast – so you remember to take it with you.
1997 Exploration of ambient information delivery through light, shadow, sound, airflow, water movement

Click photo to navigate to project page
Speech Recognition
Speech Recognition as UUI

Speech Recognition – Computational interpretation of human speech through audio input
Has emerged as a popular UUI in recent years
Speech Recognition History

Speech Recognition has a long history:

- 1952 – Bell Labs single digit recognition ("Eight")
- Late 1960s – Raj Reddy Continuous Speech Recognition
- Late 1960s – Dynamic Time Warping allows 200-word vocabulary
- 1970s – Hidden Markov Models used for speech recognition
- Mid 1980s – IBM produces Tangora w/ 20k word vocabulary
- 1990s – First commercial successful speech recognition technologies
Limitations

End-to-End Automatic Speech Recognition – Joint model combining pronunciation, acoustic, and language models

Benefits: HMM-based systems require an n-gram language model, taking several GB of memory

Siri/Google Assistant require network connection to cloud holding n-gram model