IMGD 3100 – Novel Interfaces for Interactive Environments: Interactivity

Robert W. Lindeman
Associate Professor
Interactive Media & Game Development
Human Interaction in Virtual Environments (HIVE) Lab
Department of Computer Science
Worcester Polytechnic Institute
gogo@wpi.edu
Introduction

☐ This course is about going beyond traditional interfaces
   ■ Keyboard, mouse, gamepad

☐ There are many ways of providing feedback to users, and many ways to gather input

☐ The key is to find the effective and efficient ones

☐ Depends on three main components
   ■ User
   ■ Task
   ■ Environment
What is Interaction?

- "The exchange of information between two or more active participants" (Noble)
- "An iterative process of listening, thinking, and speaking between two or more actors" (Crawford)
- HCI means that at least one of the participants is a system, and at least one is a person.
- As a designer, you are trying to understand what the user wants to do and how the system should respond to support this.
The feedback Loop

- Many systems have a "regulatory system" to maintain good operation
  - Sweating, breathing, balancing, blinking

- No user intervention for these to work
  - "Automatic" (actually, *nothing* is automatic)

- We are looking more at active interaction
  - Still need to provide feedback loops when appropriate
Levels of Interaction

- **Pacing**
  - User controls movement through pre-specified material

- **Reaction**
  - System reacts to user input
  - This gets you thinking about what the user might do, and how the system should react

- **Monitoring and Guiding**
  - System performs an on-going task, and user controls it as needed
  - Game engines, interactive installations
Levels of Interaction (cont.)

□ Adaptive
  ■ System learns something about the user, and alters responses accordingly
  ■ User learns from information provided by the system, and alters his/her actions accordingly
  ■ As user becomes expert, interface morphs

□ Conversational
  ■ User and system work as a team to determine proper actions
  ■ Multi-modal (e.g., sound, facial expressions, hand gestures)
Messages

- Input and output happen using *messages*
  - Text, speech, visual feedback, physical input/feedback

- Ambiguity of messages can be a problem
  - Cryptic error messages
  - Pointing in a crowded (real or virtual) space

- Every new interface requires training to achieve mastery
  - Though training time may be short

- Can leverage *previous experience*
  - Desktop metaphor

- Principle of Least Surprise
  - Familiar interactions are preferable
The Interface(s)

- The interface is the medium of the communication between user and system.
- It limits or enables **efficiency** and **effectiveness**
  - The user should never apologize for doing something wrong. The designer should.
- There is a balance between form (attractiveness) and function (usefulness)
  - Some systems make you choose one or the other
  - Some people choose one over the other
Pause
The Process of System Creation

The process of creation can be differentiated from the content of the creation.

Steps help us in several key ways:
- Thinking before doing
- Not re-inventing the wheel
- Participatory design
- Iteration
- Prototyping
- Graceful escape
- Planning for future features/additions
Steps in the Process

1. Concept
2. Research
3. Design
4. Build
5. Test
Concept

- What is the initial idea for your application?
- Draw pictures, diagrams, etc.
- Talk to the client (if there is one)
- What should the application do?
- How should it look?
- Sketching without a clear plan can lead you to exciting places.
- Don't write any code!
Research

☐ Who is your target audience?
☐ What environment (context) will they be working in?
☐ What have others done that is similar?
☐ What parts are needed to make up the whole?
☐ What approaches could you use for the individual parts of the system?
☐ Will you use existing components, build new ones, or buy new ones?
   ■ Classic build vs. buy decision
☐ Is what you are proposing really feasible?
Design

- Need to design both the hardware and software
- What are the tradeoffs for your choices?
  - Speed vs. space (in computation)
- What are the constraints on your system?
  - Size? Weight? Battery life? Cost?
  - Distraction of the user?
- Clearly define
  - How all the parts will appear to the user, and
  - How the user will interact with them.
- Flow diagrams (control and data) will help describe the system
- What do the interfaces between components look like?
  - APIs
  - Protocols
Build

- With your design(s) in hand, start building!
- Good approach
  - Don't try to build the whole thing at once!
  - Build a little, test a little, integrate, repeat
- Hardware
  - Assemble (build or buy) your hardware
  - Do low-level testing (debugging)
- Software
  - You need to talk to the hardware, user
  - What language(s) will you use?
- Integration
  - Always takes longer than you think it will
  - Designing is hard
Test

- Testing is always the first thing to be sacrificed
  - Ever played any buggy games?
  - Ever patched a game, or any software, right after you bought it?

- Many levels of testing
  - Components
  - Integrated system
  - End-user testing
  - Balance testing (games)
  - Alpha, Beta, open, closed?

- Hardware
  - Build it in simulation
  - Build a breadboard version
  - Build a "quickboard" version
  - Have PCBs made and populated
  - Revise
Final Thoughts

☐ Be open to iterate at any step!
   ■ This is not a "waterfall model"

☐ Many projects have milestones
   ■ Show to client/publisher
   ■ May be canned at that point (graceful escape)

☐ Teams can make better solutions than individuals
   ■ Usually, anyway
   ■ More heads thinking about problem
   ■ Greater breadth of experiences to draw upon
   ■ Variety of expertise

☐ Need to instill ownership of each part
   ■ Who is the go-to person on this part?