Lecture 5

Nondigital Prototypes
Review: Prototypes

- An *incomplete* model of your product
  - Implements small subset of the final features
  - Features chosen are the most important now

- Prototype helps you visualize *gameplay*
  - Way for you to test a new game mechanic
  - Allows you to tune mechanic parameters
  - Can also test (some) user interfaces
Software Prototypes

- **Gameplay Prototype (2/23)**
  - Throw-away prototype (not in final submission)
  - Does not have to be on device
  - Should demonstrate core gameplay

- **Technical Prototype (3/9)**
  - Evolutionary Prototype (part of final submission)
  - Should be on a device except in extreme cases
  - Should demonstrate important mobile challenge
Next Week: Nondigital Prototype

- No software involved at all
  - Board game
  - Card game
  - Something different?

- Goal is to **model gameplay**
  - How? Nondigital/digital is very different
  - Model will be far removed from final result
  - What can we hope to learn from this?
Understanding Game Progression

- Level design about *progress*
  - Sense of closeness to goal
  - Choice of “paths” to goal (*dilemma challenge*)
  - Path choice can relate to play style and/or difficult
- Easier to design if *discrete*
  - Flow-chart out progression
  - Edges are mechanic(s)
- But game state values are *continuous* (sort of)
Discrete Progression

- Design is **discretization**
  - Impose flow chart on state
  - Each box is an **equivalence class** of game states

- **Spatial Discretization**
  - Contiguous zones
  - **Example**: past a doorway

- **Resource Discretization**
  - Range of resource values
  - **Example**: build threshold
Spatial Discretization
Discretization and Turns

- Discretization requires *turns*
  - Represent a unit of action
  - When done, game “at rest”

- Turns can be *multistep*
  - Multiple actions in a turn
  - Environmental interactions

- Turns can *alternate*
  - between other players
  - with a gamemaster
  - not at all (one player?)
Discretization and Reaction Time

- Allow opponent to **interrupt**
  - Action that reacts to yours
  - Played after you act, but before action takes an effect
  - Core mechanic in *Magic: TG*

- Make play **asynchronous**
  - Players still have turns
  - But take turns as fast as can
  - Conflicts resolved via speed
  - Often need a referee for aid
Reaction Time: *Runaway Rails*

- “Free runner” with coaster
  - Coaster can go faster/slower
  - Speed tests reaction time
- Model with hidden info
  - Cannot “process” all at once
  - Faster go, less screen to see
Reaction Time: *Runaway Rails*

Speed changes # of columns at each turn
What Can We Do Discretely?

- **Evaluate emergent behavior**
  - Allow player to commit simultaneous actions
  - Model interactions as “board elements”

- **Model player cost-benefit analyses**
  - Model all resources with sources and sinks
  - Focus on economic dilemma challenges

- **Test player difficulty/usability**
  - Ideal for puzzle games (or puzzle elements)
  - Can also evaluate unusual interfaces
What Can We Do Discretely?

- Evaluate emergent behavior
  - Allow player to commit simultaneous actions
  - Model interactions as “board elements”
  - Model player cost-benefit analyses
  - Model all resources with sources and sinks
  - Focus on economic dilemma challenges

- Test player difficulty/usability
  - Ideal for puzzle games (or puzzle elements)
  - Can also evaluate unusual interfaces

New issues for mobile games

Not that different from CS 3152
Discretizing Resources

Jetpack expends oxygen (=health)
Discretizing Resources
Interactions: RoboRally

- Player “programs” robot
  - Picks 5 movement cards
  - Committed to that choice
- After each card
  - Obey board elements in order
  - Check robot collisions
- Move = board elements + cards + collisions
Usability Analysis

• Unusual user-interfaces
  • Recall that actions correspond to inputs
  • Some inputs are not simple buttons
  • Example: touch gestures, motion controls

• Puzzle-style games
  • Create a game with module elements (e.g. cards)
  • Laying out levels creates a new game level
  • Allows you to quickly change and test levels
Usability Testing: Angry Bunny

Early Design:
Bunny movement affected by multiple battery “attraction”
Modeling Movement Controls

Strings attached at board corners

Control piece by pulling strings
Modeling Gestures

- Usability includes custom touch gestures
  - How accurate is player gesture?
  - How repeatable is the gesture?
  - How much screen does it require?

- Test out gestures with a writing tool
  - Write directly on the game board
  - Put a transparency over the game board

- Does not help with multi-touch (solution?)
Example: Space War

- Board split in half
- Side to each player
- Ships are dots
Example: Space War

- Board split in half
- Side to each player
- Ships are dots
- Variations exist
Gestures in Space War

Shoot by flicking pencil quickly
Gestures in Space War

Shot must start on player’s ship
Gestures in Space War

Must be a quick flick (judgment call)
Processing Laser Fire

Hit!

Miss
Space War Rules

- Players alternate turns shooting
  - **Remember**: Must start at active ship
  - Friendly fire usually ignored
  - Variations allow limited ship movement

- Successful hit destroys a ship
  - Variations may need multiple hits

- Continue until one side has no ships
  - Or paper is too messy to play anymore
Experiential Prototypes

• Some prototypes do not test gameplay
  • They test an experience or feeling
  • You determine if the feeling is enjoyable
  • Then go back and design gameplay for that

• Be very *careful* with this!
  • A very advanced design technique
  • Can easily end up with worthless prototype
  • Have only seen a few successes at this
Experiential Prototype: Aeronautical
The Experience of Threat
Most Important Thing: *Progression*

- Do not want a **one-level** game
  - Major problem with “flick” games in this course
  - Endless runners also have this problem

- We want some evidence of a **progression**
  - What is an easy level?
  - What is a medium level?
  - What is a hard level?

- Your prototype should be **reconfigurable**
Easy
Medium
Hard
The Difficulty Curve

Easy  Medium  Hard
Easy: Iridescence
Medium: Iridescence
Hard: Iridescence
Reconfigurable Prototypes