Lecture 7

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 (3/2)**
  • Throw-away prototype (not in final submission)
  • Does not have to be on device
  • Should demonstrate core gameplay

• **Technical Prototype (3/14)**
  • 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
Spatial Discretization
Spatial Discretization
Nature of Discretization

• State must be **unambiguous**
  • Must be an accurate, precise way to determine state
  • **Example**: string to measure distance in a wargame

• Actions must be **significant**
  • May correspond to several animation frames
  • **Example**: movement and attack in single turn

• Mechanics must have **compact interactions**
  • Avoid mechanics that depend on iterated interactions
  • **Example**: physics is *iterative* and hard to discretize
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

---

**Mobile Prototypes**
Case Study: 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 as Hidden Information

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

Not that different from CS 3152

New issues for mobile games
Evaluating Emergent Behavior

- **Recall**: coupled, context-dependent interactions
  - Requires an action and interaction
  - Or (alternatively) multiple actions

- Model interactions as “board elements”
  - Rules to follow after your action
  - May follow several in succession
  - **Examples**: Chutes & Ladders, Bonkers, RoboRally
Case Study: *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
Cost-Benefit Analysis

- Where nondigital prototypes really shine
  - Resources are very easy to discretize
  - Economic choices easily map to turns
  - Understanding dilemma challenges is important

- Some believe this is *all* of game design
  - Claim everything can be reduced to a resource
  - Common in board game adaptations of other media
  - **Example**: balance game with instability resource
Case Study: *Bounce*

Jetpack expends oxygen (=health)
Tracking Oxygen as a Resource
Case Study: Trino

Can switch w/ resources
Measuring Shapeshifting Resources
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
Case Study: Angry Bunny

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

Strings attached at board corners

Control piece by pulling strings
Case Study: Coalide
Modeling Flick Controls
Case Study: *Family Style*

**PASS INGREDIENTS FROM PHONE TO PHONE**
Modeling Multiplayer Restrictions
Case Study: *Operation Bitwise*
Configurable Prototype from Elements

Mobile Prototypes
Case Study: *Magic Moving Mansion*
Configurable Puzzles at Scale
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
Case Study: *Gathering Sky*

Mobile Prototypes
Feel of Movement Controls
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

Mobile Prototypes
The Difficulty Curve

Easy

Medium

Hard
Case Study: *Iridescence*
Easy: *Iridescence*
Medium: *Iridescence*
Hard: *Iridescence*
Case Study: *Project Apollo*
Prototype is a Puzzle Sandbox
Reflecting on What You Have Learned

• Your prototype should teach you *something*
  • About one of the things covered today
  • Even if it is “this design will not work”

• You will be asked about this at *presentation*
  • Must be prepared to answer
  • Write-up as part of submission

• Lesson matters more than *physical artifact*
  • You are not going to sell this prototype
Case Study: Flourish
Case Study: Flourish

Our game seemed unclear at the beginning for some players because [they had to conceptually] balance growth above ground and below ground.

...  

In general, we learned about the specificity we need for different rules that we had thought needed less explanation.