Lecture 2

Mechanics Revisited
Purpose of Today’s Lecture

- Give a review of formal **design elements**
  - Not everyone here has had the Intro Games course
  - And for the rest of you, it has been over a year

- Develop a deeper understanding of **mechanics**
  - Understand the important of interactions
  - Understand the **analysis** challenges

- Set us up for the **later lectures** on mechanics
  - Why do we need **game grammars**?
Actions

- **Verbs** that describe what the player can do
  - Walk  (left or right)
  - Run    (walk, but faster!)
  - Jump   (up; jump/run for left or right)
  - Shoot  (left or right)

- Does not need to be attached to an avatar
  - Build  (RTS or simulation)
  - Swap   (Bejeweled clones)
  - Rotate (Stacking games)
Primary Actions

- How do verbs, goals relate?
  - Imagine there are no challenges
  - What verbs *must* you have?

- **Example**: Platformers
  - **Goal**: reach exit location
  - Only need movement verbs
  - Killing enemies is *optional*
  - Other actions are *secondary*

- **Design Goal**: Primary only
  - Secondary verbs lead to bloat
  - Add features with interactions
Secondary Actions are Acceptable

- Often in **puzzle platformers**
  - Platformer verbs + something
  - “Innovation on the cheap”
- Verb that alters “geography”
  - Access hard-to-reach areas
  - Directly overcome *challenges*
  - Not directly needed for goal
- But do this sparingly!
  - Indies have one new verb!
  - Other features are *interactions*
Interactions

- Not a *direct* action of player
  - Outcome of the *game state*
  - Can happen without controller
- **Example**: collisions
  - Accidental or player forced
  - May be bad (*take damage*)
  - May be good (*gain power-up*)
- **Other Examples**:
  - Spatial proximity
  - Line-of-sight
  - Resource acquisition
Game Mechanics

- **Game mechanic**
  - Relationship between verbs and interactions
  - Often call this relationship the “rules”
  - **Gameplay** is manifestation of these rules

- **Example**: Joust
  - **Verbs**: Flap; go left or right
  - **Interaction**: Collision with opponent
  - **Rule**: If hit opponent, lower player dies
Gameplay Example: *Joust*
Design Goal: Verb Minimalism

- Can we limit to **one** verb?
  - Mechanics are all interactions
  - Common in mobile, tablet
  - Due to lack of input modes

- **Example**: Sneak Beat Bandit
  - Has only one verb: *move*
  - Rhythm game; move to beat
  - All movement on rails
  - If obstacle in way, turn
  - Line-of-sight mechanics
Beat Sneak Bandit
Avoid Verb Proxies

• **Proxy**: verb that activates another verb
  • “Use an item” (what does the item do?)
  • “Shoot” (what does the weapon do?)

• Make your verbs **outcome oriented**
  • Fire standard projectile (like shoot, but says what it shoots)
  • Fire freezing beam (what is does and how it is applied)

• Important questions to ask
  • Does it help me reach a goal?
  • Does it overcome a challenge?
Avoid Verb Proxies

- **Proxy**: verb that activates another verb
  - “Use an item” (what does the item do?)
  - “Shoot” (what does the weapon do?)

- Make your verbs **outcome oriented**
  - Fire **standard projectile** (like shoot, but says what it shoots)
  - Fire freezing beam (what does and how it is applied)

- Important questions to ask
  - Does it help me reach a goal?
  - Does it overcome a challenge?
Combining Actions

- Verbs can combine in interesting ways
  - Run and jump in a platformer
  - Strafing fire in a shooter

- Typically result of the interactions
  - Each verb interacts with environment in different way
  - Combination of two give extra feature for “free”
  - This is an example of emergent behavior

- Not all combinations are emergent
  - Example: Double jump is not a feature of interactions
  - This type of verb combination is a distinct action
# Combining Actions

## Running Jump
- Can move while in midair
  - Just horizontal movement
  - Not realistic; it is a game
  - Many platformer challenges assume this type of control
- Different than a *long jump*
  - Less height than reg. jump
  - No control once in the air
  - Would be a *distinct action*

## Strafing Fire
- Based on “real life” property
  - Bullets travel in straight line
  - Movement changes origin
  - Walking side-side makes a spray (used in covering fire)
- But some features are gamy
  - Bullets slower than life
  - Character faster than life
  - Creates interesting effects
Combining Actions

**Jump**
- Can move while in midair
  - Just horizontal movement
  - Not realistic; it is a game
  - Many platformer challenges assume this type of control
- Different than a *long jump*
  - Less height than reg. jump
  - No control once in the air
  - Would be a distinct action

**Strafing Fire**
- Based on “real life” property
  - Bullets travel in straight line
  - Movement changes origin
  - Walking side-side makes a spray (used in covering fire)
- But some features are gamy
  - Bullets slower than life
  - Character faster than life
  - Creates interesting effects

*Interaction(?)*
Is this an example? **Why** or **why not**?
Interactions and Limitations

- You cannot always perform an action
  - Shooting may require ammo
  - Cannot (always) jump in mid air

- **Limitation**: requirement to perform action
  - Boolean test (like an if–then)
  - Only **one** limitation per verb
  - If more than one, split into more verbs

- **Example**: double jump is different from jump
Game State

- **Game State**: values that represent the game world at a specific moment in time
- **Interaction**: function between game states
- Many of these values are **spatial**
  - Represent location of various game objects
  - Also physical values like velocity, acceleration
- Other **changeable** state values are **resources**
  - Governed by your **game economy**
# Spatial Interactions

## Extrinsic
- State affects *other* objects
- Often **detection** mechanics
  - Line-of-sight
  - Proximity
- But can include **collisions**
  - Damage an opponent
  - Move an object

## Intrinsic
- State affects the *player*
- Affect player **resources**
  - Player takes damage
  - Player transfers gold
- Can effect player **position**
  - Bounce off obstacle
  - Attraction to magnet

---

**Most difficult to handle**
Intrinsic Effects Have Feedback Loops

- Current pos. → new pos.
  - Which leads to new position
  - Unless player can escape…

- Game state is a **vector field**
  - Each pos. a velocity vector
  - Indicates pos. at next frame
  - Implicitly defined by code

- So you can use science of vector fields to understand
  - With several caveats
Singularities: A Design Problem

- **Convergent Singularities**
  - Vectors lead to line or point
  - Sucks player into a location
  - Removes player agency

- **Divergent Singularities**
  - Conflicting directions
  - Player “trapped” there
  - Worse than convergent

Particularly a problem in “pure physics” games.
Visualizing Vector Fields

- Seeing fields helps design
  - What is stable (convergent)?
  - What is unstable (divergent)?

- Have to draw “possibilities”
  - What if player is here?
  - Rerunning physics sim at multiple player locations
  - Very compute intensive

- Also, possibility of whom?
  - Just player or other objects?

- **Visualization Ideas?**
Non-Spatial Interaction

**Numeric State**
- Does not change: **attributes**
  - **Example**: item strength
  - Part of *data-driven design*
- Can change: **resources**
  - **Example**: ammo, health
  - Often *limit* actions
  - Also mark “partial” success

**Symbolic State**
- Lock-and-key mechanics
  - Do you have item X?
  - Possession needed to pass
- Also for “slot” mechanics
  - Item gives attribute boost
  - Only one of item type
  - Creates *dilemma challenge*

Most difficult to handle
Balancing Resources

• **Sources**: How a resource can increase
  • **Examples (player)**: ammunition clips, health packs
  • **Example (external)**: spawn points

• **Drains**: How a resource can decrease
  • **Examples (player)**: firing weapon, player damage
  • **Examples (external)**: monster death

• Adjust sources and sinks to “balance” economy
  • Together, determine “price” of resource
  • Price of resource should reflect its “power”
Design Problem: Pricing Resources

Underpricing

- Cheap, powerful actions
  - Players favor these verbs
  - Limits play variety

- Examples:
  - Buff spells in most RPGs
  - *Dragon Age* cold spells
Design Problem: Pricing Resources

Overpricing

- Expensive, weak actions
- Usage is “penalized”
- Waste of designers’ time

Examples:
- Shredder ammo in ME2
- Raise Dead in early D&D
# Design Problem: Pricing Resources

<table>
<thead>
<tr>
<th>Underpricing</th>
<th>Overpricing</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Cheap, powerful actions</td>
<td>• Expensive, weak actions</td>
</tr>
<tr>
<td>• Players favor these verbs</td>
<td>• Usage is “penalized”</td>
</tr>
<tr>
<td>• Limits play variety</td>
<td>• Waste of designers’ time</td>
</tr>
<tr>
<td>• Examples:</td>
<td>• Examples:</td>
</tr>
<tr>
<td>• Buff spells in most RPGs</td>
<td>• Shredder ammo in ME2</td>
</tr>
<tr>
<td>• <em>Dragon Age</em> cold spells</td>
<td>• <em>Raise Dead</em> in early D&amp;D</td>
</tr>
</tbody>
</table>

- Resource usage determines difficulty
  - *Resident Evil*: Availability of ammunition
  - D&D 3.x: 20% resource per encounter
Resource Analysis: Dungelot

- Simple combat mechanic
  - Each round, swap damage
  - Enemy dies when health is 0

- Player goes until health is 0
  - There is healing in game
  - …but too sparse to go forever

- Two primary characters
  - **Paladin**: can lessen damage
  - **Vampire**: drains blood to heal
  - Which is better?
Bad Design: “Engines”

- Actions combine to make resources free
  - Spend one resource to get another
  - Use new resource to get old one back

- Example: Dragon Age
  - Resources: Health, Mana
  - Small health loss; regain much mana
  - Small mana loss; heal much damage

Solution? Cool-down time
Resources Have a Visualization Problem

- Resource mechanics often economic transactions
  - State is available resources
  - Action trades old for new
  - New state: result of “purchase”

- How can we see change?
  - State is high dimensional
  - Paths are non-exclusive

- Need special models
  - Resource flow graphs?
  - Petri Nets?
Summary

• **Mechanics** combine *actions* and *interactions*
  - Actions are a direct result of player controls
  - Interactions triggered by a particular game state

• Interactions depend on the **game state**
  - Spatial state associated with physics, detection
  - Resources associated with limitations, unlocking

• **Visualization** is a difficult problem