

Lecture 2

Mobile Gameplay

Focus of Today's Talk



Smartphones



Tablets

And In a Few Ways, This



Challenge: Input Modality

- Don't have standard gamepad controls
 - Add-on hardware is unpopular
 - Not standard, few games use
- Loss of a lot of functionality
 - D-Pads, joysticks for avatar control
 - Buttons for performing core actions
- Have to **rethink game input**



The Cheap Way Out



The Cheap Way Out



So What Can We Do?

- (Multi) Touch Controls
 - Pointing, dragging
 - Clicking, selecting
 - More advanced gestures
- Accelerometer Support
 - Tilting
 - Rotating



So What Can We Do?

- (Multi) Touch Controls

- Pointing, dragging

- Clicking, selecting

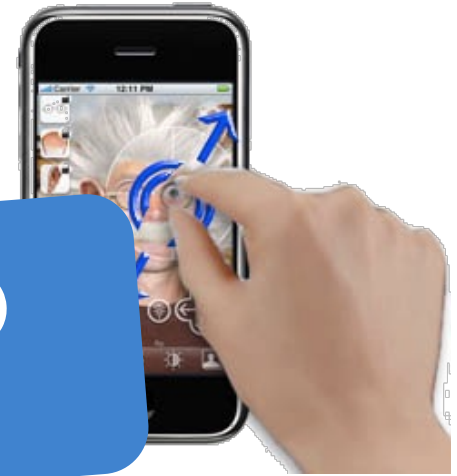
- More

AR features (light, camera)
are also a possibility.

- Accelerometer Support

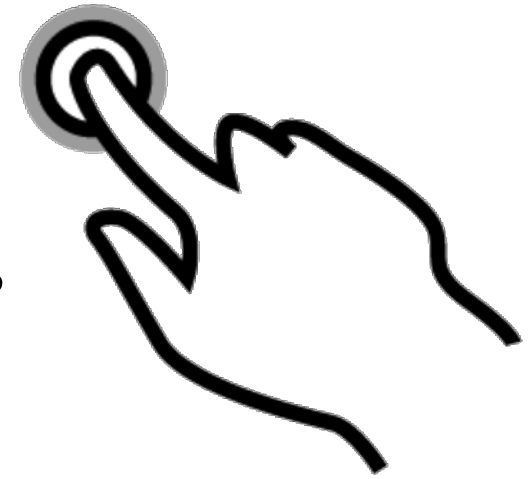
- Tilting

- Rotating



Touch: Basic Approach

- Can use touch interface like a **mouse**
 - Touch to click on a point,
 - Trace from touch to drag
- Port mouse-heavy PC/Mac games
 - Particularly strategy games/RPGs
- Keyboard exists, but is limited
 - Have to obscure screen to pull up keyboard
 - Use very sparingly (e.g. save file)



Example: *Plants vs. Zombies*



4152 Example: *Gathering Sky*



Balancing Multitouch

- PC games are “balanced” for a single pointer
 - Multitasking requires a lot of back and forth
 - Challenge is to do actions in an efficient order
- Multitouch eliminates this challenge
 - Fingers everywhere!
 - Movement is fast
 - **Ex:** *Whack-a-Zombie*



Size Matters

- Small screen makes multitouch *hard*
 - True multitouch only on a tablet
 - Phones are largely limited to gestures
- Fingers are **fatter** than pointers



Size Matters

- Small screen makes multitouch *hard*
 - True multitouch only on a tablet
 - Phones are largely limited to gestures
- Fingers are **fatter** than pointers



Click versus Pointing

- PCs use **hover** to give information
 - Gives pop-up menus, tool-tips
 - Used in RPGs, strategy games
 - Major UI design technique
- There is no hover on mobile!
 - How to distinguish action from info?
 - **Press-and-hold** is becoming the standard
 - So actions must happen on **release**, not press.

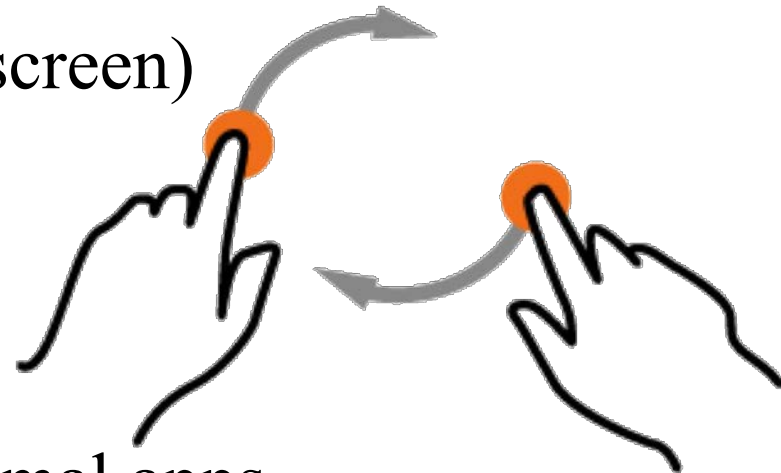


Example: Assassin's Creed Rebellion



Touch: Gestures

- Can also leverage device **gestures**
 - Manipulation strokes common to device
 - **Example:** Pinching for zoom
 - **Example:** Rotating (object, screen)
- Natural for camera control
- **Design Approach:**
 - Think about how used in normal apps
 - How do you leverage this in a game?



Basic Gestures



Tap



Double Tap



Tap and Hold



Flick



Pinch



Spread



Rotate



Drag (Scroll)

Simple Multitouch Gestures

Two Fingers



Tap



Tap/Press



Double Tap



Drag

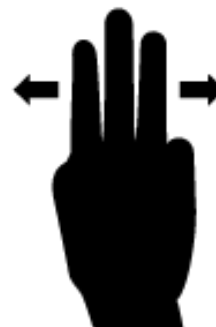
Three Fingers



Tap



Double Tap



Swipe



Drag

Simple Multitouch Gestures

Two Fingers



Tap

Sort of possible
to get position



Tap/Press



Double Tap



Drag

Three Fingers

Getting position is
largely hopeless



Tap



Double Tap



Swipe



Drag

4152 Example: *Phantom Escape*



“Stretch”
boxes to
launch

4152 Example: *G.M.P.*

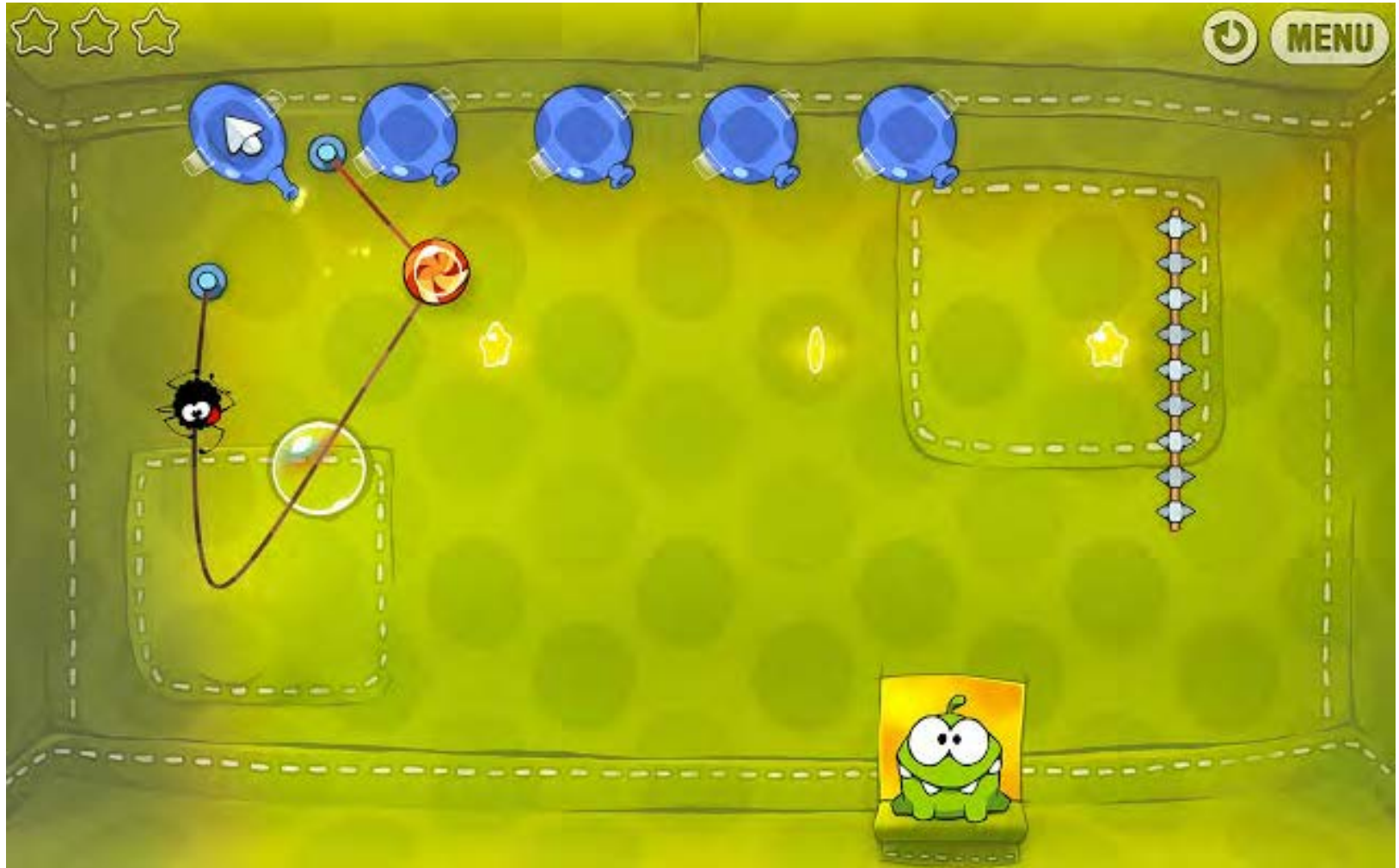


Touch: Natural Controls

- Successful games strive for **natural controls**
 - Verb controlled by a single movement/gesture
 - Gesture has a very natural physical feel to it
 - Maps naturally on to the action in the game
- **Examples**
 - Cutting (Cut the Rope)
 - Tracing (Flight Control)
 - Pulling (Angry Birds)
 - Twisting (Monument Valley)



Example: *Cut the Rope*



Example: *Flight Control*



4152 Example: *Flick Ship Spaceship*



Example: *Zen Bound*



Example: *The Room*



Example: *Monument Valley*



Specialized Gestures: *Infinity Blade*

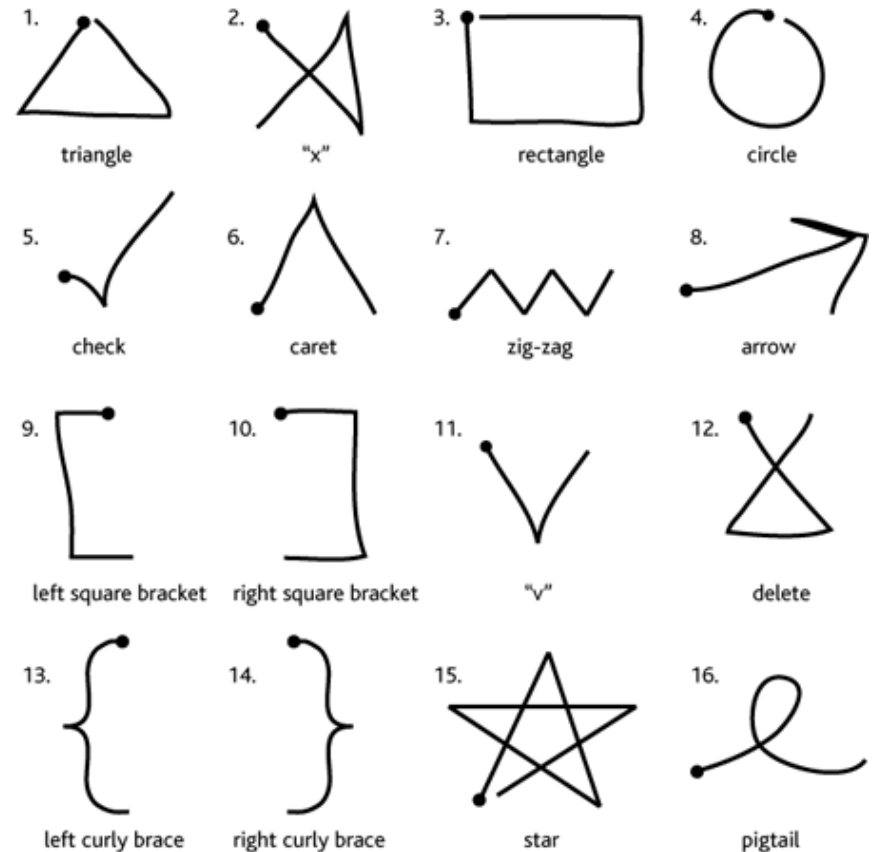


Draw
symbol on
screen

Spell is
cast when
done

Dollar Gestures

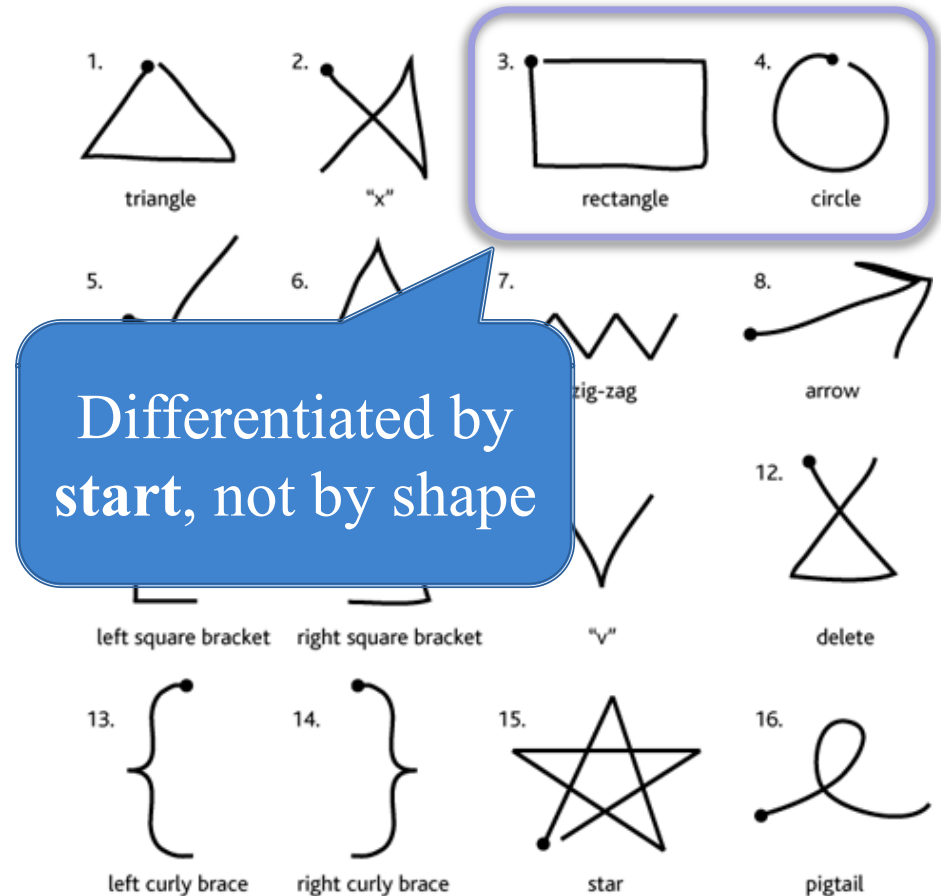
- Recordable gesture API
 - Created a U. Washington
 - Code freely distributed
- Very limited resolution
 - Scales gesture to pixel grid
 - Grid uniquely identifies
 - Shape AND start matter
- **MEng project from 2020**
 - **Included in CUGL API**



<http://depts.washington.edu/madlab/proj/dollar>

Dollar Gestures

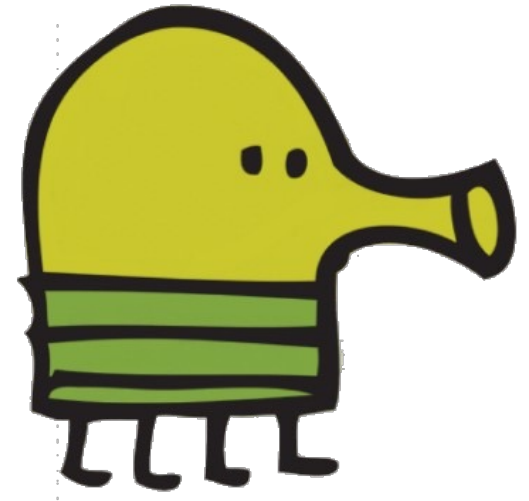
- Recordable gesture API
 - Created a U. Washington
 - Code freely distributed
- Very limited resolution
 - Scales gesture to pixel grid
 - Grid uniquely identifies
 - Shape AND start matter
- **MEng project from 2020**
 - **Included in CUGL API**



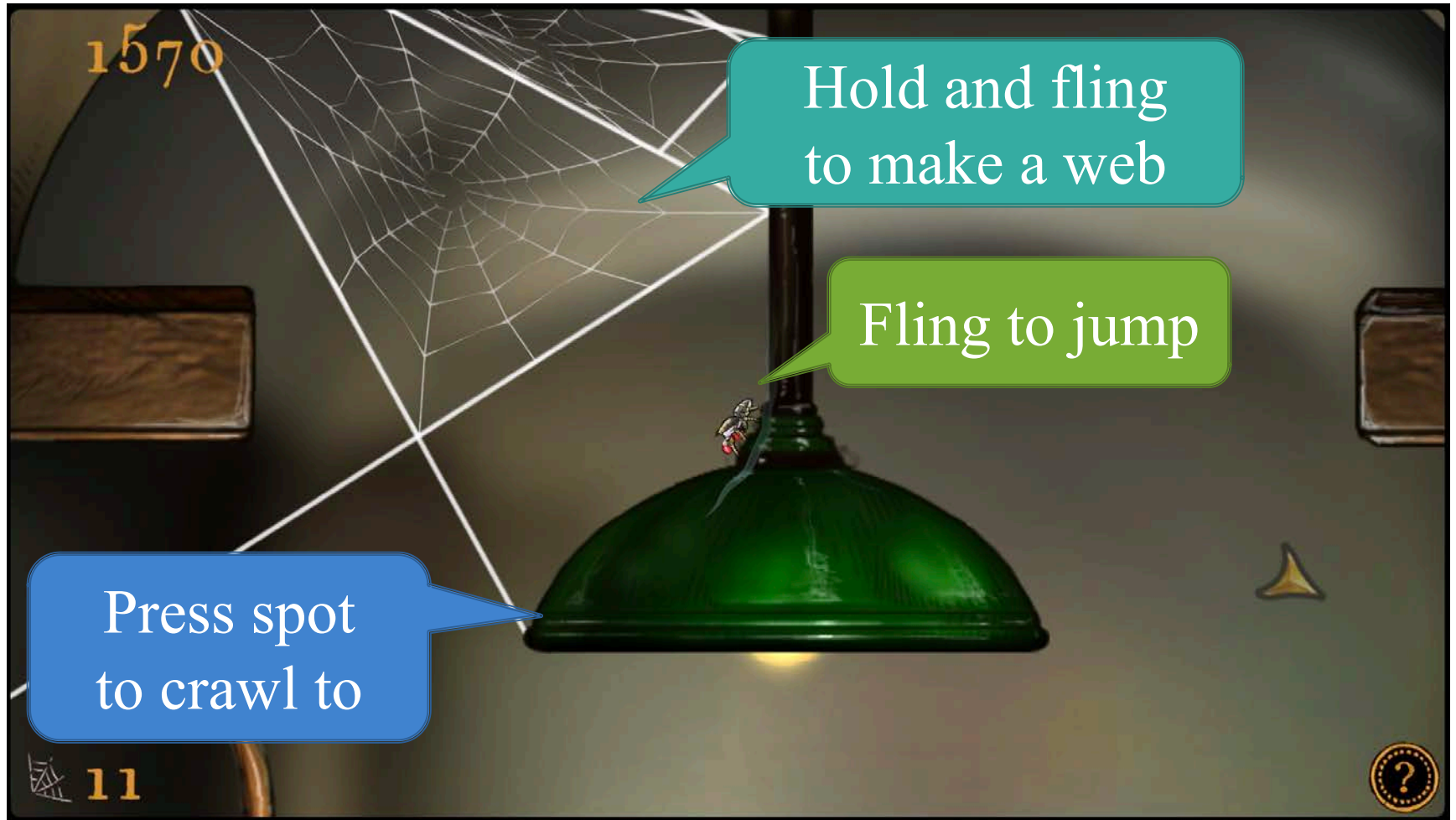
<http://depts.washington.edu/madlab/proj/dollar>

Touch: Avatar Controls

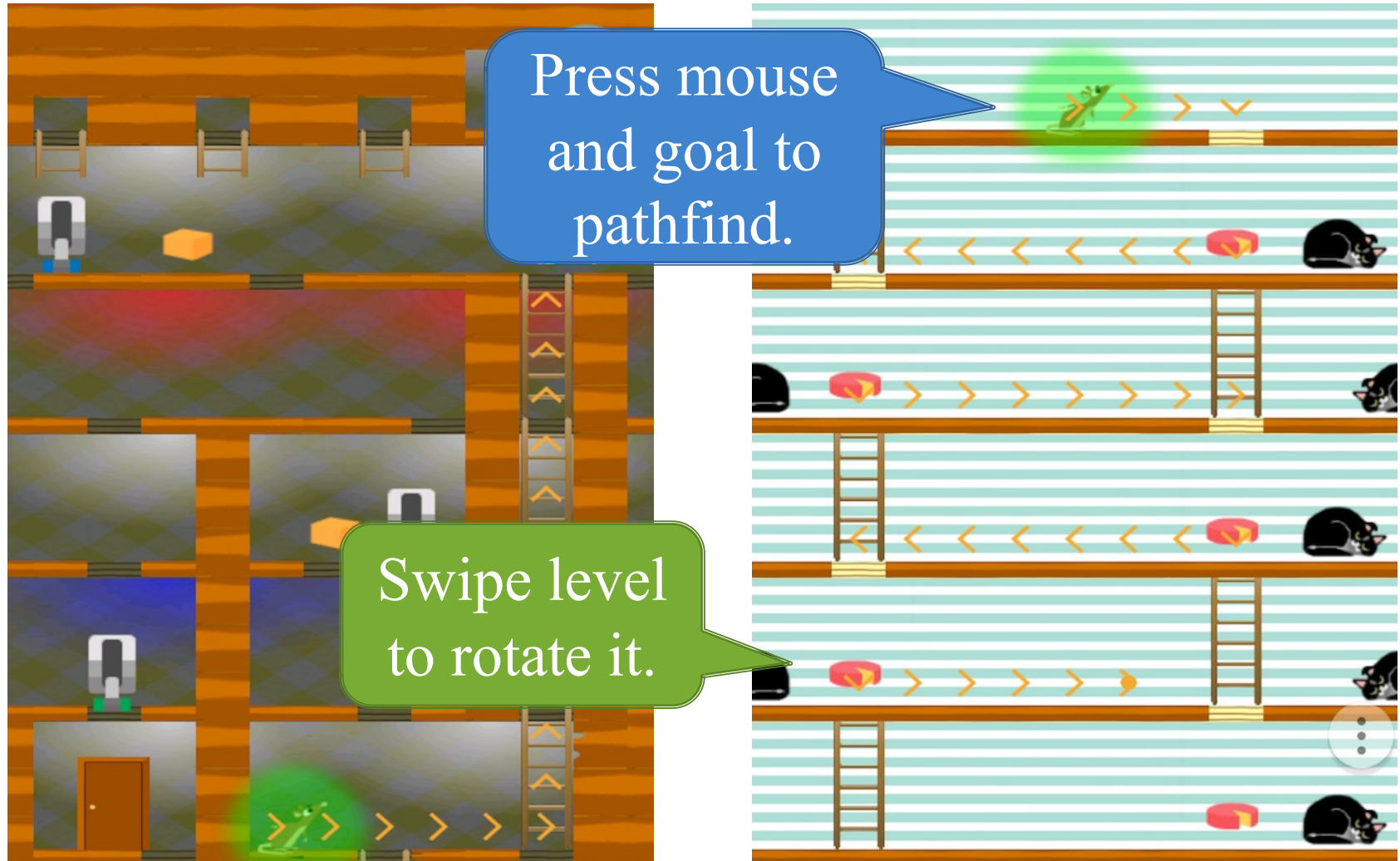
- Several (non-joystick) options for movement
 - Drag the character
 - Point to a waypoint
 - Point to direction
- But how to indicate avatar actions?
 - Want to move and act at same time
- **One Solution:** put actions into movement modes
 - Drag versus waypoint
 - Press+hold drag versus drag



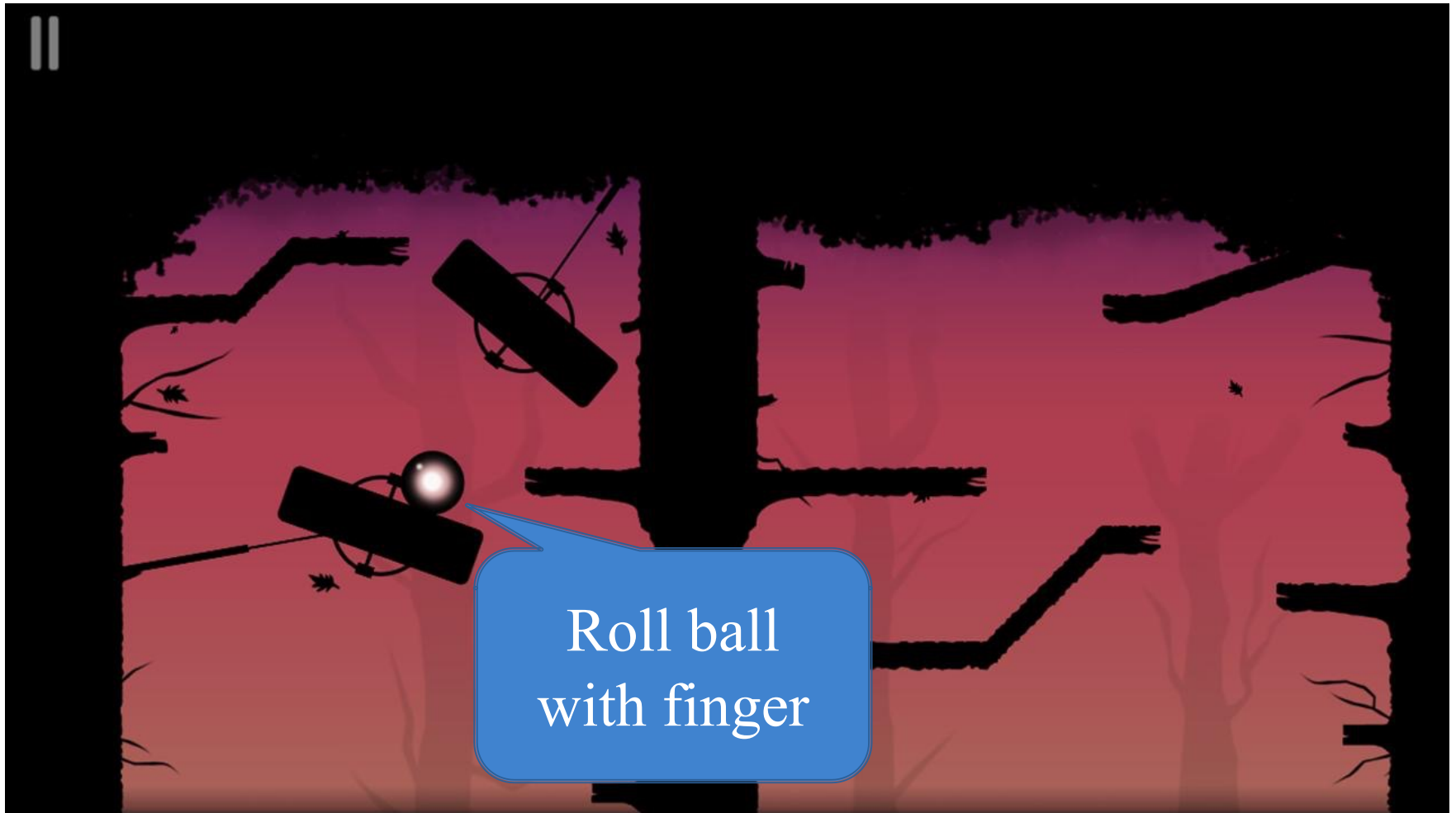
Example: *Spider*



4152 Example: *Squeak & Swipe*



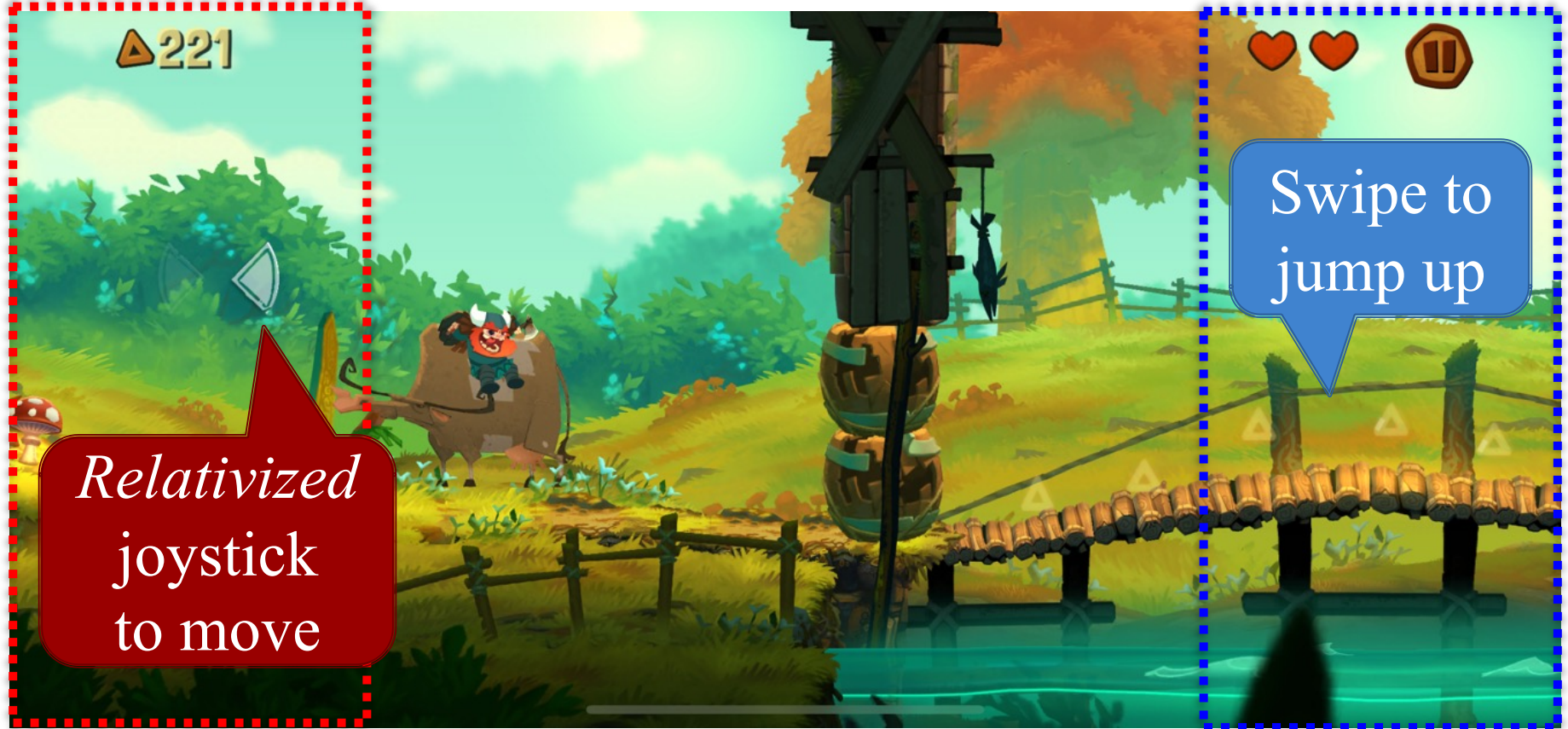
Early Platformer: *Night Sky*



Early Platformer: *Type:Rider*



Modern Platformer: *Oddmar*

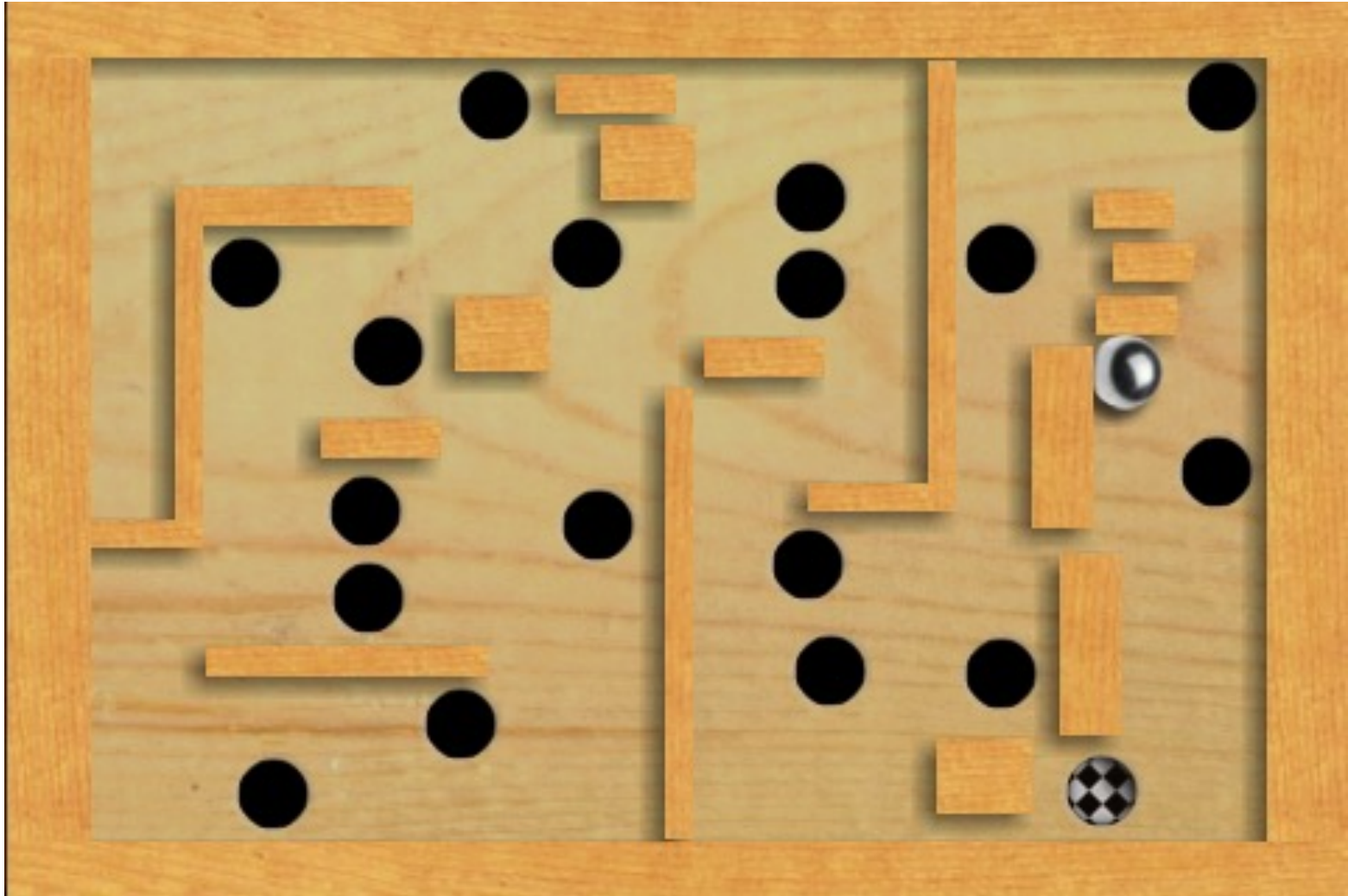


Accelerometer: Basics

- **Can** detect rotational movement
 - Rotate from flat plane
 - Rotate around edge
- **Cannot** detect other movement
 - Lateral movement of device
 - Absolute position of device
- Ideal mechanic for
 - Marble-style games
 - Steering/On-rails games



Example: *Labyrinth 2*



Accelerometer + Touch

- Solves the problem of actions
 - Use accelerometer for movement
 - Use touch for other actions
- But have to hold the device
 - Hard to gesture as well
- **Idea:** Keep actions unobtrusive
 - Avoid "button mashing" mechanics
 - Allow touch to use thumbs as much as possible



Example: *Nightmare Tower*



Accelerometer: Challenges

- The control device is the **display**
 - Extreme controls make game hard to see
 - Even worse when combine with touch
- Even basic movement is a **challenge**
 - Hard to quickly change directions
 - Prone to overcorrection
 - **Example:** *Labyrinth*



Accelerometer: Orientation

- Can detect device orientation
 - Either portrait or landscape
 - Use for different game modes
- *Sword & Sworcery EP*
 - Landscape for exploration
 - Portrait for combat
- Supported in SDL/CUGL
 - 3rd year in CUGL
 - Add listener to **Display**



Example: *Flipped Out!*



What About the Steam Deck?



Touch screen: Yes
Accelerometer: No*

Final Word: Know Your Audience

- **Phone games** are meant for "quick play"
 - Must be able to start, play, and save in 2 minutes
 - Should be able to pick up where left off quickly
 - Controls should be (relatively) simple
- **Tablet games** can be more complex
 - Supports longer play units (why?)
 - Larger screen permits more complex controls
 - Games are closer to PC indie games
 - And can also cost more!