CSCI 1106
Lecture 05
Player Movement
Announcements

• Quiz is tomorrow in class
• Today’s Topics
  – Finish Collision Detection
  – Motivation for player movement
  – Mouse Movement
  – Easing
  – Keyboard Movement
Player Motion

• All interactive games have player movement
  – Players can move their character or avatar on the screen
  – Players can react to the game and move their avatar
• How the avatar moves is dictated by the game’s
  – Laws and physics of the game
  – Goals and objectives
  – Environment and level of play
• Common ways to move the avatar are through
  – Mouse
  – Keyboard
  – Dedicated game controllers and joysticks
Direct Mouse Movement

• Idea: Make the player the ”mouse”
  – The avatar appears where the mouse is pointing to
  – No need to control the velocity of the avatar
  – Position and velocity is managed by the mouse movement

• How:
  – Set the player sprite’s coordinates to the mouse coordinates at each FRAME event
Direct Mouse Movement

• Pros:
  – Easy
  – Not much code required

• Cons:
  – Restrictions on movement may be needed, e.g.,
    • Disallowing movement in some dimensions (paddle)
    • Checking if mouse is over the game panel area
  – Violates most accepted laws of physics
    • Avatar can accelerate and move instantly

• How can we solve these problems?
Mouse Movement using Easing

• Idea: gradually move avatar toward the location clicked on with the mouse pointer
  – A mouse click sets the target to move toward
  – Calculate distance between the avatar and target
  – Incrementally move the avatar toward the target
  – Note: the avatar isn’t guaranteed to reach the target because the target will change if another location is clicked first

• Pros:
  – Makes the physics of the game more realistic
  – Restricts avatar movement by ignoring clicks on illegal areas of the stage

• Cons:
  – Allows only coarse-grained movement
Implementing Easing

- Declare an EASING constant
  - $0 < \text{EASING} < 1$
  - Smaller constant implies slower movement

- Create a transparent “Target” sprite

- Set “Target” at avatar’s location

- On each FRAME event If the mouse is down
  - Move “Target” to mouse location

- On each FRAME event
  - If avatar's distance to “Target” is greater than 1
    - point avatar at target
    - move avatar an EASING fraction of the distance to the target
Keyboard based Movement

• Idea: Move the player with the keyboard
  – The arrow keys control the direction that the avatar moves
  – These directions allow the player to move diagonally as well
  – Need to respond to the KEY PRESS events or check if keys are being pressed.
  – More than one key can be down at the same time

• Pros:
  – Very precise movement

• Con:
  – Requires the player to learn the control keys
Implementing Keyboard Controls

• On a FRAME event
  – Check which of the arrow keys are pressed and move in that direction

\[\text{left-90°, up 0°, right 90°, down 180°}\]