Lecture 18
GE211 & Animation

CS211 – Fundamentals of Computer Programming II

Branden Ghena – Winter 2022

Slides adapted from:
Jesse Tov

Administrivia

- Should be getting feedback on specs soon
  - Go ahead and get started now!

- Quiz 4 is next week Tuesday

Today's Goals

- Continue playing around with GE211

- Demonstrate how to find and fix bugs
  - Because I'm definitely going to have some

- Explore animation in games

Getting the code for today

- Download code in a zip files from here:
  https://nu-cs211.github.io/cs211-files/lec/15_finalProject.zip
  https://nu-cs211.github.io/cs211-files/lec/18_animation.zip

- Extract code wherever

- Open with CLion
  - Make sure you open the folder with the CMakeLists.txt

Outline

- **Game Motion Planning**

- Animations

Plan for game

- Image sprite that represents a character in the game
  - Moves towards a given position at a set velocity
- Text sprite to explain what position is being moved to
- Each character keeps a list of positions to move to
  - Moves towards the first position until it reaches it
  - Then starts moving towards the next position
- Add to list of positions with mouse clicks

What have we done so far?

- Created a character class
  - Holds the image sprite for the character
  - Keeps track of position and can update position
  - Keeps track of a destination and moves toward it

- Updated model, view, and controller
  - Creates a character (eevee)
  - Updates the character each frame
  - Draws the character at its position

- Fixed SEVERAL bugs along the way

Handle multiple characters

- Have model keep a vector of characters
  - Call update on each one
  - Draw can draw each one

- Game should be extensible for N different characters
  - Each doing with their own destinations
  - Each should have their own position, sprite, and transform

Add a text sprite to explain each character's movement

- View gets new private members
  - ge211::Text_sprite explanation_
  - ge211::Font sans28_

- Build output string in draw()
  - Create an Image_sprite::Builder
  - Set a font and a Color
  - Set the string to be displayed based on the character
  - Reconfigure the Image_sprite
  - Add the sprite so it appears

Upgrade characters to hold a list of destinations

- Probably want to use an std::queue
  - push() positions to the end of the queue
  - pop() positions from the front of the queue

- Change to the next destination after we reach it
  - Occurs in on_frame()

- Make sure the initial destination is the initial position
  - Or we'll start moving somewhere right away

Use mouse clicks to specify waypoints for a character

- Respond to mouse clicks in the Controller
  - Forward click to the model to act upon

- Model uses mouse click to add destination for first character

Outline

- Game Motion Planning

- **Animations**

General principle

- An animation is just multiple still pictures
  - That are moved through over time
  - Frame 1, then Frame 2, then Frame 3, …

- GE211 can animate in the same way
  - Hold multiple Image sprites
  - Choose which image sprite to display based on time

  - We can add animation to our existing “game” without too many modifications

Our animation source

https://www.deviantart.com/d-o-9-bessa/art/Eevee-Walk-Animated-806837776

Split into multiple frames

- We'll cycle through these in our game code to animate Eevee
  - I pulled the existing gif apart with ImageMagick, a command-line tool, and plenty of googling to figure out what the right commands were
  - You could also draw your own animations!

Updating our character class for animation

- Additional private members
  - Add a vector of image sprites
  - And an index within those image sprites

- Revise the constructor to be initialized with multiple filenames

- Revise the update() function to step through sprites

Further additions

- Only animate while moving
  - Track if there is a destination

- Flip the image horizontally based on travelling direction
  - Maybe even rotate image based on destination
    - atan2(unit_vector.width, unit_vector.height)*180/π gets angle
    - Need to subtract or add 180 if going left

- Different screens for game states
  - Start screen and Pause screen
  - Keep state in the model about which mode we are in (an enum class)
  - Add on_key to pause game
  - Adjust update, on_click, and draw to depend on state

Outline

- Game Motion Planning
- Animations