HW3: Air Race

CSE165 - Discussion 5

Agenda

- HW2 Recap
- Homework Overview
- Creating the Racetrack
- Leap Motion, Oculus, and Unity
- Flight Controls
- Intro to Wayfinding

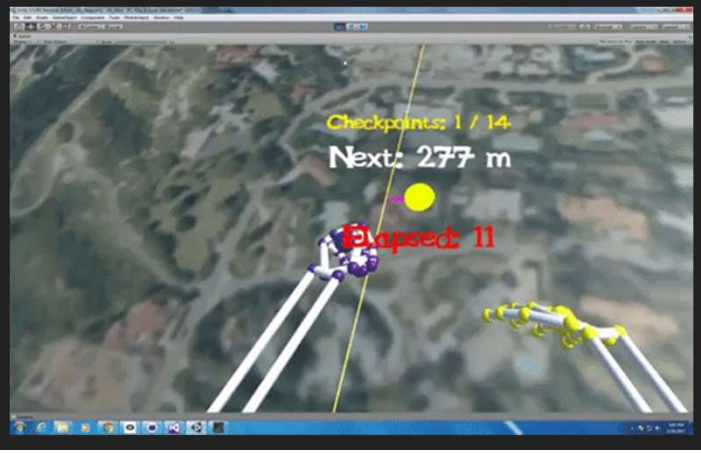
Next Week: Wayfinding, Gameplay, and Extra Credit

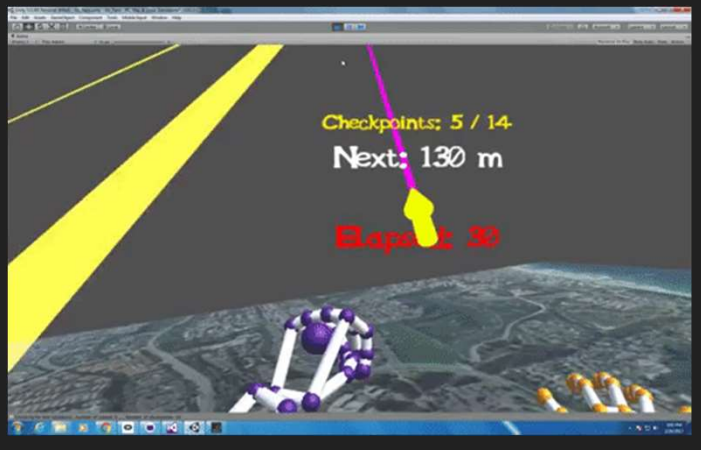
HW2 Recap

- Great job with HW2! It was much, much harder than HW1
- Remember: Much of this class is about interaction **design**
  - How can you make an interaction system that's easy to use?
  - Intuitive? Doesn't need much explaining? Not too complex? Few buttons?
  - If the tutors can't figure it out, how will your users?
  - Though we don't necessary grade your design, it's the whole point of the class!
- For all future assignments, consider your designs carefully
  - Try some user testing! Show your app to someone new! Take notes!

HW3: Air Race

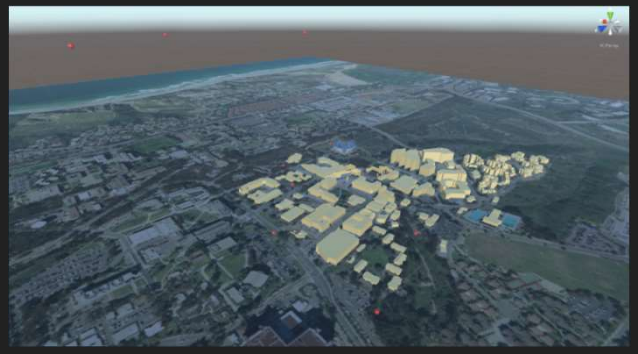
- Race through waypoints over the UCSD campus!
- Control your flight entirely with your hands - using Leap Motion!
- It really is a race - on grading day, you will be timed! (Extra credit for top 3!)
- Focus on Leap Motion interaction and wayfinding
  - Interaction: Flying the ship quickly and efficiently to reach the destination as fast as possible
  - Wayfinding: Helping the user find the next waypoint and determine where to go





Creating the Racetrack

- Load in the campus model. Remember: The unit size of the model is 1 inch
- Unity's unit size is 1 meter
  - Change the import settings of the model to scale it properly
- Parsing the checkpoints
  - StreamReader.ReadLine() and float.Parse()
  - Refer to the sample checkpoints and images to make sure it's right!
- You'll have to parse on the spot at grading, so test with different files!

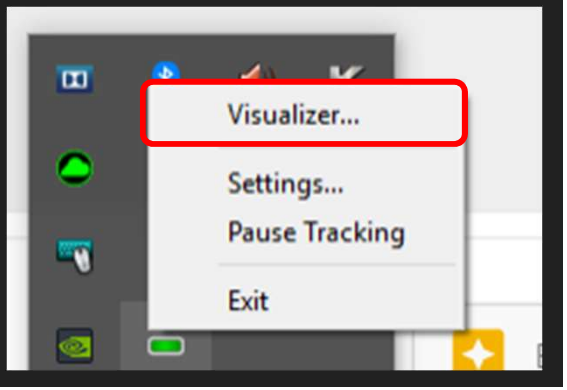


Flight Controls

- User should be able to control the aircraft using only Leap
- No autopilot or teleportation allowed! Ship must always be player-controlled
- User should be able to look around without affecting ship movement
- Create controls that are intuitive and easy to use
  - Avoid difficult hand motions or complex gestures
  - Get creative!
- Remember... the Leap Motion field of view is not that good
  - Keep the interactions centered around where you are looking

Leap Motion, Oculus, and Unity

- Attach your Leap to the front of Oculus and connect it using USB extender
  - Download Orion and run installer: https://developer.leapmotion.com/get-started
  - Make sure leap motion is connected to computer, and Oculus home is launched.
  - Find its icon in the status bar and click “Visualizer” to take a look at how leap motion works!
  - See detailed setup tutorial at: https://developer.leapmotion.com/vr-setup/oculusrift/
- LeapMotion SDK for Unity
  - Unity Core Assets for Orion Beta: https://developer.leapmotion.com/unity#116
  - It is a Unity package that you can directly import into your project!



Leap Motion, Oculus, and Unity

- Set up Leap in your Unity project:
  - Replace “MainCamera” with “LMHeadMountedRig” found in the Orion SDK
  - In “CenterEyeAnchor”, change Camera clear flags from “Solid Color” to “Skybox”
  - Make sure XR/VR supported is checked
  - Click “Play” and put your hand in front of your HMD.
  - Try bending your fingers and figuring out what gestures leap detects most accurately!
- For learning how to use this as input, documentation will be helpful!
  - Full Docs: https://github.com/leapmotion/UnityModules/wiki

Introduction to Wayfinding

- Create **two** mechanisms that helps the user **find** waypoints. Some ideas:
  - Arrows pointing to the next waypoint
  - A 3D compass that always points to the next waypoint
  - Lines to follow that lead to the next waypoint
  - A minimap that shows the player relative to the next waypoint
  - A distance or hot/cold mechanism that helps the user get closer to the waypoint
  - Come up with your own!
- Your technique should clearly make it easier to find waypoints
  - Try changing up the course track randomly, and see if you can still find waypoints easily!

Questions?

Feel free to ask on Piazza!

(Making your questions public is helpful to everyone!)