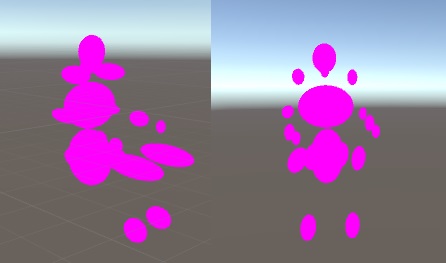
Unity Gesture Recognition Tool

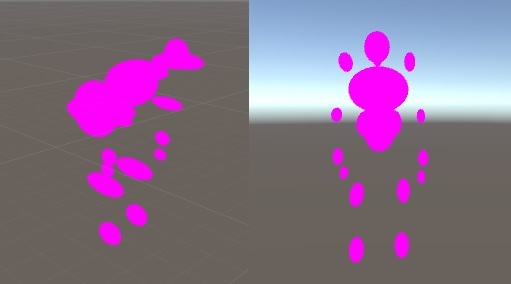
Project statement: The goal of this project is to create a tool within the Unity editor that is capable of heuristically detecting basic full-body gestures such as waving, shrugging, etc. using Microsoft Kinect skeleton data.

Project Breakdown:

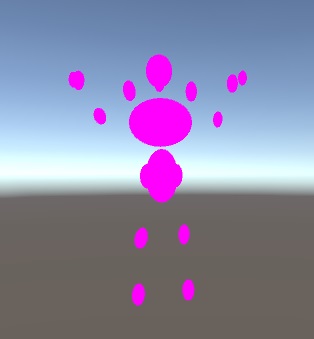
* Unity script for heuristic detection of gestures
  + Works with HUNA scripts to parse Kinect skeleton joint data
  + Runs alongside any other Unity program, can use the same skeleton data to detect gestures
  + Allows the user to select which, if any, gestures should be tracked by the program.
  + Using HNSkeleton joint data which updates every frame, check a set of parameters for each gesture, including relative positioning of joints in 3D space and the current velocity in the horizontal and vertical directions using frame-to-frame displacement of the most recent position of any joint.
  + Displays the status of every gesture currently being tracked in a UI Text element
* Gestures Detected:
  + Sitting



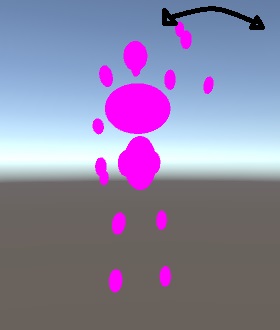
* + Bowing



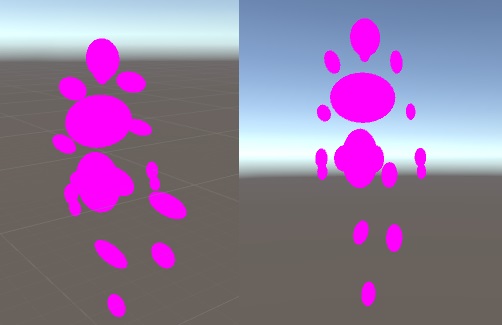
* + Shrugging



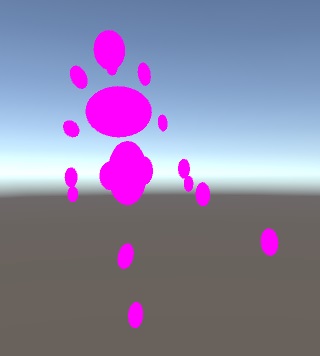
* + Wave Right/ Wave Left



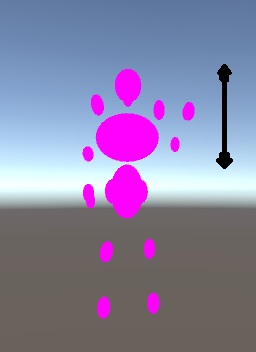
* + Step Right/ Step Left



* + Kick Right/ Kick Left



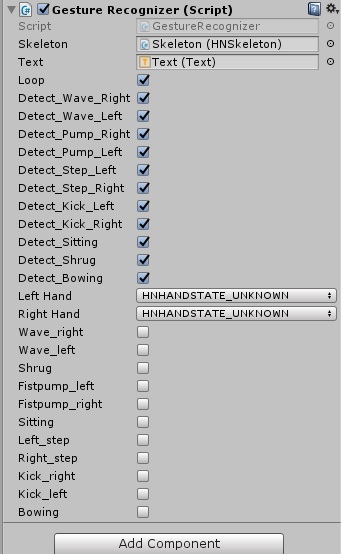
* + Fist Pump Right/ Fist Pump Left



* Other Scripts Used:
  + FileReader.cs/NetworkReader.cs – HUNA scripts which read either recorded skeleton data or the network port for the data
  + HNSkeleton.cs – HUNA script for parsing raw data into skeleton joint data
  + HNSkeletonRenderer.cs – HUNA script for displaying the skeleton data in the scene

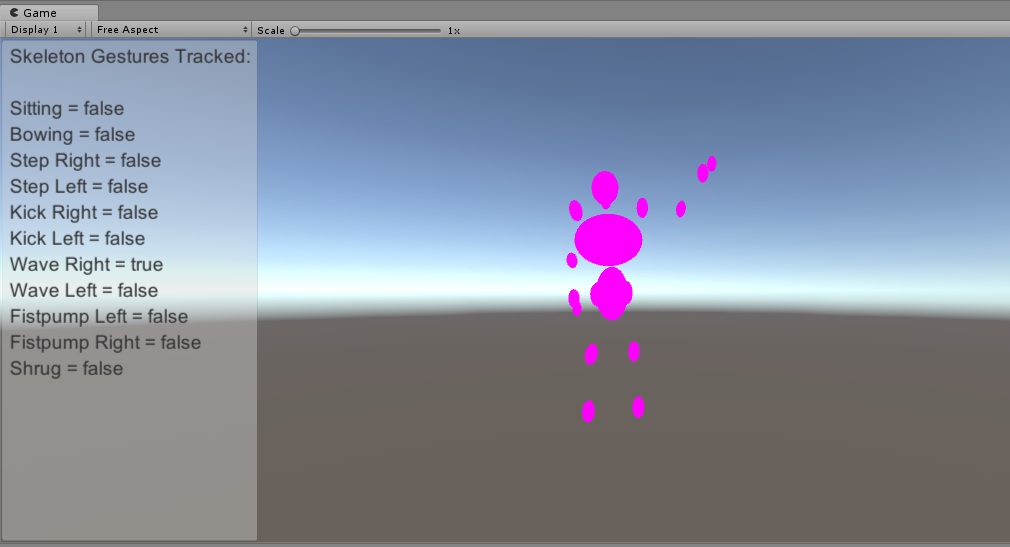
Instructions for use:

The main script is GestureRecognizer.cs, which can be placed in an empty game object and needs both a HNSkeleton object and a Text object to function. The HNSkeleton needs either a FileReader or NetworkReader to work. If you are you using recorded sensor data, use a file reader and point it to the sensor data. Otherwise, use a network reader and point it to the correct port.

GestureRecognizer calls a detection function and displays the state of any gesture that is currently being tracked. It is easy to change which is being tracked by simply checking a “Detect” box in the inspector. If a gesture is not being tracked, then the program does not spend any resources performing the calculations.

The Booleans for each gesture are public, so another script can easily read and use them.

In my scene, I created a UI panel to display the text on. This could be customized to any degree, and doesn’t need to be displayed at all.



Problems/Difficulties:

* Some of the skeleton data, especially the hand state data, is extremely unreliable. For gestures that rely on the hand being open I had to simply check that it wasn’t closed, and vice versa. The hand state fluctuates to ‘unknown’ or ’not tracked’ too much otherwise.
* The detected state of a gesture can change rapidly and it is possible to have a false positive, although if a false detection happens it should not last long. For example, when sitting down, the program might detect that a person is bowing while they bend down.