Andrew Moran
4/10/15
6.835: Project 4

1. All files in Stellar submission under andrewmo@mit.edu
   You can view the Battleship game under https://andrewmo.scripts.mit.edu, make sure you have a Leap Motion attached to your computer

2. There were some challenges while trying to complete this project.
   a. Initially, offsets needed to be calculated to ensure Leap Motion data was correctly being translated on the screen. In order to correctly grab the ships, I needed to create a screen x and y offset so my hand can precisely reach the lower and right sections of the screen. For ship deployment, I had to play around with rotation and mess with parameters to produce a better user experience. I decided to clip the `hand.roll()` values between 0 and -π/2 so it only transitions from a vertical to horizontal rotation. I also added an influence multiplier to `grab.strength()` so that the ship rotation happens faster. Releasing ships were difficult, however, can be mitigated by slow releases and no sporadic movements.
   b. On the player's turn, there were a few difficulties. As it is waiting for a player's speech to say 'start' and/or 'fire', recognition was not always clear/precise. Sometimes the player would have to say a word multiple times to register. There was a slight delay from where the player says an action and when it was registered so sometimes aiming at a desired tile would sometimes not register the correct tile of the player is moving to much.
   c. On the CPU's turn, I had most difficulty trying to correctly incorporate CPU personality based on player's response. Again if the player's response was mumbled and did not recognize the speech correctly, it could sometimes register a player's response as a lie even if it was truthful (and vice versa). In this section, I also noticed the speed of turn transitions, which I mention below.

   For some other notes, I noticed that the computer was also recognizing the long responses I told the CPU say in response to an action, especially when it was awaiting a player response. I tried to fix this with a variable keeping track of when the CPU is speaking in the `generateSpeech` function. I also played around with `recognition.onend`. This led to not having any speech being recognized so I decided to not dive into this deeper. Another note I would like to add is always having your hand above the leap in order for the game to progress (speech, turn transitions, etc.) This became tiresome over time. I again tried to mess around with delay constants to get around this.

3. I have made a few extensions on this project. As suggested, I added some personality to my Computer player. In the models.js script, I added a CPUPersonality model. I created cached sayings based on results of both the player's and CPU's actions. If an action is repeated, I make the statements more extreme. (e.g Missing – if a player misses a lot, CPU kind of taunts them. If the CPU misses a lot, it claims the player can't be far off). In addition, I added a lying component. If a player is caught lying in response to a CPU shot, the CPU recognizes it and let's the user know. Overall, this was a fun project!