\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

This code was used for the paper "Stealthy Behavior Simulations based on Cognitive Data" of Taeyeong Choi and Dr. Hyeon-Suk Na in 2015. Also, it was built on the previous work of Jonathan Treamblay at McGill University (<https://github.com/GameResearchAtMcGill/unitytool>), so many parts were done by him and reused in our work. The following is a direction to simply simulate player behaviors with Q-Learning with Artificial Neural Networks (ANNs), which was tested completely in Unity 4 (version 4.6.0f3). If you have any question, feel free to contact [ctyeong@gmail.com](mailto:ctyeong@gmail.com) . Thank you.

8. 15. 2015

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Open Unity, and click [File] - [Open Project]

2. Select this project folder.

3. If you cannot see the [Mapper] tab, Click [Window] - [Mapper].

4. In the [Project] tab, we can select the [Levels] folder and a level you want.

- After choosing a level, you can create or remove objects in the [Hierarchy] tab, modify their properties in the [Inspector] tab, and change map’s geometrical shape in [Scene] tab.

5. In the [Mapper] tab, you can see <1. Map>

- On the row of <Player Prefab>, click the dot (right side), and select the player.

- On the row of <Floor>, click the dot, and select the floor in the [scene] tab.

- Click <Edit Grid>, and click <Finishing Edit>.

6. Set appropriate <Time samples> in <3. Map Computation> of [Mapper].

7. Before training the ANNs, in [Mapper] you can set player's sight range (# of sensors) and initial direction, training epochs, initial and final temperature (action selection policy), and the area of the goal.

8. To train the networks, click <Train the Q-Function with ANN>.

- If you do not want to train ANNs and use previously trained ANNs, check the box of Import ANN, and skip to 9.

- If you want to export the trained ANNs into files after the training, you should first check the box of [Export ANN] before starting the training.

- If you want to see the trained paths for each epoch after the training, click [Compute Path] and insert the training number that you want to see in [path number to be shown]. After clicking <Activate the Player using ANN>, the trained paths cannot be shown.

9. After the training, click <Activate the Player using ANN> to simulate player's behaviors.

10. To see the simulation's result, click [Compute Path].

- In [5. Visualization], click [Play] to show the simulated results adjusting [Time].

11. While proceeding from 1 to 10, following files have been created in the project folder.

- collision\_info : the number of collisions for each epoch

- cumulative\_rewards : the accumulated rewards for each epoch

- goal\_visiting\_learning : the number of visits to the goal (target) point.

- player\_traces\_learning : the traces of inputs fed into ANNs for each training time

- training\_records : the collection of sensor data learned by the player agent

- trained\_agent/player\_traces : the traces of inputs fed into ANNs for each simulation time

- trained\_agent/ann\_dictionary, trained\_agent/ann\_network : the files exported from trained ANNs