Kevin Doyle, Shane Staret, Melissa Tjong, & Devyn Myers
CSCI 357: AI & Cognitive Science
Professor Dancy
17 April 2021

User Manual

Our goal is to create AI agents that will play a game of tag. To do this, we will have one AI agent act as a tagger, whose task is to chase and tag the other agent, and a runner agent, another AI agent whose task is to avoid the tagger. We will then place these agents in a contained environment with obstacles in order to have the agents maneuver around these obstacles and use them to their advantage. A successful outcome will involve the runner agent avoiding the tagger for as long as possible. Through our implementation, we also want to see whether the runner agent is truly aware of its task and learns any specific strategies for avoiding the tagger.

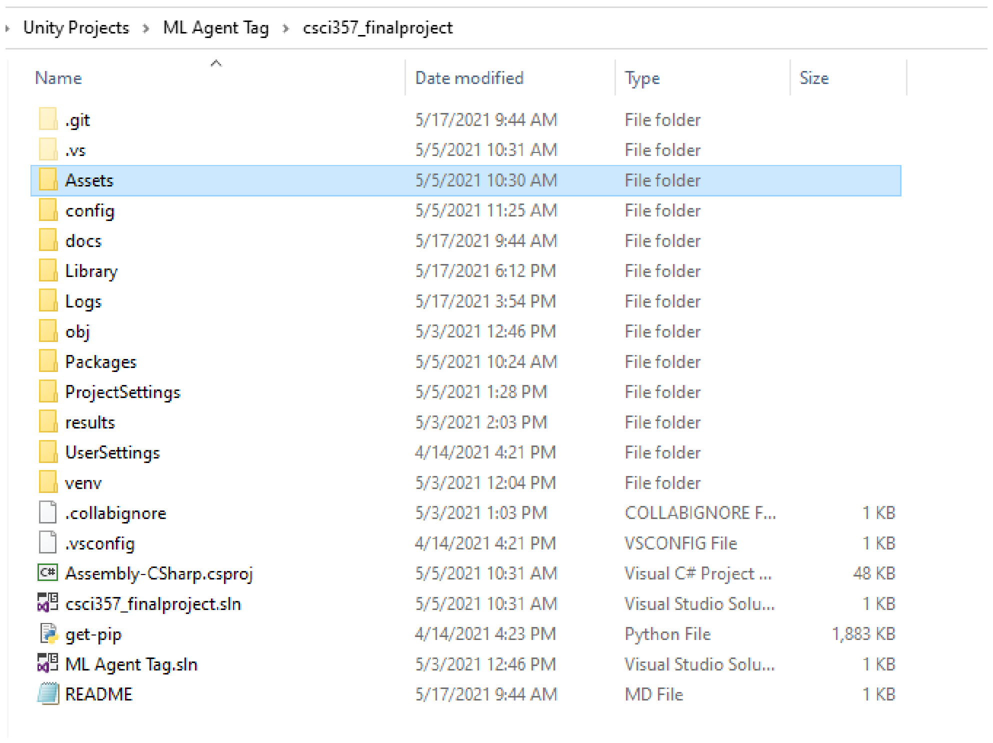
While there haven't been any notable instances of a game of tag being played between two AIs, there is a popular implementation of AI playing hide and seek by OpenAI. This example is similar to our own problem, mainly because both the hider and the runner avoid a seeker, or tagger, that is trying to find them. In OpenAI's example, the agents are also playing in an environment with obstacles. The obstacles in the hide and seek environment are movable, which allows the hiders to use them to their advantage. Although the obstacles in our environment aren't movable, we wanted to mimic the same approach of using obstacles to the advantage of the runner. The seekers in this example used a radar-like sensor to detect any objects nearby, which is a strategy that was later used in our system to avoid obstacles. The agents in OpenAI's hide and seek implementation were trained using reinforcement learning, where a reward of 1 was assigned to hiders if they avoided being found, and -1 if they are found by a seeker. We adopted a similar reward system in our implementation, where the runner was assigned a reward of 1 for avoiding the tagger and a punishment of -1 for being tagged. We wanted to adopt the strategy of reinforcement learning to have the agent learn multiple strategies over time to avoid the tagger, just as the hiders developed strategies to avoid the seekers.

The environment and agents are built in Unity v. 2020.3.1.1. The required Python packages are Pytorch, used for performing computations using data flow graphs, and the ML-Agents library created by Unity, which is used to create the machine learning agents in our system.

Build Instructions

Below are detailed instructions for building this project yourself. Some software required to build this project are not included in the gitlab repository for file size reasons, and some steps will rely on downloads from other sources.

1. **Install Unity 2020.3.1.1 on your local machine**
   a. **Downloads for specific Unity versions can be found here**
2. **Install Python version 3.7.8 on your machine**
   a. **Other versions of Python may also work, but this project was specifically built with Python 3.7.8**
3. **Create a new blank Unity project**
4. **This project requires installation of the ML-Agents Unity package. Install the ML-Agents package in your blank Unity project.**
   a. **This project was built with ML-Agents release 17, although later releases may still work**
   b. **A link to the ML-Agents github can be found here**
   c. **Instructions for setting up ML-Agents on your machine can be found here**
      i. **Ensure that you have set up the python environment as detailed in the ML-Agents documentation, as none of the python libraries are included on our gitlab repository**
5. **Once ML-Agents is installed in your blank Unity project, download the contents of this gitlab repository, and drag them into the folder for your Unity project.**
   a. **This will overwrite some files, allow them to be overwritten**
   b. **For safety purposes, close the blank Unity project before copying these files over**
   c. **An example of what a Unity project folder looks like can be seen below**



Unity Projects › ML Agent Tag › csci357_finalproject

Name	Date modified	Type	Size
.git	5/17/2021 9:44 AM	File folder	
.vs	5/5/2021 10:31 AM	File folder	
Assets	5/5/2021 10:30 AM	File folder	
config	5/5/2021 11:25 AM	File folder	
docs	5/17/2021 9:44 AM	File folder	
Library	5/17/2021 6:12 PM	File folder	
Logs	5/17/2021 3:54 PM	File folder	
obj	5/3/2021 12:46 PM	File folder	
Packages	5/5/2021 10:24 AM	File folder	
ProjectSettings	5/5/2021 1:28 PM	File folder	
results	5/3/2021 2:03 PM	File folder	
UserSettings	4/14/2021 4:21 PM	File folder	
venv	5/3/2021 12:04 PM	File folder	
.collabignore	5/3/2021 1:03 PM	COLLABIGNORE F...	1 KB
.vsconfig	4/14/2021 4:21 PM	VSCONFIG File	1 KB
Assembly-CSharp.csproj	5/5/2021 10:31 AM	Visual C# Project ...	48 KB
csci357_finalproject.sln	5/5/2021 10:31 AM	Visual Studio Solu...	1 KB
get-pip	4/14/2021 4:23 PM	Python File	1,883 KB
ML Agent Tag.sln	5/3/2021 12:46 PM	Visual Studio Solu...	1 KB
README	5/17/2021 9:44 AM	MD File	1 KB

6. **Upon opening the Unity project, all of this projects assets and scripts should now be visible in the asset browser**
7. **You may now run our training environment by entering your python virtual environment and entering the 'mlagents-learn --force' command**
   a. **Details about running an ML-Agents environment can be found here**