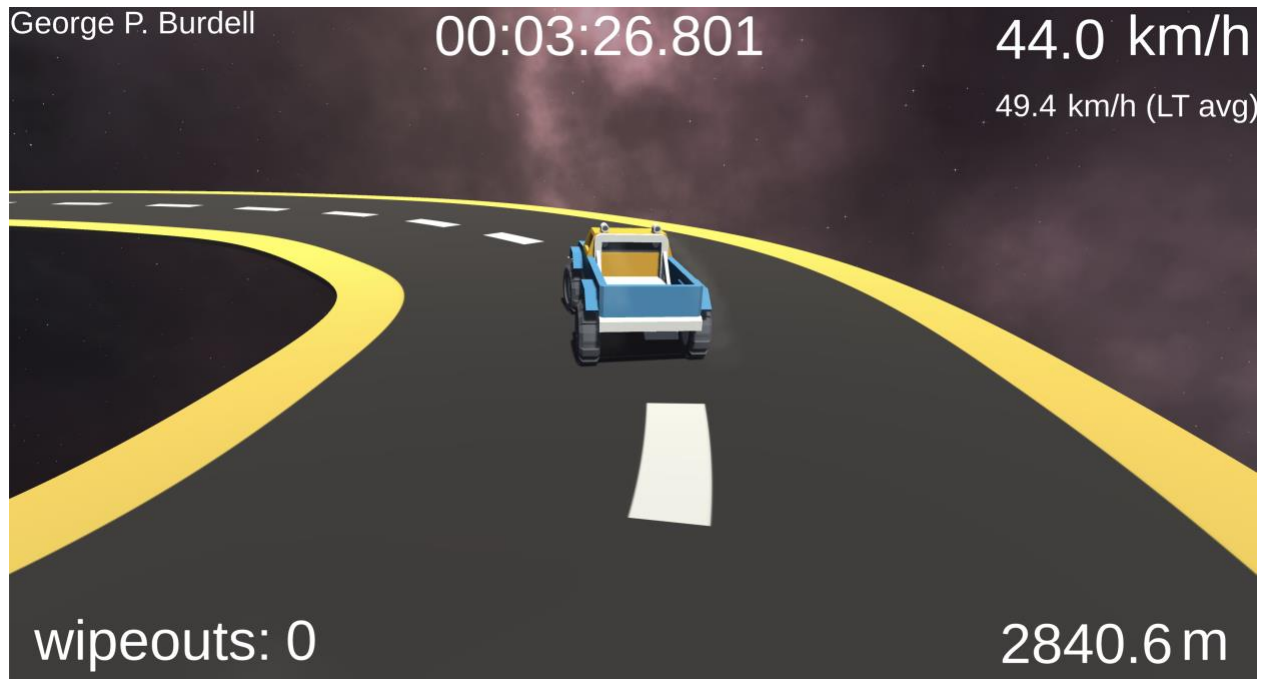**Race Track**

In this assignment you will be implementing a Fuzzy Logic Agent that can race on a procedurally generated track. You can also train a Neural Net agent to race on the same track.

Your agents will go as fast as they can without crashing, getting, stuck, turned around, etc.

The vehicle is a complex physically simulated truck provided by the Free Unity Asset Store Package: Arcade car physics by saarg. You can try driving it yourself if you open the /scenes/RaceTrackHuman. It is quite challenging with interesting dynamics.

Fuzzy Logic

A fuzzy logic library is provided for you to use. It was developed by t0chas on Github and is based on Buckland's *Programming Game AI by Example*. See https://github.com/t0chas/FuzzyLogic

Note that some minor changes have been made to the library.

You will use what you have learned from the Fuzzy Logic lecture to build Fuzzy Logic control of your racing truck.

Check out the Weapons example that is straight from the Buckland book. It is a great example to refer to designing your Fuzzy Logic driving agent. https://github.com/t0chas/FuzzyLogic/blob/master/UnitTest/WeaponsExample.cs

You need to develop metrics related to the vehicle state and the track that can be fuzzified, and interpreted by fuzzy rules and then defuzzified to crisp outputs (Throttle and Steering).

Neural Net

[NOTE: The Neural Net part of this project is not graded and is not required to be completed.]

You will use the ml-agents Unity Package to train a neural net to drive the racing truck.

Follow the installation instructions here to set up the tools that you will need:
https://docs.unity3d.com/Packages/com.unity.ml-agents@1.0/manual/index.html

Note that you need the dependencies that match **ML Agents Version 1.0.7 Unity Package**

You will use the ML Agents package and tools to train your agent to drive the truck effectively on the track.

Tips for the neural net are provided below, but it is recommended that you read through all the introductory documentation that Unity provides.

Vehicle

Your agent (either FuzzyVehicle.cs or NNVehicle.cs in Scripts/GameAIStudentWork/) extends a controller for the vehicle. You can directly set Throttle [-1, 1] (brake through full throttle) or Steering [-1, 1] (full left through full right). Also, you can access the Speed, transform position, and other settings. You should not use the hand brake, boost, or other advanced features, or change the default settings.

Procedurally Generated Track

The procedurally generated track is based on SebLague's Path-Creator, a Bézier curve tool for Unity. https://github.com/SebLague/Path-Creator
It has been heavily modified to support real time procedurally generated curves.

Your agent (either FuzzyVehicle.cs or NNVehicle.cs in GameAIStudentWork/) can access details about the track with the PathTracker. These are useful for generating crisp input values for the Fuzzy Logic or for Neural Net observations.

Useful PathTracker details:
pathTracker.distanceTravelled
  - The distance the car has traveled on the current segment (since it is always adding new segments and deleting old segments, this value changes relative to the current path)

pathTracker.totalDistanceTravelled
-    The overall distance the car has traveled regardless of the current path segments
pathTracker.closestPointOnPath
-    The Vector3 world position that is on the path closest to the car
pathTracker.closestPointDirectionOnPath
-    The curve tangent of the closest point on path
pathTracker.currentBezierSegmentIndex
-    The index into the array of Bézier segments that the car is closest to
pathTracker.currentClosestPathPointIndex
-    The index into the array of discretized linear segments that the car is closest to
pathTracker.maxPathDistance
-    The full length of the current segments
pathTracker.pathCreator.bezierPath
-    The curved path. This data is difficult to work with and is discretized in a linear format
     below
pathTracker.pathCreator.path
-    This is the discretized linear form of the path representing the Bézier path
pathTracker.pathCreator.path.GetPointAtDistance(v)
-    This method gets the world Vector3 point on the path a certain distance down the path,
     possibly offset from where the car is
pathTracker.pathCreator.path.GetDirectionAtDistance(v)
-    This method gets the tangent on the path a certain distance down the path, possibly
     offset from where the car is

Grading Criteria

Your Fuzzy Logic agent must achieve a 40 Km/h average speed on the track over 3 minutes from
the start with no more than one wipeout (fall, stuck, flip, turned around, etc.) for full credit.

You must change the HUD student name as well.

Your Neural Net agent will not be submitted but you can participate on Piazza regarding
training ideas, performance, etc.

Submission:

You will submit FuzzyVehicle.cs. Don't forget to change your name!

Machine Learning with ML Agents Tips:

What is Proximal Policy Optimization (PPO)?

https://openai.com/blog/openai-baselines-ppo/

Any guidelines to number of layers/nodes?

https://machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network/

Guidance on rewards:

https://github.com/Unity-Technologies/ml-agents/blob/main/docs/Learning-Environment-Design-Agents.md

Using an Environment Executable (faster training)

https://github.com/Unity-Technologies/ml-agents/blob/main/docs/Learning-Environment-Executable.md#using-an-environment-executable

Also helps to reduce graphics demands by setting windowed mode with low resolution

Might need to increase buffer size per: https://forum.unity.com/threads/num-envs-n-help-what-are-unity-instances.975918/

Training Using Concurrent Unity Instances (faster training)

https://github.com/Unity-Technologies/ml-agents/blob/main/docs/Training-ML-Agents.md#training-using-concurrent-unity-instances