CSE-4410 Programming Assignment 2 (Section 2: Due April 3rd; Section 1: Due April 4th)

**Part 1 (20 points) – Follow any Unity tutorial you find interesting, or you think will help with your video game. Submit the following:**

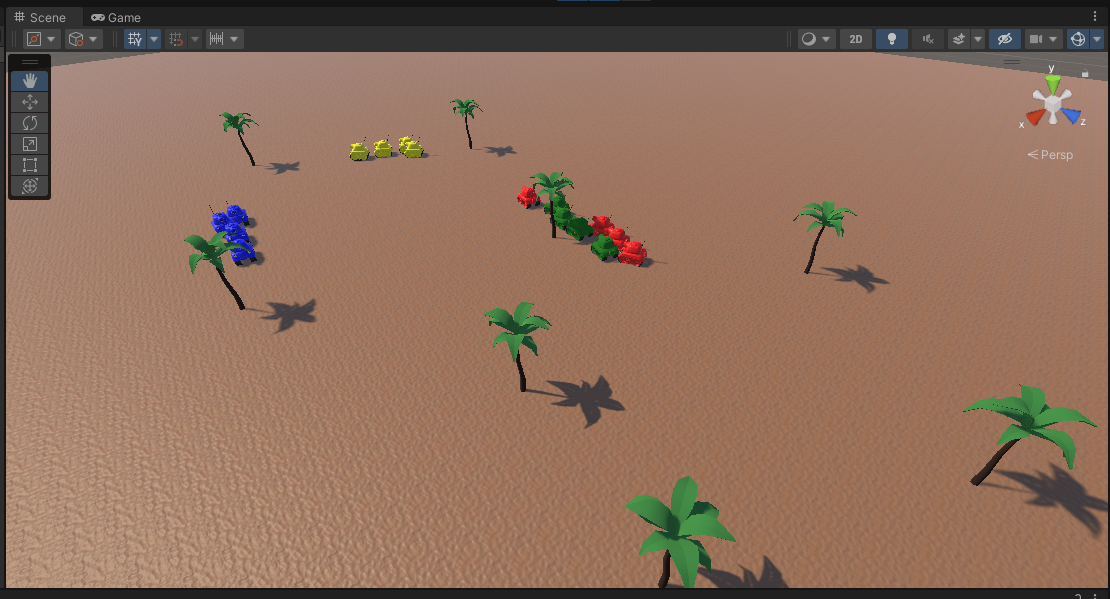
* **A description of what the tutorial is about (with a link to the video). Explain why you think it’s interesting.**

I followed the project titled “Waypoints and Graphs” found here: <https://learn.unity.com/project/waypoints-graphs?uv=2021.3&courseId=5dd851beedbc2a1bf7b72bed>

This is part of a larger course on Unity Learn titled “Artificial Intelligence for Beginners”. That course is quite extensive, so I selected the “Waypoints and Graphs” project because it dealt with a situation I was facing in developing patrolling enemies in our tank game for this class. The project showed us how to set waypoints and pass them to a script attached to a game object so to get it to move from waypoint to waypoint in a circuit. There were additional features covered, such as smoothing out the quaternion rotation between waypoints to give a more natural movement, as well as upgrading the logic for the game object to follow an invisible “perfect tracker” object in case it gets knocked off course. The second half of the project was very code intensive and dealt with coding a full A\* algorithm graph API for later use with a navigation mesh.

* **A screenshot of what you made.**

I’m adding a screenshot of the tanks in mid-race in case the embedded video below doesn’t work.



* **You may submit a video instead of screenshots.**

In this video, you can see 4 different types of tanks (denoted by color) with differing movement and rotation speeds. They have been given rigid bodies and colliders so they do knock each other off course. They are actually following an invisible “perfect tracker” game object instantiated for each tank that waits for them if it gets too far ahead. In this way, none of the tanks get stuck and they all end up being able to course correct and make their way around the waypoints (palm trees).



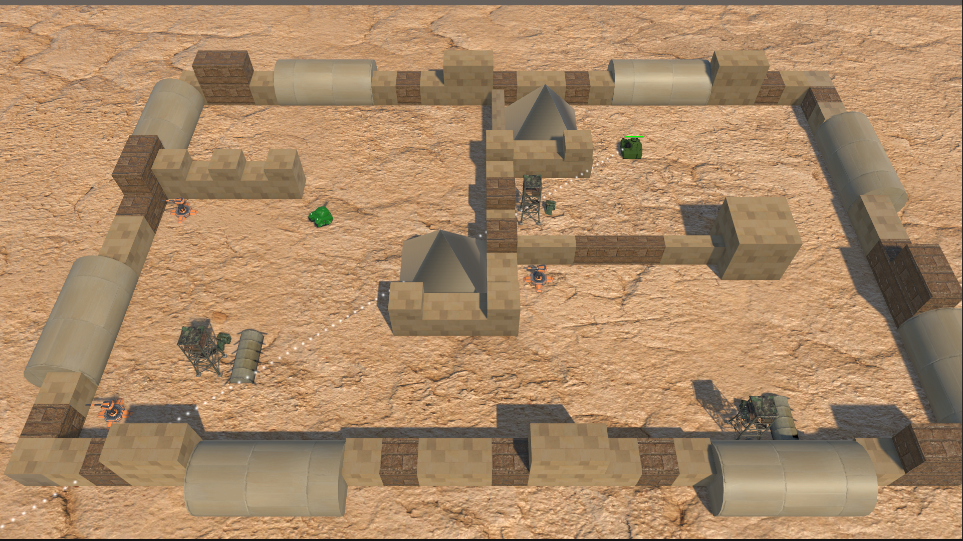
**Part 2 (30 points) – Using what you learned from the tutorial you watched, develop what you made so that it fits with the video game you’re developing.**

* **Explain why you think the video you watched will be helpful with what you’re developing. You can explain any differences you had to make from what was described in the video you watched.**

This tutorial was incredibly helpful as it helped me implement a patrolling tank enemy in our game. I had already implemented a stationary turret enemy that detected the player when in range and attacked accordingly. The patrolling tank enemy needed the same behavior, but to be travelling a predefined patrol route and stop and attack when the player is in range. The tutorial gave me the tools I needed to implement the waypoint/tracking system and I folded in the turret attack logic when the player is in range. I also liked the tank asset from the tutorial so I took that and modified it to be able to shoot tank shells. The result is a patrolling enemy tank that stops and fires at the player when in range, and continues patrolling if the player leaves the range.

* **Provide at least one screenshot of what you made.**

Adding a screenshot just in case this video embed doesn’t work, but it doesn’t really show the AI behavior implemented:



* **You may submit a video instead of screenshots.**

In this video, you can see the enemy patroller tank created with the AI logic learned in Part 1. It follows a predefined set of waypoints by focusing on a “perfect tracking” ghost object. The ghost object stops and waits for the tank when it gets too far ahead, like if the tank stops to shoot at the player. The patroller in the video follows its waypoint path until it gets in range of the player, where it stops, rotates toward the player, and starts shooting. When the player exits the range of the enemy patroller, it continues on with its waypoint route.

