**Jared White**

**IGME 202 Section 5**

**Autonomous Agents Documentation**

**Description of World:**

Upon entering the project, immediately one will notice several dozen spiders scurrying out from underneath the camera together. Immediately after that, several dozen fireflies will flock across the land from wherever they are to one specific area of the map. The world has a somewhat ominous tone, and takes place in a dense forest in the middle of a valley. The leaves of the trees are turning, indicating that it is fall/winter, and likely a cold environment. There is low lighting in the world, and the sky is an ominous dusk color. There is a flashlight pointing at whatever the player is looking at, as well.

At the bottom of the valley lies a small lake covered in rolling steam from the cold air. Several campfires are scattered across the hills and the lake beach – possibly made from other travelers trying to make their way through the woods – fireflies dance across the lake as they travel from fire to fire, attracted to the light. While the spiders scurry about and the fireflies dance, a horde of zombies will shortly emerge from the fog lumbering after the player’s brains.

**Steering Behaviors:** (Note: In my code, these behaviors are primarily found in the VehicleMovement and Flocker scripts.)

**Flocking:** Two flocks of spiders will flock together around the world with no motivation other than just being typical weird creepy-crawly bugs that run around without ever having a good reason to, and for the most part stay together as a cohesive group – until zombies start threatening them. When threatened, the flocking weights will drop as the spiders scramble away to safety.

**Path Following:** The fireflies follow a path from campfire to campfire, and dance around the fires along the way. My implementation of path following differs from the provided definitions of path following just a bit:

Paths can be marked as either proximity-based, or point-based. Proximity-based paths generally work the same as the provided definition. The path followers will follow the closest path segment that they are on at any given moment.

When a path is marked as point-based, an agent will not start seeking the next node in the path until the node they’re currently chasing after is reached, determined by a Node Proximity Radius value which is separate from the path radius – this distance can be larger or smaller than the path radius. As soon as this area is entered, the path follower will start going to the next point in the path.

Additionally, instead of seeking the next node in the path, path followers will align themselves to the same direction as the path segment they are on.

By marking the path as point-based and giving the path a relatively large path radius but an extremely small Node Proximity Radius, and by playing around with weighting and speed values - combined with the high separation weights - I was able to make it so that at each stop along the path, the fireflies would keep flying past the target and doubling back to reach the center of the path node, but they would often miss it and keep flying around in circles for a short time until they reached the point. This gives the illusion that the fireflies are dancing around the fires.

**Leader Following:** The zombie horde uses leader following – one large leader zombie leads all of the other zombies towards their next meal (which is the player).

**Characters’ response to other characters:**

Spiders will **flee** from the zombie horde when they get close to the spiders, as they’re scared of pretty much everything. (They were originally going to evade, but the zombies were so slow that they’d end up running straight into the zombies instead.)

The zombie leader will constantly use the **arrive** steering behavior to chase after the player, and in turn, lead the other zombies to the player.

Fireflies use strong a relatively strong **separation** force between each other to make them dance around the fires in the world (along with the changed path following algorithm).

All of the characters will at least make an attempt to **obstacle avoid** the trees of the forest.

**Resources used to guide steering algorithms:**

I did use Shiffman’s Nature of Code for some pointers on some of the algorithms – however, I mostly used class notes as a general guide for writing the algorithms, as Shiffman’s readings were often confusing and hard to follow and understand.

**World Exploration:**

The world uses a first-person camera to be explored. The lighting in the world is dark, and the player’s flashlight does not reach far. But, luckily, all of the behaviors to be examined can be examined without looking very far – one of the two spider flocks spawn beneath the player’s feet, so they can be examined doing their flocking behaviors immediately.

Additionally, the zombie horde is always chasing after the player, so unless they get stuck, the horde will always come to the player – you shouldn’t have to go out seeking the horde.

The fireflies can be seen from anywhere in the scene – they are each a particle system that glows and can be seen through the darkness, and they’re very obvious. When they first spawn, they spawn all over the entire map and will immediately be seen, and will find their way to the closest point on the closest path segment that they can find, then they will begin their dance.

**If you have difficulty finding the autonomous agents**, press the **“E” key** to toggle the mouse cursor and debug lines. This will turn on the blocks for each flock that represent the centroid of the flock, along with turning on the debug line that shows the average direction the flock is facing. This should help you find any flocks of agents.

**Other notes:**

Although not quite immediately obvious, there is a quiet music track looping in the background. (I did not want it to be extremely loud and annoying like that Christmas project that was shown to the class. Sorry to whoever made it, but it was annoying!)

The scene uses a modified and improved version of the TerrainGenerator and WaterBehavior scripts that were written for the Random Project at the beginning of the semester. The lake uses this script to create the lake’s water effect using randomly generated values and randomly generated offsets – each time the project is run, the lake will behave a different way. Some runs may have very calm water; some may have really choppy water.

Try entering the water! Check out the screen overlay effect that’s applied.

The trees in the world are randomly place, and will never be placed underwater. With that being said, there are a couple caveats: The trees can occasionally – but rarely – spawn on one of the campfires, and prevent the fireflies from completing their pathfollowing. (Although, if this does happen, they will light up the tree like a Christmas tree!)

Furthermore, while it is socially acceptable for the undead to walk underwater, I was unable to figure a way to get the spiders to avoid flocking underwater. We’ll just have to assume they’re some sort of mutated super-spider that can breathe underwater for now.

Additionally, I am aware that if you try hard enough, you can get the player to fall out of the world. Please don’t do this. Although, you are more than welcome to if you’d like to do so.

Finally, although proximity-based pathfollowing was written, it was not implemented. The SmoothFollow camera script provided to us in class is also in the project, but it, too, was not implemented.

**Asset Resources:**

Dead Grass Texture: <http://www.textures.com/download/grassdead0054/31990>

Sand Texture: <http://dpanoply.s3.amazonaws.com/blog/20-free-sand-and-water-textures-for-the-summer/Free-Sand-Textures-2.jpg>

Dirt Texture: <https://blenderartists.org/forum/attachment.php?attachmentid=272798&d=1385436435>

Water Texture: <http://www.textures.com/download/waterplain0012/9438?q=water>

Trees (only one model used): <http://u3d.as/kNy>

Campfire: <http://u3d.as/jff>

Rocks (unused): <http://u3d.as/8xf>

Music: <http://u3d.as/m79>

Spider: <http://u3d.as/9ze>

Zombie: <http://u3d.as/bFW>

Unity Standard Assets were used for fog particle systems (Dust Storm) and the first-person controller, as well as the Image Overlay screen effect that is used on the First Person Controller.

The fog screen overlay texture was created by myself.