Eardstapa

Hackathon One and Two Report

Jordan Underwood
000973993

CONTENTS

TABLE OF FIGURES

WHAT I CREATED DURING HACKATHON ONE

For the Hackathon I chiefly focused on creating a base script that the AI agents within Eardstapa use as a base for their movement, behaviour and their interactivity.

FIGURE 1 – THE TEST CHICKEN MODEL

The AI I created consists of four key states which dictate its behaviour, and animation. The four states are; Idle, Move, Run, Travelling. The AI will swap between the Idle and Move states based on a timer, unless interrupted by the Player.

When the AI is in idle state it will stand still and play an Idle animation. Whilst the AI is in Idle state it will also check its surroundings using a Sphere case for game objects tagged “Player”, or “Food”. If the Sphere cast returns a ray cast hit tagged “Player” the AI will then switch into its Run state and flee from the “Player”. However, if the Sphere cast returns “Food” the AI will then switch to its Move state, and travel to the “Food” game object, where it will re-enter its Idle state on arrival.

The Move and Run states act as intermediary states, as they’re simply used to determine the location of travel, and the speed at which the AI will travel, utilising the “directionAndLocation()” method to do so. Upon reaching its destination the AI will then re-enter the Idle state.

```
    public void directionAndLocation()
{
    //Sets the direction the chicken is facing
    moveDirection = new Vector3(0, Random.value * 360, Time.deltaTime * rotSpeed);
    //The trigger that rotates the chicken
    transform.Rotate(moveDirection);
    //Sets the location for the chicken to move to
    Vector3 newPos = roamLocation(transform.position, roamDistance, -1);
    //Moves the chicken to it's new location
    agent.SetDestination(newPos);
}
```

FIGURE 2 – DIRECTION & LOCATION CODE

The AI will enter its Travelling when the host agent is moving. Whilst in the Travelling state the moving walking animation will be applied, as well as the script frequently checking for loss of movement, in which case it will swap states into Idle.

Ultimately if the AI is uninterrupted it will loop between its Idle and Move states to replicate the grazing behaviour of natural fauna; in which the AI agent will wander around its local area in the Nav Mesh, and visually graze on the surrounding flora using the idle animation.

```
void UpdateState()
{
    if (time >= animationTime)
    {
        time = 0;
        SetAnimationTime();

        if (currentState == States.Idle)
        {
            currentState = States.Move;
        }
        else
        {
            currentState = States.Idle;
        }
    }
}
```

FIGURE 3 – UPDATE STATE CODE

As seen in Figure 4 below, the AI is in its Idle state, in which the Idle animation is playing, giving it the visual appearance of grazing on the grass below it.

FIGURE 4 – THE CHICKEN GRAZING IN ITS IDLE STATE

FIGURE 5 – THE CHICKEN WALKING IN ITS MOVE STATE

FEEDBACK RECEIVED AND CONCLUSION

The primary focus of the feedback that I received was the movement of the AI agents. Due to the AI agents beginning to move before they're facing the direction they're traveling in, it creates a gliding effect, which makes the movement visually look as if the AI is sliding rather than walking.

Another key area of feedback received is that the AI requires further refining, such as increasing the time between State changes. This is to create a more natural behaviour pattern for the AI rather than rapidly changing between animations. The speed at which the AIs move also needs to be slightly changed as the AI currently moves faster than the speed of its walking animation.

In conclusion, during the Hackathon I created the fundamental basis for all AI agents that will feature in Eardstapa. However, the behaviour and timing of the AI need to be greater refined to ensure its as natural looking as possible.

WHAT I CREATED DURING HACKATHON TWO

For Hackathon Two my first primary focus was implementing Advanced Lighting, and Post-Processing to create a Post-Impressionist inspired visual style.

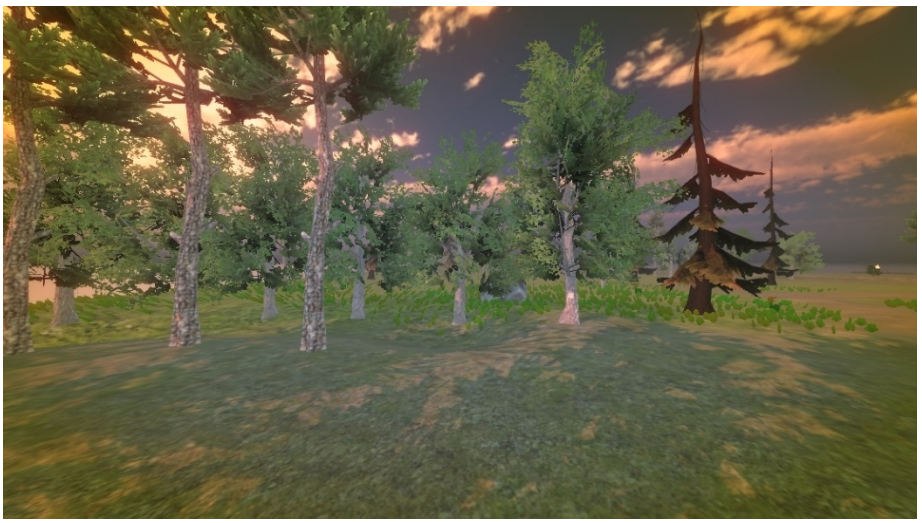


FIGURE 6 - EXAMPLE OF CREATED VISUAL STYLE

The first major implementation was using Real time global illumination, to create a realistic sunset, with improved shadowing and reflections.

I achieved this by adding a secondary directional light to the scene with an orange output and positioned it in such a way that it shines across the map from one side. This created long shadows across the scene, and a light red glow to the sky as would be visible in a real-life sun set. I created the visual effect of the sun, and its rays appearing through the trees by utilising point lights, spotlights, and Light probe groups.

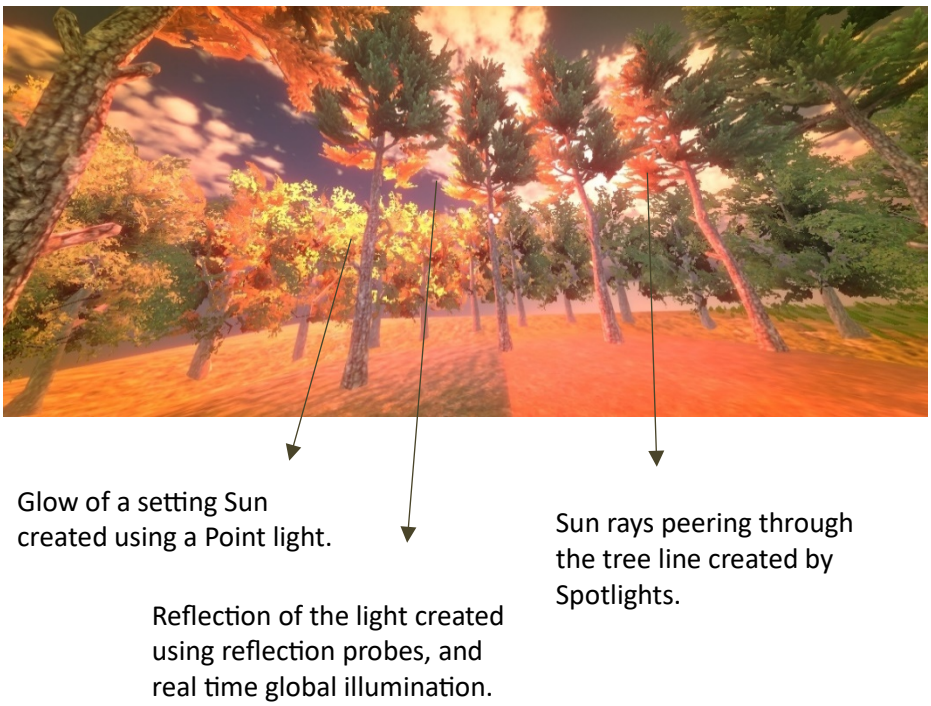
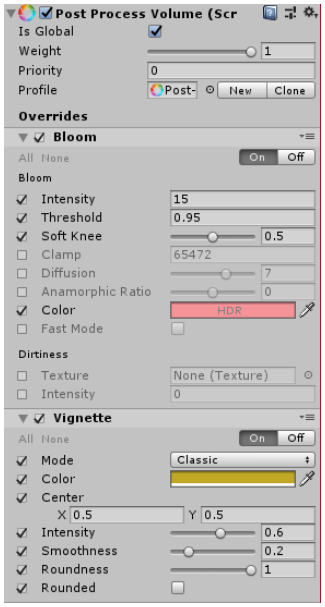


FIGURE 7 - EXAMPLE OF ADVANCED LIGHTING IMPLEMENTATION

The second key feature I implemented during the Hackathon was Post-Processing, using the Bloom and Vignette effects. By using two key effects, I implemented the Post-impressionist inspired visual style intended for Eardstapa.



I used Bloom to smooth and blend the colours within the scene. As well as to add an ambient red glow.

I used the Vignette effect to add a yellow glow to create a filter type effect to the game.

FIGURE 8 - POST PROCESSING EFFECTS

FIGURE 9 - GAME PRIOR TO POST-PROCESSING

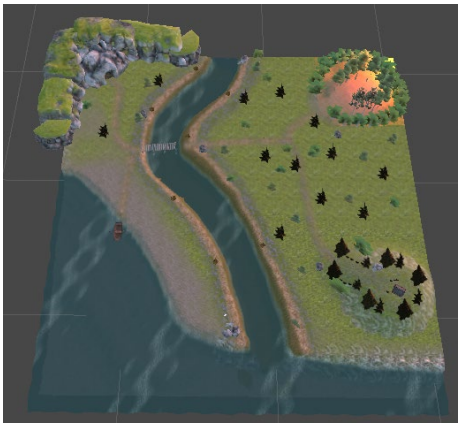
Yellowing of edges caused by Vignette.

Ambient red glow added by Bloom.

Smoothing of colours added by Bloom.

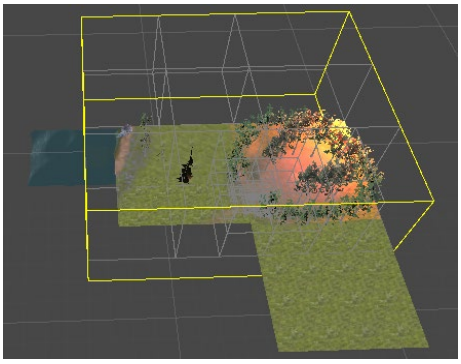
FIGURE 10 - GAME AFTER POST-PROCESSING ADDITION

The secondary focus during the Hackathon was improving the performance of my game by further optimisation. The key area of optimisation was the addition of Occlusion culling, by lessening the amount of rendering required in the game scene.



Game scene prior to Occlusion culling addition.

FIGURE 11 – GAME SCENE PRIOR TO OCCLUSION CULLING



Game scene after addition of Occlusion culling.

FIGURE 12 – GAME SCENE AFTER OCCLUSION CULLING IMPLEMENTATION

FEEDBACK RECEIVED AND CONCLUSION

The biggest piece of feedback I received regarding my progress in the Hackathon, was that while the Advanced lighting and Post-Processing created a good visual style, my game world lack presence.

A lack of presence within my game level subtracts from the players immersion, as the world seems flat, and unrealistic. To improve I must look to create a more natural game world, with improved depth and busier scenes with more game objects, such as rocks, bushes, and trees.

Another area of feedback received regarding my Occlusion culling was that it created inconsistent results, where areas which were not in the players point of view where still rendered.

To amend this issue, I must limit the width of the player cameras view and ensure that models leave no gaps in which the player can see game objects through, to prevent out of view rendering.

The other feedback I received was regarding my UI, and Teleportation mechanic. In which it was suggested that I use interactable objects rather than traditional UI features to better suit VR interaction.

It was also recommended that I further limit the Raycast of my teleport mechanic and add an offset on the teleport height to prevent the player was teleporting below the terrains mesh collider.

In conclusion during the second Hackathon I implemented Advanced lighting, post processing and basic optimisation. As well as creating the desired visual style for Eardstapa. However, these mechanics require further refining to ensure they’re implemented as effectively as possible.