Progress Report

- Increment 1 -

Group #3

1) Team Members

Grace Brill (FSU ID: KGB19A | Github ID: kbrill25)

Elias Diez (FSU ID: EAD19F | Github ID: EliasADU).

Jason Graham (FSU ID: JG19M | Github ID: JasonAGraham)

Miguel Malayto (FSU ID: MAM18N | Github ID: mizai125)

Santiago Ruiz (FSU ID: sr19w | Github ID: santiruz)

2) Project Title and Description

Project Title: Owl's Omen

Owl's Omen is a first-person 3D parkour game, which is being developed in Unity using the C# programming language. The gameplay revolves around the ability for users to launch off surfaces by clicking on them. Levels are located in one centralized area, with many interconnecting paths and routes. The user's objective is to complete delivery routes within a certain time constraint while being pursued by flying enemies that can track the user. The game environment is interactive, including elements like dialogue and surfaces that affect player mobility and speed.

3) Accomplishments and overall project status during this increment

Overall

We have made excellent progress on our project, due in part to starting as early as possible with prototyping the MVP and discussions on the game's design, as well as the team's willingness to learn Unity in a short notice.

Player Movement System

All major aspects of the movement system are implemented, thoroughly tested, and working. All that remains for it is fine tuning the variables (such as impulse strength) for perfect game-feel, something that will happen once the level design starts to come underway, as the two are closely linked.

In specific, all ground-based and air-based movement aspects are finished, with the only feature left being the momentum conservation when landing.

First Person Animation

Staff sway, staff angling, and staff spinning on aim are all finished although slightly jittery due to issues with syncing movement with framerate. To-do is fixing the jittery-ness as well as the view bobbing during ground movement.

Surfaces that affect player movement

In this increment, 3 surfaces that affect player movement were implemented: unidirectional speedboost, multidirectional slowdown, and a jump boost. All three features are fully functional and have corresponding animations to indicate how they affect player motion.These features represent the "Special Surfaces" portion of the project proposal. These features pretty much encompass what was included in the initial scope and functionality proposed. The next steps would be to refine these surfaces to make them more nuanced and to determine if there are any more surfaces that affect motion that should be added.

Visuals / FX

General visual assets imported and implemented such as the staff and skybox (which is itself imported from an Unreal project and manually modified to work with Unity). Visual effects added to staff and impulses.

Character development

In addition to the development of software, a robust set of characters is being written and defined. This step is necessary so that the world and characters can start being built and developed in Unity and C#. The following characters have been created and thoroughly developed: Nim (Main character), Mazikeen (Rebellion Leader), Tamsin (Rebellion Leader), Mulciber (City Watch Commander), Ayaz (Rebellion Member), Claude (Circus Owner), Ivan (Debt Collector), Victor (Rich Citizen 1), Florence (Rich Citizen 2), Miriam (City Watch Official 1), and Jakab (City Watch Official 2). All characters will be implemented in future increments, but the most important will be the main character, rebellion leaders, rebellion members, and city watch commanders. The other ones will serve a minor role, mainly for decorative purposes and casual dialogue. The character development is connected to the "Visual novel-style storytelling" portion of the project proposal.

Dialogue system

Currently, the dialogue system is in its early stages of development. So far, there is a basic template with a Canvas. The Canvas includes all UI components, which so far are an image, which is a white rectangle where the text is shown, and the text. There are multiple scripts that allow custom text from a Dialogue GameObject to be written to the image. One interesting feature is that the text is written in the typewriter style, meaning that letters appear one-by-one, which is common in many games. The dialogue system is connected to the "Visual novel-style storytelling" portion of the project proposal. There was nothing explicitly written in the project proposal about a dialogue system, so this system goes beyond what was written in the proposal.

Enemy AI

An enemy AI that will be used to chase the main character, it will be able to essentially track and patrol to find the main player. Using Unity's physics the enemy will be able to chase the main player through different heights and areas, but also has a vision cone that allows it to only chase the player when the player is within its vision. If the player gets too far from the enemy the enemy will go back to patrolling. The enemy is a vital factor in the main game because it gives the game a goal. Having the enemy functioning and reacting the way we want is an integral part of the game.

Main Menu

A main menu has been added to the game and will be the first Unity scene the player is launched into. It has the option for new game, load game, settings, and quit. The new game button launches our test scene and removes the main menu scene to allow the user to start playing. The load game button is a work in progress and by the next increment will save the players information. The settings button does not open a new scene but opens a settings panel that lets users modify resolution, volume, graphics, and toggle fullscreen.

Pause Menu

The pause menu pauses the game when the user hits the escape button. Allows the user to select from resume, save game, load game, main menu, and quit. Resume starts the game again and removes the menu. Save and load have not been fully implemented yet and the main menu button returns to the main menu scene. The quit button like on the main menu quits the application.

HUD

A heads-up display (HUD) was implemented to give the player visual tools that aid gameplay experience. Contains elements such as a jump boost cooldown meter and impulse tracker to allow the player to keep track of their resources. Also includes a minimap and distance-to-goal indicator for ease of level traversal.

4) Challenges, changes in the plan and scope of the project and things that went wrong during this increment

One challenge that was encountered was creating a surface that slowed down the user in multiple directions. Originally, the way the code was designed, it was flawed and would only slow the user down if the user was moving forward. This issue was resolved through teamwork. To resolve this issue, the velocity of the CharacterController was normalized and a negative impulse was applied to the normalized velocity. Ultimately, this solution fixed the issue of the slow down not being multidirectional.

Another challenge that the team faced was when one of our features we were implementing was affected by another member's code. This happened when working on the pause menu inside the game. When pausing time for the menu the player's camera kept moving while the whole game was paused. The player movement and the pause menu were made by two different members so with quick and concise communication we solved this issue by implementing a new method in the player camera script.

An additional challenge that the team faced was in designing the enemy AI. We had to go through many different phases of design because we weren’t able to use a NavMeshAgent which takes care of the physics and tracking of a player within the game engine. Without this we had to create our own implementation of player tracking which isn’t trivial to do. Also creating patrols and making sure the enemy can go around obstacles, has been a very big challenge.

A challenge that was also faced was when implementing the goal indicator. The indicator appeared twice in the scene due to the properties of WorldToScreenPoint, appearing in front of (where the goal's position is located) and behind the camera. This was resolved by making it so that the indicator only appears when the z-axis is greater than 0, since a negative z-axis indicates that the object is behind the camera. Also, figuring out how to clamp the indicator to the sides of the screen instead of moving off screen when looking away from the goal's position has been a challenge that will be resolved in the next increment.

Elias:

A challenge I faced when implementing the movement system was that of creating a realistic enough air-momentum / air friction system that was also fun to use. In doing this, I resorted to asking a physics major friend for help figuring out a formula to get the current velocity due to a force with X strength applied to, Y seconds from current time, having this, I can keep a list of all impulses still affecting the player's movement and aggregate their resulting velocities every frame, then applying them to the player.

Another challenge the subtle smoothing of the player's directional movement, as by default the input system switches from a flat 0 velocity to a flat x velocity whenever the directional keys are pressed, however, this transition can be smoothed out by just a small amount to give a less robotic feeling movement.

It was also challenging to get the first person staff movement working correctly, since it is rotating due to the player's turning or the player aiming (spinning it) at any given time, there are two rotations that need to be mixed together every frame. This was fixed by making the parent object of the staff rotate due to the player's rotation, and the staff to rotate itself for spins, thus letting Unity handle the mixing of the rotations.

Another challenge was getting the animated skybox working, as I wanted to use one originally made for the Unreal Engine, I downloaded Unreal to download the skybox into it, then took it out and (with the creator's permission) exported it into Unity. I had to get a special type of shader working for it that would take a "Flowmap" image and use it to animate the static image, this was nontrivial and required a lot of digging through shaders that I had never used before. Ultimately, I found a handy shader that required some minor modifications to work with the skybox (as the skybox is an image projected onto a dome with its normals flipped, and the shader was for a flat image).

5) Team Member Contribution for this increment

Grace Brill

a. Contributed to project title and description section (first paragraph), accomplishments (portion about surfaces that affect player movement, character development, and dialogue system), and challenges sections (portion about multidirectional slow down region).
b. Contributed to the overview section (first paragraph), functional requirements (speed boost surface, slow down surface, jump boost surface, managing player systems, and dialogue), operating environment, and assumptions and dependencies.

c. Contributed to the programming languages and execution-based functional testing (speed boost surface, slow down surface, jump boost surface, managing player impulses, and dialogue) sections.
d. Source Code
    i. Scripts (Entirely/Majority Contributor): Response.cs, DialogueUI.cs, DialogueObject.cs, ResponseHandler.cs, TypeWriterEffect.cs, Superjump.cs, and SlowdownCollisionCheck.cs.
    ii. Scripts (Minor Contributor): PlayerController.cs (contributed portions related to the cooldown period) feature.
    iii. Slowdown Region Visual
        1. Red pulsing region to demonstrate the region where a player receives a multidimensional slow down effect.
    iv. Superjump Region Visual
        1. White box with purple lightning shooting out to demonstrate the region where a player who is already engaged in a jump receives an additional jump boost.
    v. Dialogue Scene
        1. Located under the grace folder in Unity as the Dialogue scene.
        2. Canvas
            a. Contains an image, which shows text that represents dialogue.
            b. Contains a button on the top right corner (light mint green color) that the user can click to respond to a dialogue prompt.
            c. After the user clicks the button, the dialogue from the prompt they show appears on the white canvas image.
    vi. Dialogue and Plot Development
        1. Wrote character descriptions in a Google Doc in order to have a set list of characters to implement in Unity during the next increment.
        2. Important to have these characters written down so they can be developed with respect to dialogue that they may be associated with in the game.
e. Recorded video showing what the surfaces that affect player motion do and the overview of the dialogue system for the game. Link: https://youtu.be/-U1NHUIMTQs

Santiago Ruiz

a. Contributed to the progress report with accomplishments (portion about the enemy), challenges sections (portion about changing another member's code), and plans for next increment (Enemy).
b. Contributed to the functional requirements (Enemy).
c. Contributed to the programming languages and execution-based functional testing (Enemy)
d. Worked on the Enemy AI of the project. I started from scratch on the enemy's functioning chase mechanic. I was able to design an AI that would essentially chase the main player, around and over objects. I also was able to give the enemy an AI model and animation to make the overall aspect of it more aesthetically pleasing. I also made the enemy AI able to patrol within the map. When placing patrol points throughout the map the enemy will choose a random patrol point to fly to and I also created a vision cone within the enemy. The enemy will not be constantly chasing the player, unless the enemy is within vision of the enemy. The enemy will then see and chase,

but if the enemy loses sight of the player then the enemy will go back to patrolling with what it was before. Enemy.cs script and Enemy prefab.

e. Recorded video showing the enemy tracking and patrolling. Link: https://youtu.be/AC25hATps2A

Elias Diez

a. Wrote the Overall, Movement System, and First Person Animation section. Wrote everything under "Elias" in challenges.
b. Sections 1, 2, 3 of functional requirements. FPS requirement and responsive inputs in nonfunctional requirements. Odin and Octave 3D in dependencies.
c. HLSL and Shaderlab mentioned in Programming Languages. Odin and Octave 3D explanations in the second section. Movement system, first person animation, and general VFX in the functional testing section. Framerate testing for non-functional testing section.
d. Source Code:
    i. Scripts written:
        1. PlayerController.cs, AimParticleHandler.cs, DestroyPointer.cs, FPObjectStateSwitcher.cs, Impulse.cs, ImpulseCharges.cs, Jump.cs, KnobBehavior.cs, KnobManager.cs, Lockscreen.cs, LungeMeterManager.cs, SpeedBoostCollisionCheck.cs, MarkerParticlesBehavior.cs, RecoveryMeter.cs, SurfaceImpulser.cs, AimReticleBehavior.cs, FPObjectWeights.cs, FPStaffSpin.cs.
    ii. Skybox Implementation
        1. Animated clouds with black-hole parting them, pulled from an Unreal Engine package and modified to work with Unity (with author's permission).
    iii. VFX created:
        1. Impulse VFX, Staff spin VFX
    iv. First person animations created:
        1. Staff spin animation, staff "weight" animation on movement.
    v. GUI created:
        1. Charge counter GUI, Leap meter GUI, Aiming Reticle GUI
    vi. Speedboost Region Visual
        1. Bluish white spinning ring to demonstrate the region where a player receives a unidirectional speed boost effect.
    vii. Wrote dialogue for introduction / tutorial start, story writing for major plot points / character arcs, wrote main character, Nim.
    viii. Wrote all worldbuilding for the game's setting and lore.
e. https://www.youtube.com/watch?v=7z8FadB1nP4

Jason Graham

a. Contributed to the progress report with accomplishments (portion about the main menu and pause menu), challenges sections (portion about changing another member's code), and plans for next increment (Save, transitions, and audio).
b. Contributed to the functional requirements (main menu, settings menu, and pause menu).

c. Contributed to the platforms, APIs, Databases, and other technologies used (TextMeshPro) and execution-based functional testing (main menu, settings menu, and pause menu) sections.

d. Source Code

   i. Created MainMenu scene which include the main menu script, and settings menu script.

   ii. Created the pause menu panel that includes the script pause menu and modified the player script mouse look.

   iii. Created scripts for data, save, and player for save and load features not yet implemented.

e. Video presentation: https://youtu.be/dSUB_bgWyRk

Miguel Malayto

a. Contributed to the progress report with accomplishments (portion about the HUD), challenges sections (portion about WorldToScreenPoint and icon clamping), and plans for next increment (HUD).
b. Contributed to the functional requirements (HUD elements).
c. Contributed to the execution-based functional testing (minimap and goal indicator) sections.
d. Source Code

   i. Created testing scene which includes minimap objects (camera, border, mask), minimap icons, canvas objects (goal distance), and a goal object.

   ii. Created new layer (Minimap) for only the minimap camera to render.

   iii. Scripts written: Minimap.cs, InsideMinimap.cs, DistanceToGoal.cs.

e. Recorded video showing minimap and goal indicator functionality: https://youtu.be/6ryNcjCgklU

6) Plans for the next increment

Characters

Begin creating characters in Unity based on the character and plot development Google Doc. Planning to create certain dialogues for each character and develop potential conversations that the player can have with each character to make the game more interactive.

Movement System

Implement the momentum conservation when landing mechanics, polish other mechanics with fine tuning.

First person animations

Implement viewbob and effects during high speed.

VFX

Diegetic indicators for charge/leap on the staff through particle vfx.

Main level

First draft of greybox for main level.

Dialogue System

Continue to develop and refine the dialogue system in the game. This will be largely intertwined with the goal to implement characters to the game, since each character will say specific things according to their personality. Looking to fix a few UI issues with the current dialogue system and make it feel more natural.

Enemy

In this next increment the enemy will be able to actually do damage to the player. The player as of now is invincible, but in the future the enemy will do periodic damage to remove health from the player. We will also add different effects and animations for the enemy to make it look more aesthetically pleasing. Will also work on player tracking because as of now the enemy can go through walls and we want to make sure that the enemy will take the most efficient route when chasing the enemy.

Save

Have the save and load buttons in both the main menu and pause menu implemented and save basic things like the player's level, health, position, and cooldown status. This is partially implemented but will be working by the end of the next increment.

Load Transitions

Make transitions between scenes and menus smoother. Right now the Main Menu goes straight into gameplay and when loading it exits gameplay instantly. We want to make this transition smoother with a short loading type screen.

Audio

Start implementing basic sound effects in menus and in game as well as music in menus and in game. This will start this increment but I do not believe it will be finished due to more things being added later. We just want to get sound effects in music in menus and for the players actions which are mostly refined now.

HUD

Resolve goal indicator screen clamping by creating a 2D HUD "screen" to clamp to, and also add more visual elements to the indicator if needed. The next increment will focus on improving existing HUD elements and adding new elements when needed (such as health bars once the health system has been developed).

7) Link to video

Grace Brill Video Walkthrough: https://youtu.be/-U1NHUIMTQs

Santiago Video Walkthrough: https://youtu.be/AC25hATps2A

Jason Graham Video Walkthrough: https://youtu.be/dSUB_bgWyRk

Elias Diez Video Walkthrough: https://www.youtube.com/watch?v=7z8FadB1nP4

Miguel Malayto Video Walkthrough: https://youtu.be/6ryNcjCgklU