CMP302 - Game mechanics development project report

Dannielle smith 2101323

Terminology

“drawing”, the act of pulling a bow back in preparation of it being fired.

“arcing”, move with a curving trajectory.

“stamina”, is a resource that is held by the player, dashing, and drawing the bow costs stamina. the ability to sustain prolonged physical effort.

“health”,

Summary

A bow mechanic that allows the user to fire arrows in a manner inspired by a popular RPG game, Monster Hunter: Rise (Capcom, 2021) and other games from the franchise. The system is made up of several components:

1. Multiple abilities/attack that can be used. Including:
    a. ‘Arcing arrow’, an ability that shoots a projectile in an arc to produce an area of effect on a targeted location.
    b. ‘Default arrow’, an ability to shoot a projectile to cause damage to an enemy, damage will vary depending on how long the bow has been drawn.
    c. ‘Special attack/arrow’,
2. Stamina system, where:
    a. Stamina is constantly depleted whilst bow is drawn or if dash# is activated, if fully depleted player can no longer draw for a set amount of time.
    b. Player Stamina will be regenerated over time.
3. Health system, where:
    a. Stamina is constantly depleted whilst bow is drawn or if dash
    b. Player Stamina will be regenerated over time.

A video demonstrating these systems is available here: https://youtu.be/WSQasH05COw

Requirement Specification

Introduction

Purpose

Many games have an implementation of a spellcasting or ability system. The purpose of this project was to build a bare-bones and easily extensible implementation of the same, or similar, mechanics to investigate how that mechanic could be structured in code, and improved upon in a later implementation.

The project is a basic framework and beginnings of a spell and ability system for any RPG, MMO or ability-driven game. The components delivered include a spell selection system, casting system, cooldown system and a test area for spells and their effects. The product is designed to be easily extensible, so it can be built upon if desired.

///

This project is the implementation of a bow mechanic build into the player. It allows the use of different range attacks that could be reused or further built upon later.

Overall Description

Product Perspective

The primary aim of this project was to deliver a working range weapon (bow) system, where each attack/ability has a unique behaviour#. Other parts of the product, such as the provided mannequins were provided so that these abilities can be appropriately tested.

Product Functions

1. Allow the user to choose between multiple ranged abilities.
2. Displays the player heath and stamina to the user.
3. Allow the user to test the abilities in a enclosed environment with mannequins
4. Give a 3rd party the ability to use from already created abilities to as a base for different ranged weapons if so desired, or to further develop the current player skill set.

User Classes & Characteristics

This project is made for those familiar with the unreal 5 engine, and can be used as is or be added on and improved upon for future use. It is up to those who chose how, why, and where this project is implemented and can be customized to fit the developer's needs.

Design & Implementation Constraints

Since there is no artist assigned to the project, the developer sourced a free player model from the unreal marketplace for the player character to use, along with the arrow provided with this pack. (Find official pack name)

System Features

Ability: Default arrow attack/spawn

Description:

The basic attack the player can use. The bow must first be drawn to be able to trigger the attack, a crosshair will appear when the bow is drawn. Once triggered an arrow will spawn at a default speed and be at the targeted location //look up if gravity is affected//, the damage done to the intended

target will be determined by how long the bow is drawn for. After hitting a target, the arrow will also be ‘stuck’ to the hit surface and disappear after a set period of time.

Priority: 8

Stimulus / response sequences:

For gameplay, this player ability is accessed via combination on the mouse.

For development, the ability blueprint will open the UE5 blueprint editor - and the C++ code will open Visual Studio.

Functional requirements:

REQ-1: Bow drawn.

This is triggered by an appropriate player input.

REQ-2: Fire input

There must be “fire” or “trigger” input by the player for the arrow to be spawned.

REQ-3: Enough stamina

The player must have an appropriate amount of stamina to keep the bow drawn, if the stamina bar reaches zero the player is unable to draw the bow until a set amount of time.

Ability: Arcing/AOA arrow attack

Description:

A ranged attack the player can use. The bow must first be drawn to be able to trigger the attack, a crosshair will appear when the bow is drawn. Then when arcing is activated a projectile line will appear and the cross hair is removed, this line will show the user the trajectory of the bow and the area it will affect. Once triggered an arrow will spawn and move to the targeted location, the damage done to the intended area will set. The arrow will also be ‘stuck’ to the hit surface and disappear after a set period.

Priority: 8//

Stimulus / response sequences:

For gameplay, this player ability is accessed via combination on the mouse.

For development, the ability blueprint will open the UE5 blueprint editor - and the C++ code will open Visual Studio.

Functional requirements:

REQ-1: Bow drawn.

This is triggered by an appropriate key bind.

REQ-1: Arcing is activated.

This is triggered by an appropriate key bind.

REQ-2: Fire input

There must be “fire” or “trigger” input by the player for the arrow to be spawned.

REQ-3: Enough stamina

The player must have an appropriate amount of stamina to keep the bow drawn, if the stamina bar reaches zero the player is unable to draw the bow until a set amount of time.

Ability: Special arrow attack/spawn

Description:

A ranged attack the player can use. The bow must first be drawn to be able to trigger the attack, a crosshair will appear when the bow is drawn. Once triggered an animation will play, the player character cannot move or trigger another attack whilst the animation is ongoing, on completion an arrow will be spawn at a default speed and move towards targeted location, the damage done to the intended target will be determined by how long the bow is drawn for, this will also be accompanied by a multiplier. After hitting a target, the arrow will also be 'stuck' to the hit surface and disappear after a set period of time.

Priority: 8

Stimulus / response sequences:

For gameplay, this player ability is accessed via combination on the mouse.

For development, the ability blueprint will open the UE5 blueprint editor - and the C++ code will open in Visual Studio.

Functional requirements:

REQ-1: Bow drawn.

This is triggered by an appropriate key bind.

REQ-2: Fire input

There must be "fire" or "trigger" input by the player for the arrow to be spawned.

REQ-3: Enough stamina

The player must have an appropriate amount of stamina to keep the bow drawn, if the stamina bar reaches zero the player is unable to draw the bow until a set amount of time.

Ability: Quick Dash

Description:

The basic movement the player can use will be increased for a short amount of time after the dash action is triggered, will drain a burst of stamina.

Priority: 4

Stimulus / response sequences:

For gameplay, this player ability is accessed via combination on the mouse.

For development, the ability blueprint will open the UE5 blueprint editor - and the C++ code will open Visual Studio.

Functional requirements:

REQ-1: Be already moving.

The character must already be in motion.

REQ-2: Dash input

There must be a “dash” input by the player.

REQ-3: Enough stamina

The player must have an appropriate amount of stamina to dash, if the stamina bar reaches zero the player is unable to dash.

Enemy Mannequin

Description:

A mannequin to be used as practice target in the testing area that responds to the players presence, moves, and attacks when close. It will follow the player around if in field of vision/detectable area and attempt to attack.

Priority: 5

Stimulus / response sequences:

For gameplay, the player has no direct control over the dummy. The mannequin will be damaged by player attacks.

For development, the mannequin blueprint will open the UE5 blueprint editor - and the C++ code will open Visual Studio.

Functional requirements:

REQ-1: Health

Mannequins all have their own individual health value independent from others.

REQ-2: Damage

Target dummies can receive and apply damage to player.

REQ-3: Display health

Target dummies will display their current health in a health bar above their heads.

REQ-4: Regenerate health over time / respawn.

Target dummies health will regenerate over time and will respawn after death.

REQ-5:Movement

One player is detected by the mannequin the mannequin will independently move towards the player to attack.

Level Area for testing

Description:

An area for the player to explore and test their abilities in.

Priority: 6

Stimulus / response sequences:

For gameplay, the player will explore this area, but will not be able to leave.

For development, the map file will open in the unreal engine editor.

Functional requirements:

REQ-1: Mannequin enemies

The test area must have a set of enemy mannequins for testing.

REQ-2: Enclosed area

The area must be closed so the player cannot leave the area.

Stamina and health bar UI

Description:

The stamina and health bar UI displays the players stamina and health.

Priority: 6

Stimulus / response sequences:

When the player has drawn the bow, the stamina will be constantly depleted, on triggering a dash stamina will also be removed. The stamina bar will display this response as well as the stamina regained every tick. This is the same for the health bar but will only be depleted when attacked by the mannequins and regained over time or pressing “e”.

For development, the unreal widget editor will open.

Functional Requirements:

REQ-1: Display health

Displays the variable “health” as a “progress bar” to show the amount of health the player has left.

REQ-2: Display stamina

Displays the variable “stamina” as a “progress bar” to show the amount of stamina the player has left.

Crosshair UI

Description:

The Crosshair must display the arrows target direction.

Priority: 7

Stimulus / response sequences:

Will only appear when bow is drawn and not producing a projectile path, otherwise will not be viewable.

Functional Requirements:

REQ-1: Display

Must show arrow direction and location of where the arrow will be shot at and disappear when not in use.

Player character/ Archer

Description:

The player character must be able to move around the map according to the player input. The player must also be able to draw a bow and produce range attacks with it.

Priority: 9

Stimulus / response sequences:

In game, the player character will respond to the correct movement keys (W/A/S/D) to move around the scene. The player must also be able to respond with user input for other Abilities, for example drawing the bow will be triggered by holding down the right mouse button.

In editor, the player blueprint will open in the blueprint editor.

Functional requirements:

REQ-1: Respond to input

The player character must respond to input to move and use range attacks.

Other non-functional requirements

Performance requirements

While the game is in play, there must be a stable frame rate for the spells to function correctly. A low frame rate could cause effects like the fire zone to jump in size, when it should be a smooth animation. Low frame rates could also delay response to input actions.

Software quality

The software implementation must be kept at a high quality and concise standard. The developer strongly believes in the KISS principle (*KISS*) for development. All C++ code will be written with proper object-orientation in mind, but also towards the Unreal Engine 4 online coding standard. (*Coding Standard*)

While the development of this project was in a waterfall manner, testing and iteration was performed regularly to assure the validity and safety of the code base.

Method

Bow Draw

A core requirement of the project is to allow the player to draw a bow, for all attack abilities the bow must first be drawn before any of them can be used.

Bow Fire Trigger

The bow will fire if it is both drawing and if left mouse is clicked.

Ability: Default Arrow

First check if bow is drawn, then call function stamina drain to slowly make the player lose stamina. Then if projectile path is off call the function “Default Arrow Fun”. The Arrow speed is then set to default arrow speed and draw time is calculated, the crosshair is then set to true and is added to the view port. If the bow damage has 4 stages, all damage progressively goes up the longer the bow is draw capping out at 100.

Ability: Arcing Arrow

First check if bow is drawn, then call function stamina drain to slowly make the player lose stamina. Then if projectile path is on (this is triggered by holding the 2nd mouse button), call the function “Arc

Arrow Fun". Damage is set to 35 and the speed of the bow climes until it reaches its cap of 1000, then the clear Arc function is called. The clear arc function destroys the old spline meshed and clears the arrow paths point, as they will not be in use anymore. After that the projectile path is calculated and the results get passed through to the "update Arc Spline function".

Ability: Special Attack

The

Crosshair Widget

The widget is used by the bow, and is structured as so:

- Canvas
  - image

And is only shown when the bow is drawn and the Arcing arrow is not used. This was implemented with the use of a bool value that if true would create the crosshair and added to the camera viewport. Whilst false destroy it. The location of the crosshair was also calculated using the speed of the arrow at that given point in time, the default speed.

Stamina and health bar UI

The health and stamina bar UI, or "widget" is a user interface that is dynamically created by the player blueprint at the start of the games runtime and added to the camera viewport.

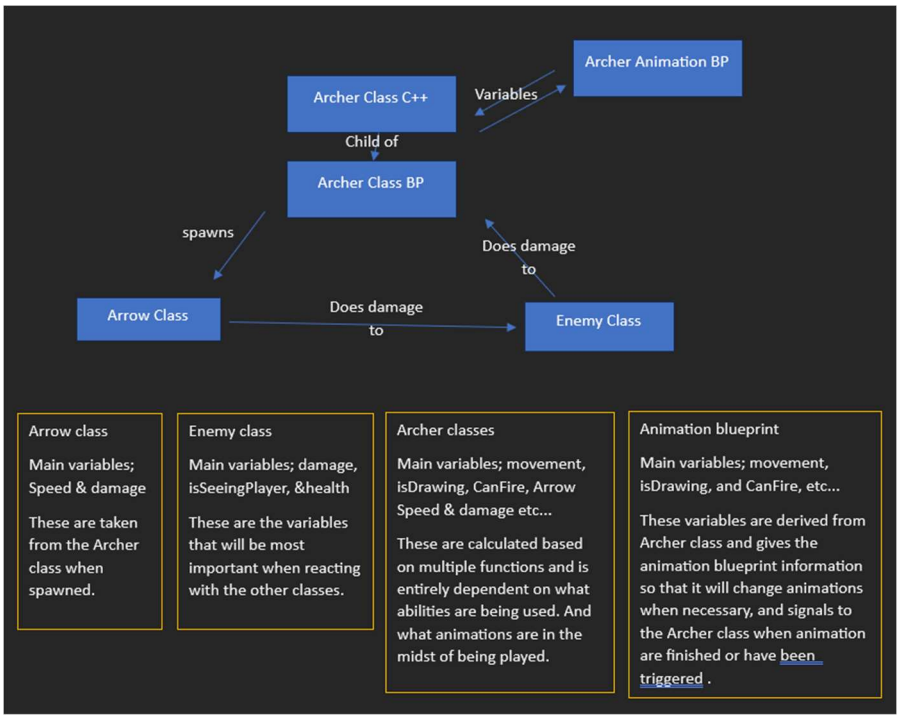
The widget is structured as so:

- Canvas
- health (Progress Bar)
- Stamina (Progress Bar)

It is constantly being update alone with the player health and stamina to display to the user. both values are shown as progress bars

Development

UML Diagram



Technical Discussion:

what you created and the techniques that you used to achieve them. This is where you describe what you did and HOW you did it

Development :

Blue prints then c++.

Conclusion:

- In “monster hunter: rise” there is an ability called “wirebug”, it is when a player is grappling up and launched to attack enemies from above. I attempted to imitate this feature but with the limited time was unable to make it truly useful and fully functioning, so it was removed from the project.
- Also another visual cue I wanted to implement was to increase the glow on the arrows when charged to different stages, sadly thanks to limited resources and time I was unable to do this but could definitely be an improvement later on.
- Ammo coating types was another thing I wanted to implement with its own unique effects and UI, but limited resources and time meant that it would have to be added later on.
- The Enemy health bar also only displays only 1 of the enemy mannequin’s health despite all having different and independent health from one another, again could not be fixed in time.
- One main thing that held me back from creating the Arcing arrow in c++ where the spinal meshes as they couldn’t be deleted or created in anyway attempted.

References:

All Web pages, Tutorials, videos etc you used. List them here.

- Bow bp tutorial

  https://youtu.be/uwKuinJCJyI?si=pHB_EFD-wpflnd-9

  https://youtu.be/2uRQj6mPR64?si=6n9caI5Xs4dWSnSH

  https://youtu.be/NAHRl7fMjK8?si=YnMG-YkQswaobd1U

  https://youtu.be/WAkiE6rQutU?si=v2Y5cMK6dD-Uzx2-

- Enemy AI
- https://youtu.be/xm-7m5Fw1HU?si=lEdcBsxBVqVhnXYO
- https://youtu.be/fp5LbdC4vek?si=Orz8u3Kf2d5KQQJC

- Paragon: Sparrow, Epic Games- Epic Content- Sep 4, 2018
- Unreal documentation