**Computer Games Development**

**Year IV**

**Project Report**

Shane Conroy

c00270031

11/05/2025

Contents

Acknowledgements...........................................................................................................................3

Project Abstract.................................................................................................................................3

Project Introduction...........................................................................................................................4

Literature Review..............................................................................................................................4

Evaluation and Discussion................................................................................................................5

Conclusions......................................................................................................................................6

### Major Technical Achievements.......................................................................................7

### Project Review...............................................................................................................7

Conclusion....................................................................................................................................7, 8

Future Work.......................................................................................................................8, 9, 10, 11

-Acknowledgements-

Thanks to my lecturers, and project supervisors.

Huge shout out to my friends, Jack and Darragh who would help me out with errors or problem solving when I got stuck and I couldn’t figure out the solution. “Apes together strong.”

I’d like to thank my really short friend Lucy for encouraging me with new features and giving me gameplay advice.

Thank you to my sister, who would stay on call with me while I worked away at code, keeping me from going insane. She definitely helped me remain sane during the late nights.

-Project Abstract-

“Save Our Sheep!” is a 2D, economy driven, farm management game. It is built using C++ and the SFML library. The game focuses on a core gameplay loop of raising sheep, producing wool, and managing the farm's economy. Players sheer sheep to earn money and upgrade their farm while also protecting their flock from wolves at night. When the player has enough Sheep they can use the farms Combiner to mix sheep together and return an advanced sheep. The game incorporates an AI behavior system, boid-style flocking, a basic economy, a UI system, and a simple particle system for extra effect.

-Project Introduction-

The motivation behind "Save Our Sheep" was to create a huge flock of sheep, with an emphasis on emergent behavior through AI flocking and enemy interactions through Wolf attacks. The project aimed to, as closely as possible, replicate the behaviours of Sheep flocks and Wolf attacks. The inclusion of an economy and player-controlled Dog, where a late addition to the game with the purpose of making the game less boring and enhancing the gameplay loop with additonal aspects.

-Literature Review-

While not a research project in the traditional sense, inspiration was taken from existing works on flocking algorithms, particularly the Boids system for autonomous agent group behaviour, which I extensively used.

Games like “*Slime Rancher”* and “*Egg Inc.”* were also used for design influence, particularly in having an economy driven by your output like “*Slime Rancher.”* My other main influence, *“Egg Inc.”* inspired the major clustering of groups and the main game loop.

Concepts from behavioural AI, finite state machines, and economic game design informed many implementation decisions.

**Flocking algorithms** (Craig Reynolds, 1987): Used to simulate realistic sheep herding behaviour.

* Wikipedia contributors. (2023, December 10). *Boids*. Wikipedia. <https://en.wikipedia.org/wiki/Boids>

**Finite-state Machines**: Used for controlling the logic behind the entities in my game

* Wikipedia contributors. (2024, March 31). *Finite-state machine*. Wikipedia. <https://en.wikipedia.org/wiki/Finite-state_machine>

-Evaluation and Discussion-

The motivation behind “Save our Sheep!” was at first to simulate a much, much larger flock and have it interact with a Wolf Ai. In my final build of the game the Player’s flock caps at around 80 sheep, I did this so it doesn’t look too cramped, but originally I wanted to have upwards of a thousand sheep on screen, flocking. I put a lot of emphasis on efficiency in my early drafts of the game, thinking I was going to have the huge herd.

My original idea was absolutely nothing like the game I have now. At first my game was going to be this platforming game, where you either played as the Shepard or the Sheepdog. The goal was to help navigate your flock across treacherous terrain. I pictured moving the flock across high, thin mountain ledges where it’d be eaay for sheep to fall.

Then I pivoted into a more ‘demonstration’ project. I was going to make the very large flock of sheep have an advanced reinforcment learning Ai. So during the day they would’ve been rewarded for eating grass and during the night they would’ve been punished for letting sheep die. In my head, I thought it’d be cool to see how the sheep would learn over generations to react accordingly to the Wolf attacks.

Then I was told it was boring, so I had to pivot from a tech demo back into making a game. I added an economy into the game, so the Player has something to build up and upgrades so the players has goals. Then I added in a sheepdog so the player had something to actively do in the game. I turned the game into whatever my final build turned out to be, which I think is pretty cool.

-Project Milestones-

For the project milestones I didnt really have any particular milestones in mind, I had major features that I would work up to. For example if I wanted to implement my Combiner function, I would first add sheep stats, then add tracking sheep greatness, etc etc. I would chip away at my “milestones.”

Throughout the project I was logging how much time I spent on the project and my total time was around 154 hours. The Sheep’s flocking taking me the longest for a single task with 11 hours.

Roughly for each month, this is what my milestones looked like

### Project Milestones

* **October** – November: Game concept and GDD finalised.
* **December**: Base SFML setup, main start loop.
* **January**: Entity systems built (Sheep, Grass).
* **February**: Flocking and AI behaviour systems added.
* **March**: Economy and upgrade mechanics implemented.
* **April**: UI, Dog, Wolf, particle systems, and polish.
* **May**: Final build and documentation.

### -Major Technical Achievements-

* Full flocking AI using cohesion, separation, and alignment.
* Multi-state behaviour for sheep, integrated with world events.
* Dynamic day/night cycle affecting gameplay and visuals.
* Custom economy and upgrade systems.
* Visual feedback via particles and UI buttons.

### -Project Review-

What went right:

* Flocking AI and sheep states worked cohesively and look natural enough.
* Economy upgrading mechanics provided clear progression.
* Good, solid and repeatable game loop.
* Combiner adds nice ‘reset’ keeping the loop fresh.

What went wrong:

* Initial wolf spawning had bugs (“ghost wolf” issue resolved with std::optional).
* Early iterations lacked engaging gameplay
* Early attempts at passive income broke progression pacing, replaced with wool selling system.
* Wolf is kinda basic, would like more advanced behaviours.
* Some user experience elements like tutorials or tooltips were not implemented.

### -Conclusions-

The game feels fun to play, it’s easy to pick up and learn, even without tooltips. It has a neat gameloop and isn’t too challenging for players. The wolves add a good night time threat to the game. Some of the mechanics, like sheering and collecting wool are very satisfying.

-Future Work-

I originally had a “Scalibilty” section in each of the Classes in the TDD, so I took them out and placed them here.

**Scalibilty -- world**

Weather elements like rain which would spawn grass, free of charge for the player, or thunderstorms which would scare the sheep, or sunny days where the day would be longer, giving the sheep more time in the field.

Flocking system could have smaller flocks ‘break off’ from the main herd

I really wanted to have a seperate screen for the upgrading builings. I wanted it to be somehting the player clicks, goes to a new screen, and then there would be lots of dilapidated buildings and the player would rebuild them.

**Scalibilty -- Sheep**

For future scalability, I would like to have added even more stats to the sheep’s internals becuase I really liked seeing all the different variations of sheep that can spawn.

Also, I’d like to have more levels or prestige as currently now the prestiges cap at 4, which seems quite low to me.

Another thing I’d like to add, given more time would be more sheep variations. I have normal, gold and infected currently which I think add a lot to the game, so having more could add a LOT of fun gameplay and unique interactions.

**Scalibilty -- Wolf**

For future improvements, I would love to give the wolf a more complex behaviour system. Instead of having him seek the closest sheep, have him find the one with most wool, or if sheep aging is ever added, have him seek a young lamb.

The wolf’s movement is also too simple, in my opinion. Adding a sudden ‘ambush’ sprint towards a Sheep would be cool to have, cool and exciting. Imagine he’s crawling real slow then bursts towards you? That’s cool.

**Scalibilty -- Dog**

The only thing I can think of is having a new building upgrade, like a dog’s trainer or something. When upgraded it would extend how long the Wolf stays stunned for.

Other than that, maybe I could add advanced features like having the player be able to give the Dog commands?

**Scalibilty -- GRASS**

Could support growth stages (fresh, eaten, regrowing) which would reward more ‘growth’ for the Sheep’s wool

Have some sort of visual change for the grass nodes because seeing a BRIGHT GREEN node on a very dark background doesn’t look the best.

If different seasons get added, have the grass change colour based on the season. Bright green for summer and spring, then orangey-browns for autumn and winter

**Scalibilty -- ECON**

The ugrade map system is very extensible — easy to add new upgrades

Could allow random sales, inflation, or event-driven pricing (Turmoil)

UI could benefit from categories or tabs if more upgrades added

Could introduce achievements or unlocks tied to economy progress

**Scalibilty -- Combine**

Could implement a combining cooldown or rare resource cost

**Scalibilty -- HUD**

Hover over button, red X when cant purchase. Or greyed out when POOR

Wanted to add particle effects when buying upgrades

**Scalibilty -- Partcile**

Could add particle trails, shockwaves, or burst styles based on different events

WoolParticles could have rarity tiers with different values or colors. Like with prestige sheep, they still die and pop white