Jeffery Meverden
COMPSCI 657 – 203
Functional Programming
Final Project Proposal
3/31/2022

Final Project Proposal

Problem Statement

I propose to make a video game in Haskell. The game will be a top-down shooting game in the style of the arcade classics. The genre will borrow from Soulslike Metroidvania as players explore to find power-ups that enable triumph over enemies of increasing difficulty. The game will culminate in a boss battle that the player cannot overcome without building a reserve of strength.

I will use the Simple DirectMedia Layer library to implement the game. The SDL2 library will allow for processing input and outputting graphics and audio. The SDL-TFF library can enable rendering text on a user-interface. Meanwhile, the SDL-Image library will support loading sprites. I will employ Monads to achieve the simulation of a game world. The IO and State Monads will be indispensable. I will also devise several Algebraic Data Types to represent objects in the game.

Novelty

My project is novel because there are few games written in Haskell, or functional languages in general. In my research, I discovered Keera Studios, a game development atelier whose achievements include several demonstrative games written in Haskell. Their titles are akin to mobile games in their simple and accessible style. A more daunting challenger arrives as Incoherent Software's *Defect Process*, a commercial game also penned in Haskell. Aside from these examples, most games written in functional languages are incomplete tech demos or rendering engines that see little use, such as LambaCube 3D.

In my literature review, I discovered the game-development journal of Ashley Smith, who wrote several articles on her experience blazing the trail of engineering games in Haskell. She opens her essay "Of Boxes and Threads" with the statement, "Making a game in Haskell is a pioneering process. Despite the fact that there's a page on the wiki and a full subreddit dedicated to the purpose of making a game in this beautiful language, not many people have actually succeeded making anything close to what current game developers can already achieve." Regarding the gallery of Keera Studios, she writes, "I don't believe there's enough there to prove anything regarding the power of Haskell in games." She concludes a related article with her encouragement, "I want to see indie developers start embracing the power of high-level languages to make development easier on their already-difficult lives. If people can make cool games in JavaScript, then why not Haskell?" I consider her inspiring blog to be the launchpad for my project.

My investigation of Haskell games has not yielded a work quite like the project I intend to develop. My game will be unique in that it will be a top-down action game, distinguished from its peers by its genre and style. Furthermore, my project will be set apart by its completeness: the game will not be a demo, but rather a full game with a definite beginning and end.

Utility

My project will be useful because it will provide an example of how games can be made in Haskell, which can be studied by developers interested in stretching beyond the bounds of imperative programming. Because the menagerie of Haskell games is nearly vacant, even a simple game can serve as a brick in the wall that builds toward popularizing functional game development. Though my project may not be superior to others, its uniqueness will help diversify the population of Haskell products.

The utility of my project is bolstered by its concrete nature. Rather than pursuing a purely academic or theoretical goal, I plan to build a piece of application software that can be enjoyed by any user no matter their background. This plan is further useful to myself because it will help me improve as a game developer. Understanding my favorite subfield of Computer Science from the new and often perplexing lens of functional programming will help me make better games and better appreciate this style.

Difficulty

This odyssey will be wrought with challenge because functional programming is incompatible with the dominant design philosophy of video games. From a ludological perspective, a game is simply a state machine that changes given user input and its own internal logic as the player flows through the game unto either a win or loss condition. Game programs are almost entirely side-effects, often run within a game loop. In Haskell, there is no mutable state, no looping, and no true flow of control like that of imperative languages. To build a game in Haskell, I must adopt a new architectural paradigm. This will involve recursive functions that accept the game world and return a new world that represents the updated state.

On a Stack Exchange forum post, Haskell game developer Jake McArthur describes the challenges of our shared endeavor, “There are not many great libraries for making games in Haskell, and not many Haskellers write games in it, so it's difficult to find resources and help on this matter.” A drought of resources is an issue I have struggled with in this course. However, I have found this hub page on the Haskell Wiki that will guide me to many useful tutorials.

Another functional game developer, Claudia Doppioslash, writes on Quora, “There is very little overlap between people who know Haskell, and people who know how to make games at a professional level.” Although there is an intimidating divide between Haskell and game development, there is also an exciting opportunity to bridge the gap. I believe my project will be a small but positive step for game developers and Haskell.

Achievability

To maintain a realistic goal, the scope of this project will be limited. The game will showcase simple core mechanics that may be expanded in the future, but are small enough to be implemented feature-complete given a brief timespan of development. Although there are not many Haskell games, there is a bounty of libraries and tutorials written by brilliant programmers upon whose shoulders I may clamber to realize my design.

When I began to study functional programming this semester, I wanted to see what games were made with Haskell, and was surprised to find so few. Game development has always been my most treasured aspect of Computer Science and also the subfield I am most confident in. I believe I will attain a strong respect and admiration for functional languages if I can complete this project in Haskell. I may fail, but if there was ever a programming challenge that I could overcome, I would want it to be this.

References

1. Haskell.org. “sdl2: Both high- and low-level bindings to the SDL library (version 2.0.6+).” https://hackage.haskell.org/package/sdl2

2. Haskell.org. “SDL-ttf: Binding to libSDL_ttf” https://hackage.haskell.org/package/SDL-ttf

3. Haskell.org. “SDL-image: Binding to libSDL_image” https://hackage.haskell.org/package/SDL-image

4. GitHub via Keera Studios. “Haskell Game-Programming – Examples” https://github.com/keera-studios/haskell-game-programming/wiki/Examples

5. Steam via Incoherent Software, LLC. “Defect Process” https://store.steampowered.com/app/1136730/Defect_Process/

6. YouTube via Kaviolalainen. “LambdaCube 3D (Haskell rendering engine) – Quake 3 example – 2012-09-08” https://www.youtube.com/watch?v=JleoASegUlk

7. Ashley Smith. “Of Boxes and Threads: Game development in Haskell” https://aas.sh/blog/of-boxes-and-threads/

8. Ashley Smith. “An Introduction to game development in Haskell using Apecs” https://aas.sh/blog/making-a-game-with-haskell-and-apecs/

9. Stack Exchange via Jake McArthur. “What are the challenges and benefits of writing games with a functional language?” https://gamedev.stackexchange.com/questions/374/what-are-the-challenges-and-benefits-of-writing-games-with-a-functional-language

10. Haskell Wiki. “Game Development” https://wiki.haskell.org/Game_Development

11. Quora via Claudia Doppioslash. “Is Haskell used in video game programming” https://www.quora.com/Is-Haskell-used-in-video-game-programming