Product Requirements Document (PRD)

Problem Description

At the moment, there isn't a very good free program that accurately simulates the physics of vehicles. If we were to create such a program and make it available to download for free, our program would be a great fit for many consumers.

Scope

This product will not be sold commercially, and a big reason this project is happening is because it will be used by our project partner, Chris Patton. Though, the project will also be open to the public to download for anyone else who wants it.

Use Cases

These user stories are adapted from the list of our project’s basic requirements.

- The user will be able to use a keyboard or gamepad as a form of input for the vehicle.
- The user will modify vehicle parameters, like the weight of each part, the size of each part, the traction of the wheels, and more with an on screen GUI (stands for “Graphical User Interface”, which is a window on the screen that the user can click and interact with).
- ~~The user will record the vehicle’s current parameters, and the vehicle’s current position, velocity, and acceleration, and be able to save those values to a file.~~
- The user will be able to load a vehicle's parameters and seed from locally saved data.
- The user will be able to generate terrain to then start a game on.
- The user will be able to interact with the environment.
- ~~The user will be able to host a world for other players to join.~~
- ~~The user will be able to connect to another player’s world through the main menu.~~
- The user will be able to interact with other users.
- The user will be able to hear sounds from the environment.
- The user will hear the vehicle's engine audio change in pitch depending on the acceleration of the vehicle.

Purpose and Vision (Background)

Our aim is to improve an existing interactive driving simulation that provides users with a realistic approximation of how real vehicles would act in certain environments, terrains, and conditions that the user can set up.

We want our program to be able to model the physics of vehicles accurately enough that in most cases, it would nearly identically show what a real-life car would do. Essentially, we want people interested in physics to be able to use this program to model specific situations for whatever purposes they may need it for.

Stakeholders

- Chris Patton (Project Partner)
    - Updated weekly
    - Update on progress made and get feedback on research/implementation
    - Can help make decisions and guide us:
        - New features and implementation
        - Project scope
        - Project timelines
        - Suggest priorities
    - Needed to make informed decisions:

        - Skill sets acquired/needed
        - Timeline and team members schedules
- Team Members
    - Updated weekly
    - Update on progress made and get feedback on research/implementation or testing
    - Can help make decisions and guide us:
        - New features and implementation
        - Project scope
        - Project timelines
        - Assign priorities
    - Needed to make informed decisions:
        - Design process
        - Testing
        - Skill sets acquired/needed
        - Timeline and team members schedules

Preliminary Context

Assumptions

- The game engine we are using for development, Bevy, is well-tested, reliable, free to use, and open-source.
- We will be developing the application using the Rust language, which is well-tested, reliable, free to use, and open-source..
- We will be using the following Bevy libraries, which will be well-tested, reliable, free to use, and open-source:
    - ~~Matchbox~~
    - Rigid body dynamics library
    - Rigid body integrator library
    - Noise library
    - Grid Terrain for terrain mesh creation
    - Cameras for basic bevy camera controls
    - Flo_curves for bezier curves
- Our program will work on Windows 10, Windows 11, macOS 14, Linux systems, and embedded systems like mobile devices.

Constraints

- Our program must be efficient enough that users with less-powerful computers are able to run it without using more than 80% of their CPU or GPU at any given time.

- We don’t have a budget, so we need to use resources that are free and open-source or create them ourselves.
- We have from September 2023 until April 2024, which is about seven months in total, to produce a version of the existing program with the basic requirements successfully implemented.
- Deliverable due dates are on canvas (some solo, some team oriented).
- Have 4 people to work on this project “part-time”.
- Limited by the compute power of team members’ development machines.

Dependencies

We’re dependent on the Bevy game engine being reliable, robust, and efficient.

- Bevy game engine
- Chris’s initial car demo: https://github.com/crispyDyne/bevy_car_demo
- We will be using the following Bevy libraries:
    - Matchbox
    - Rigid body dynamics library
    - Rigid body integrator library
    - Noise library
    - Grid Terrain for terrain mesh creation
    - Cameras for basic bevy camera controls
    - Flo_curves for bezier curves

Market Assessment and Competition Analysis

Alternatives to our program:

- **GTTrack (link to website):** This is a great tool that does most of what our program aims to accomplish, but it is very expensive (currently $899 USD as of 10/15/2023), and consists of a set of large, heavy equipment with software that runs on said equipment.
- **aVDS Advanced Vehicle Driving Simulator (link to website):** This is a great tool, but also requires an expensive set of equipment. Though the company that makes this product claims to deliver a custom product based on customer specifications, which no other company that I researched offers.
- **American Truck Simulator, Euro Truck Simulator 2, World of Trucks (link to website):** These games are limited to only cargo trucks, and they do not claim to provide a realistic truck-driving experience. Though, they do simulate road and highway traffic realistically.

- **BeamNG.drive:** Soft-body physics simulation for crashes and simulation of vehicle movement in realistic environments. Able to record and replay vehicle states from a file.
- **Car Physics Simulator:** Suspension and crash physics. Modify car properties.

Target Demographics (User Persona)

There are multiple reasons a user might want to use this program:

- The user is interested in vehicle designs, and needs a program to quickly and easily simulate a vehicle.
- The user drives or races professionally, and wants to use the program to test certain things that would be dangerous to attempt in real life.
- The user is a physics student, or at least a person with an interest in the physics involved with vehicles, and wants to use the program as a learning tool.

Here are a few user persona that fit into these demographics:

- Taylor is a 50 year old who works in robotics at a company, and they want to be able to simulate how some of their robots might function before building a prototype.
- Jessie is a 30 year old professional racer who needs to have a deep understanding of the physics involved with their car in order to stay safe.
- Charlie is a 18 year old high school student with an interest in both physics and vehicles.
- Tony is a 57 year old retired trucker who has little computer knowledge, but wants an immersive driving experience that he can record and look back at later.

Requirements

User Stories and Features (Functional Requirements)

User Story	Feature	Priority	GitHub Issue	Dependencies
As a user, I want the physics of the vehicles to be as accurate as possible to a real-life vehicle, so that I can use the program to predict how a real vehicle would behave.	Physics System	Must Have	TBD	Rigid body dynamics and integrator library
As a user, I want the weight, traction, and other properties of specific parts of the vehicles to be customizable, so that I can make the vehicle in the simulation more similar to the real-life vehicle that I am trying to simulate.	Settings Menu	Must Have	TBD	TBD

As a user, I want the environment to look realistic, so that I can better visualize the scenario happening in the real world.	Terrain Generation	Should Have	TBD	Noise library
As a user, I want to be able to save and load the settings and parameters of both the vehicle and environment in the program, so that I don't have to waste time manually recreating those conditions later.	Save/Load File	Must Have	TBD	TBD
As a user, I want to be able to manually remap the controls on both keyboard and controller, so that I can use the control scheme that is most comfortable for me.	TBD	Could Have	TBD	TBD

Non-Functional Requirements

- The program should work equally well for keyboard users and controller users.
- The program should not have a significant increase in lag when intended features are used, within reason.
- The program should work perfectly on all different screen sizes, and in both windowed and fullscreen mode.
- If the program has a fatal error, it should crash and show an error report instead of simply freezing indefinitely.
- Because many different people will be working on this project, our code should be well-documented and well-commented, following good practices and standard variable naming conventions.

Data Requirements

- For the purposes of saving and loading from a file, the program needs to be able to access, store, and manipulate the data of the vehicle that the user created in the program. This program will have its own file type for vehicle information.
- The program needs to have a settings file that it can access and manipulate, which can be a standard text file, so that any settings that the user has changed remain the same between different sessions.
- ~~The program needs to have the proper access permissions on the host computer to allow the program to set up a peer-to-peer connection with other users (if the user wants to use the multiplayer feature).~~
- The program needs to record vehicle information alongside terrain information. Specifically, vehicle parameters and terrain seeds will be saved into a readable format so that existing conditions can be recreated later. ~~The replay system will be able to read from this file to recreate the vehicle's movements during recording. Recording system must be lightweight enough as to not impact performance stuttering when recording.~~

Integration Requirements

- This program will use the Bevy game engine, which uses Rust, which is a fast and efficient programming language. It's also free.

User Interaction and Design

- <u>Main Menu:</u>
  - <u>Settings Menu:</u>
    - Modify audio volume
    - Modify control input
    - Modify video settings
    - Modify vehicle parameters, there will be text fields for
      - Custom max/min speed
      - Mass of car
      - Gravity
      - Friction
      - Terrain intensity
      - Vehicle acceleration rate
  - <u>Multiplayer Menu:</u>
    - Choose how number of players (Cars spawned)
    - ~~Allow the user to connect with other users, or host a session for other users to connect to~~
  - Seed Text Field
    - Modify/choose a seed to generate terrain before starting the game
  - Play Button
  - Quit Button
  - Load seeds/generated terrain from seed list/files
- <u>In-Game Menu:</u>
  - View Controls
  - Settings Menu
    - Modify audio volume
  - Main Menu
  - Quit Button

Milestones and Timeline

- Visualize the motion of a vehicle in an environment.
  - This feature is technically already implemented, but we plan to improve the visuals throughout the next couple of months.
  - Expected timeline: April 2024
- Translate user inputs (mouse, keyboard, gamepad) to vehicle controls.

- ➢ This feature is only partially implemented. We plan to have this feature complete as soon as possible, because the development of other features may depend on the completion of this feature.
    - ➢ Expected timeline: DONE
- ❖ Modify vehicle parameters (weight, size, etc) with an on screen GUI.
    - ➢ This feature should be completed within the first month, and gradually improved throughout the following month.
    - ➢ Expected timeline: April 2024
- ~~❖ Record vehicle state (position, velocity, acceleration) and save to file.~~
    - ~~➢ This feature could be started sometime within the next month, and completed over the course of a couple of weeks.~~
    - ~~➢ Expected timeline: April 2024~~
- ~~❖ Replay recorded state from file.~~
    - ~~➢ This feature depends on the above feature to be completed before development can begin. Once development begins, this feature should be completed over the course of a couple of weeks.~~
    - ~~➢ Expected timeline: April 2024~~
- ❖ Terrain Generation
    - ➢ Users will be able to choose a seed to procedurally generate the terrain.
    - ➢ The environment will include navigable terrain and obstacles.
    - ➢ Expected timeline: April 2024
- ❖ 3D Audio.
    - ➢ Vehicle engine audio that changes as vehicle acceleration changes:
    - ➢ Environmental audio: Dependent on Terrain Generation
    - ➢ Expected timeline: March 2024
- ~~❖ Peer-to-peer connection support.~~
    - ~~➢ Users should be able to have the option to connect to other users in order to use this program collaboratively and remotely.~~
    - ~~➢ Expected timeline: January 2024~~

Goals and Success Metrics

Goal	Metric	Baseline	Target	Tracking Method
The 3D visuals look realistic and visually pleasing	How well do the program's visuals allow the user to imagine the scenarios that take place in the simulation, in real life?	Crude visuals, with most surfaces being a solid color.	Realistic visuals, with accurate shading.	Survey

The UI is complete	Does the UI allow the user to use every intended feature?	Very little UI, if any at all	Menus and buttons for every feature.	Checklist
Simple UI	Average feature location time	30 seconds	10 seconds	Plausible Analytics
The vehicle physics are realistic, and measurable	How accurately does the program simulate real life?	Uses the game engine's default physics.	Uses finely-tuned custom settings that makes the vehicle behave more realistically, and shows the velocities and accelerations of all implemented vehicles.	Compare the program's simulation to real life vehicle tests
The multiplayer functionality works well~~, and is secure~~	~~Average success rate for attempted connections~~ More than 2 players can be on screen and functional at the same time.	90%	100%	User reports
Stability	Crash/Bugs during a single session	< 4 noticeable bugs, < 1 crash	No bugs or crashes	User reports
Product-market fit	How would you feel if you could no longer use this product?	Very disappointed < 40%	Very disappointed > 40%	Interview

Open Questions

Q: What is our target audience?

A: Engineers will use it as a tool for engineering projects. Theoretically, they might be used for Chris Patton's client.

Q: What kinds of vehicles will we be simulating?

A: Our team will be focusing on automobiles (cars, trucks, etc.)

Q: What operating systems will our program be able to run on?

A: Everywhere! Mac, Windows, and Linux.

Q: Are there any other good existing alternatives to the program that we are going to be developing that I didn't mention?

A: Yes, and they have been added to the "Market Assessment and Competition Analysis" section of this document.

Q: Will we be starting from scratch?

A: We will be starting from Chris Patton's sample program.

Out of Scope

- We will only be implementing this program onto desktop and laptop computers on Windows, macOS, and Linux systems. We will not be porting the program to consoles or smartphones unless we have extra time leftover, and we think it would be viable.
- This section will be updated with additional entries once I have more information.
- Randomly generated racetracks that you can start on the map.
- There will be no weather simulation.
- There will be no interactable animals.
- There will not be many vehicle shapes/bodies.
- There will be no ray tracing.