**Software Requirements Specification**

for

Driving Simulator

Version 1.0 approved

Prepared by Team Unicorn

9/23/2023

Version 1.0

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 3

1.1 Purpose 3

1.2 Document Conventions 3

1.3 Intended Audience and Reading Suggestions 3

1.4 Product Scope 3

1.5 References 4

2. Overall Description 4

2.1 Product Perspective 4

2.2 Product Functions 4

2.3 User Classes and Characteristics 4

2.4 Operating Environment 4

2.5 Design and Implementation Constraints 4

2.6 User Documentation 5

2.7 Assumptions and Dependencies 5

3. External Interface Requirements 5

3.1 User Interfaces 5

3.2 Hardware Interfaces 6

3.3 Software Interfaces 6

3.4 Communications Interfaces 6

4. System Features 6

4.1 Load Levels 7

4.2 Control the Player’s Car 7

4.3 AI Drivers 8

4.4 Keep Score 8

4.5 Explain Road Rules 9

4.6 Render Game Levels 9

4.7 Physics System 11

5. Other Nonfunctional Requirements 12

5.1 Performance Requirements 12

5.2 Safety Requirements 12

5.3 Security Requirements 12

5.4 Software Quality Attributes 12

5.5 Business Rules 12

6. Other Requirements.......................................................................................13

Appendix A: Glossary.............................................................................................13

Appendix B: Use Case Diagram..............................................................................14

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Team Unicorn | 9/23/2023 | N/A | 1.0 |

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to create a driving simulator game that can teach the player the rules of the road as well as how to navigate common traffic situations.

## 1.2 Document Conventions

|  |  |
| --- | --- |
| AI | Artificial Intelligence (indicates an element is computer-controlled) |
| UI | User Interface |
| User/Player | The person interacting with and being taught by the software. |
| Minimap | Miniature map included in the UI to help the player navigate the level |
| Level | The combination of streets, intersections, traffic signs, and other cars that the player navigates |
| Referee | In the use case diagram (Appendix C), the referee refers to the game logic that moves cars, detects collisions, and detects whether the player is following traffic rules |
| Robot | In the use case diagram (Appendix C), the AI or programming logic that controls all cars on the road other than the player’s |

In Section 4-System Features, the priority of each major requirement is inherited by its sub-requirements.

## 1.3 Intended Audience and Reading Suggestions

This document is intended to be a blueprint for creating a driving lesson simulator. It is also intended to express the project’s goals to our stakeholders, in this case our professor.

## 1.4 Product Scope

The purpose of this driving simulator game is to teach players about the rules of the road and how to navigate common traffic situations to improve how drivers respond in real scenarios. The project will have various levels, each pertaining to a different traffic situation or combination of traffic situations. We will create a virtual representation of several different types of roads, AI vehicles to create other traffic, and a player-controlled vehicle that the player must navigate safely to their destination. The simulation will also include traffic lights, stop signs, and various intersections for the player to navigate.

## 1.5 References

- <https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database>

- <https://www.uml-diagrams.org/use-case-reference.html>

- https://ase.in.tum.de/lehrstuhl\_1/people/people-archive/43-publications/books/264-oose-bumpers-requirementsanalysisdocument

# 2. Overall Description

## 2.1 Product Perspective

The simulator we plan to create will be a free replacement for other outdated simulators. We plan to take a new approach to the driving simulator genre by creating a level-based system where users collect points throughout their progression.

## 2.2 Product Functions

* Traffic Simulation
* Training/education
* Data Logging/ Score
* Give feedback
* Take keyboard inputs
* Identify and Communicate user mistakes

## 2.3 User Classes and Characteristics

Our product is going to be used mostly by those looking to start their journey on the road. The simulator that we're creating will help teens learn the rules of the road and give them practice on tough situations before being behind the wheel themselves. The product won't just be used by new drivers, though. It will be a helpful tool for those looking for a refresher on specific rules that may have been lost to them over time.

## 2.4 Operating Environment

The software is intended to be run only on laptops and desktop PCs. It will only be supported on Windows devices running at least Windows 10 or newer. The user must have a working keyboard and mouse/trackpad to navigate the simulator. Once downloaded, the software will not require an internet connection to run.

## 2.5 Design and Implementation Constraints

The biggest challenges we’ll face during development will stem from our lack of time and varied levels of experience across the team. We’re all at different stages of school and therefore have different levels of experience which can sometimes make it hard to be on the same page.

## 2.6 User Documentation

The simulator will provide a tutorial for each level. The tutorial will teach the user how to maneuver through each level. If the tutorial isn't enough, the user will have the option to view a reading section. The reading section will explain the traffic rules needed to complete the level so that they don't just complete the level but understand the reason behind each decision.

## 2.7 Assumptions and Dependencies

We are assuming that users of this software have: a computer running Windows 10 or 11, Python 3.10 or later, a display capable of displaying in 1920x1080 full screen, and pygame version 2.5.0 or later.

# 3. External Interface Requirements

The user will need a computer to run the software, a display monitor to be able to see situations and a keyboard to make decisions in response to those situations that they see on the monitor.

## 3.1 User Interfaces

When the user executes the software, they will be introduced to a title screen once the software has booted. At the title screen, they will press any key on their keyboard to progress to the main menu. At the main menu, the user will be given the option to open the level selector or quit the game. The level selector will be a menu with a list of available levels in numerical order, as well as a brief name for each level. Upon selecting a level, the user will be given the option to view a tutorial reading section which explains the traffic rules the player needs to understand to complete the level. After they finish reading the tutorial, or if they chose to skip it, the game will display the selected level. Each level is a combination of various roads, intersections, traffic signs, and other cars which the player must navigate. On-screen arrows could indicate which way the player should go, but each level will be designed to have only one path for the sake of simplicity. At a minimum, for each level the game will display the player-controlled car on a road. From there, using either the arrow keys or W-A-S-D keys on the keyboard, the user can drive forward (up arrow or W), backward (down arrow or S), swap to a lane on their left or make a left turn (left arrow or A), or swap to a lane on their right or make a right turn (right arrow or D). In one corner of the screen the game will display the player’s current speed, and in another, their current score. The player’s score will go up when they successfully follow traffic rules, and down when they fail to obey traffic rules. If the player successfully reaches the level’s endpoint with a sufficient score, a success screen will be displayed, and the player will be given the option to return to the main menu, where the completed level will now be marked as such. If the player does not have a sufficient score upon reaching the end of the level due to breaking too many traffic rules, a failure screen will be displayed, giving the player the option to restart the level or quit to the main menu. Likewise, if the user has a collision, the user’s vehicle will explode violently, and a message box will be displayed offering the player a choice to either restart the level or return to the main menu.

## 3.2 Hardware Interfaces

This software will be supported on Windows 10 & 11 laptop devices and desktop computers. The software also has a fixed full screen resolution of 1920x1080, so a monitor with those dimensions or larger will be required.

## 3.3 Software Interfaces

This software will require a computer running Windows 10 or 11, as that was the operating system used in development. The software is written in Python and requires at least version 3.10 for its switch-case support. The software also requires the following Python libraries:

|  |  |
| --- | --- |
| Package Name | Usage |
| pygame | Draws the game to the screen |
| math | Calculate distance and angle between level elements |
| random | Creates pseudo-random distribution of certain level elements |
| os | Accesses game elements like images |

# 4. System Features

## 4.1 Load Levels

4.1.1 Description and Priority

Allow the user to load levels from the menu. (Must)

4.1.2 Stimulus/Response Sequences

* User starts the software
* User navigates through menu and selects the level
* The program loads the level

4.1.3 Functional Requirements

REQ-1: Start the program

REQ-2: Display UI elements

REQ-3: Read a level file

REQ-4: Display the level

## 4.2 Control the Player’s Car

4.2.1 Description and Priority

Allow the user to use the keyboard to control a virtual car. The car should be able to accelerate, decelerate, and turn. There should also be other functions such as operating the car’s headlights, windshield wipers, and turn signals.

4.2.2 Stimulus/Response Sequences

* When the user presses the correct button for a specific control, the car should execute that function.

4.2.3 Functional Requirements

REQ-1: Display the car

REQ-2: Take user input from the keyboard

REQ-3: Change the car’s behavior based on the key pressed.

## 4.3 AI Drivers

4.3.1 Description and Priority

Spawn and control other cars in the level. These cars should perform the same functions as the user’s car, but they should behave automatically. Their behavior should also be able to be adjusted, with certain actions happening at certain events or different drivers behaving in different ways. (Must)

4.3.2 Stimulus/Response Sequences

* The level reaches a section where AI driver(s) is/are needed
* The AI driver(s) is/are spawned in the level, with their behavior being defined depending on the level
* The AI driver(s) execute their specified behavior
* The AI driver leaves from the bounds of the display
* The AI driver is removed

4.3.3 Functional Requirements

REQ-1: Read behavior data from level

REQ-2: Spawn AI driver with behavior from level

REQ-3: Display AI driver’s car

REQ-4: Remove AI driver from level

## 4.4 Keep Score

4.4.1 Description and Priority

A score counter should be displayed and updated based on the player’s performance. Different situations will yield different scores values, with mistakes resulting in a reduced score, and correctly maneuvering the car resulting in an increased score. Each level will have a target score to reach that is required to be reached by the end for the player to pass the level. If the score becomes negative, the user fails the level. (Must)

4.4.2 Stimulus/Response Sequences

* The player does something with an associated point value
* The point value is added to the score counter
* If the score becomes negative, fail the level
* If the user reaches the end of the level, check the score. If it is high enough, the user passes the level, else they fail

4.4.3 Functional Requirements

REQ-1: Load the level

REQ-2: Display UI elements

REQ-3: Allow the user to control the car

REQ-4: Detect collisions and relative positions of cars and other objects in the level

REQ-5: Update a running value when the conditions specified by the level are reached (good/bad driving behavior)

## 4.5 Explain Road Rules

4.5.1 Description and Priority

For each level, have an optional reading section that explains the traffic rules needed to complete the level. (Should)

4.5.2 Stimulus/Response Sequences

* User navigates through menu and selects the button for a level’s tutorial
* The program displays an explanation for each of the rules needed for the level.
* When they are finished, the user presses either a button to load the level or a button to return to the level selection screen

4.5.3 Functional Requirements

REQ-1: Start the program

REQ-2: Display UI elements

## 4.6 Render game levels

4.6.1 Description and Priority

Draw UI elements to the screen. (Must)

4.6.2 Stimulus/Response Sequences

Frame is drawn during a level sequence

Road/background must be drawn first

Cars (Player and AI) are drawn next

UI elements are drawn last

4.6.3 Functional Requirements

REQ 1: Render background. A basic solid color is probably sufficient, but we may want the color to change depending on the weather conditions we are simulating.

REQ 2: Render road elements. All of the following elements must be able to be drawn at any position on the screen or partially offscreen:

Straight road

Curved road

Roundabout

3-way stop intersection

4-way stop intersection

Traffic light intersection

Parking lot (optional?)

Highway

Highway entrance ramp

Highway exit ramp

Crosswalk

REQ 2: Render Cars

Cars must be able to be drawn at any position and rotation on the screen.

Support for multiple types of vehicles (optional?)

REQ 3: Render UI Elements

UI elements to be determined. Must need to include menus, options, could include a minimap

## 4.7 Physics System

## 4.7.1 Description and Priority

Handle car collisions, steering, surfaces, friction in real-time. (Must)

4.7.2 Stimulus/Response Sequences

Dependent on user input (gas, brake, steering), and multiple actions could be happening at once.

User brakes -> car slows down

User steers left/right -> car goes left or right (respectively)

User presses gas -> car accelerates

Physics system controls momentum, acceleration, car inputs, friction, collision detection.

4.7.3 Functional Requirements

REQ 1: Friction and surface detection

Detect what surface each tire of a car is sitting on (road, grass, other)

Detect what lane of a multi-lane road the car is currently in. Will be used in conjunction with the scoring system.

Apply friction and rolling resistance to the motion of all vehicles.

REQ 2: Respond to User Input

Calculate the physics of steering, accelerating, and braking (depending on what buttons are currently pressed)

REQ 3: Collision detection

Detect if two car bodies, or a car and a stationary object, collide. Calculate the appropriate physics, penalize user for crashing.

# 5. Other Nonfunctional Requirements

## 5.1 Performance Requirements

1. Frame Rate (FPS): try to get a nice consistent frame rate.
2. Resolution: the resolution of the game should be appropriate for whatever we use to run the software
3. AI performance: make sure the AIs within the game are able to act as a real-life person.

## 5.2 Safety Requirements

1. Ensure the provided traffic rules are accurate so users won’t violate the rules of the road in a real traffic situation, causing an accident.

## 5.3 Security Requirements

1. This game won’t take, create, or handle any user data or sensitive information, so there won’t be any security risks

## 5.4 Software Quality Attributes

1. Reliability: The simulator should provide a consistent gameplay experience with each level having an invariant sequence of roads, intersections, and AI drivers
2. Performance: As mentioned in the performance requirements(5.1), the simulator should run at a consistent frame rate which is high enough for the user to discern the motion of in-game objects
3. Usability: The user should be able to easily understand the controls for moving their own car, as well as the objective for each level
4. Realism: The situations created by the simulator should be reasonably comparable to real traffic situations

## 5.5 Business Rules

1. This simulator shouldn’t have any user-based permissions or administrative functions that would need to be secured, so there aren’t any relevant business rules

# 6. Other Requirements

Appendix A: Glossary

|  |  |
| --- | --- |
| AI | Artificial Intelligence (indicates an element is computer-controlled) |
| UI | User Interface |
| User/Player | The person interacting with and being taught by the software. |
| Minimap | Miniature map included in the UI to help the player navigate the level |
| Level | The combination of streets, intersections, traffic signs, and other cars that the player navigates |
| Referee | In the use case diagram (Appendix C), the referee refers to the game logic that moves cars, detects collisions, and detects whether the player is following traffic rules |
| Robot | In the use case diagram (Appendix C), the AI or programming logic that controls all cars on the road other than the player’s |

Appendix B: Use Case DiagramA diagram of a person's work flow

Description automatically generated