NEWGIZA UNIVERSITY
SCHOOL of
INFORMATION TECHNOLOGY

NEWGIZA UNIVERSITY
NGU.

RollRush Game

This **Final Project** is built to fulfill the requirements of

CSAI 360 - **Computer Graphics**

Spring 2025

Submitted to : Dr. Nashwa Abdelbaki

Name	ID
Mai Waheed	***202200556***
Zeina Ayman	***202200351***
Nour Helmy	***202202012***
Farida Mohamed	***202202946***
Shahd Tarek	***202202018***

Table of Contents :

1. List of Figures :

2. Introduction :

RollRush is a 3D race game written in C++ using the OpenGL and FreeGLUT libraries. It simulates a platform style race track where colored balls controlled by players move along a chain of curved tiles collecting points, remaining on track without falling off, and saving limited health and lives.

The project aimed at displaying our understanding of computer graphics principles such as 3D rendering, perspective control camera, lighting effects, texture mapping, and user input. The game is playable in single (1 player) and multiplayer (2 players) modes with personal objectives and conditions for winning and losing. The game features HUD elements such as health meters, life indicators in the form of textured hearts, and floating messages to enhance the game experience.

This paper chronicles the game's architecture, technical implementation, gameplay features, visual appearance details, and results of various game sessions. It also takes into account the problems that were encountered and how they were solved, providing information on the use of real time graphics programming in game development.

3. Game Description :

3.1 Menu System and Player Setup :

Upon launching the game, users are greeted with the main menu, where they can choose between :

- Single Player
- Multiplayer

In multiplayer mode, a submenu allows the selection of :

- Race
- Timed Score Attack
- Treasure Hunt

After choosing a mode, each player configures their profile :

- Name Input : Players type their name using the keyboard.
- Color Selection : Players choose one of four colors (Red, Green, Blue, Yellow). The system prevents both players from selecting the same color.

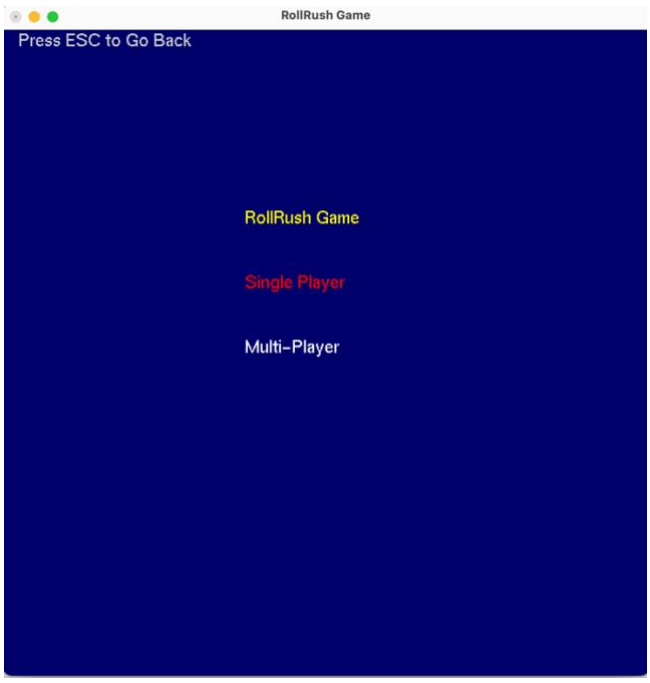

Figure 1 Main Menu.

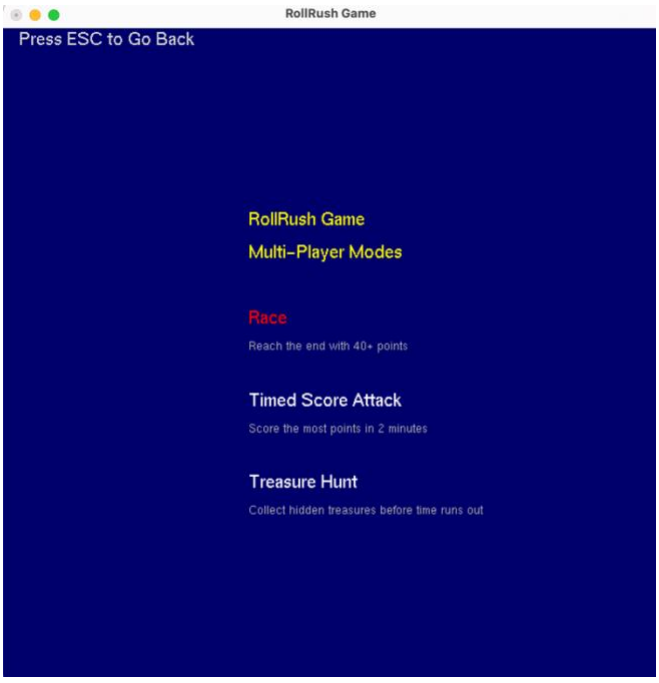

Figure 2 Multiplayer Mode Selection

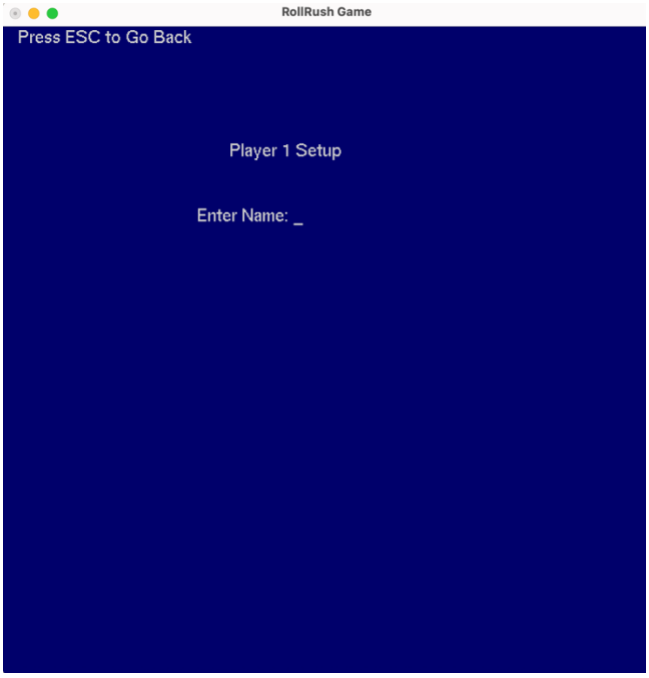

Figure 3 Player 1 Name Input

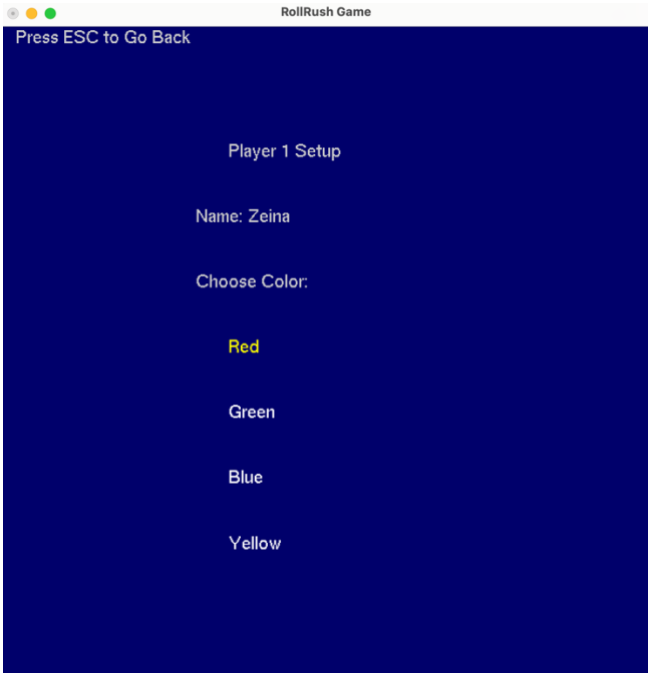

Figure 4 Player 1 Color Selection

3.2 Game Modes and Mechanics :

The game offers four distinct gameplay modes, each with different rules :

Mode	Objective	Special Rules
Single Player	Reach the finish line with at least 40 points	2-minute timer
Multi-Player Race	First to finish with 40+ points	Only one player can claim a checkpoint
Timed Score Attack	Highest score in 2 minutes	Track loops after finishing
Treasure Hunt	Reach 300 points or highest score in 3 minutes	Floating colored pickups + checkpoints

Mechanics common to all modes :

- Checkpoints : Grant 20 points when reached. Only one player can claim each.
- Pickups (Treasure Hunt only) : Floating objects that give 20-50 points.
- Health and Lives : Players have 100 health and 3 lives. Falling or staying off-track reduces health. Losing all lives results in defeat.
- Respawning : Players respawn at the last checkpoint when falling or after losing a life.

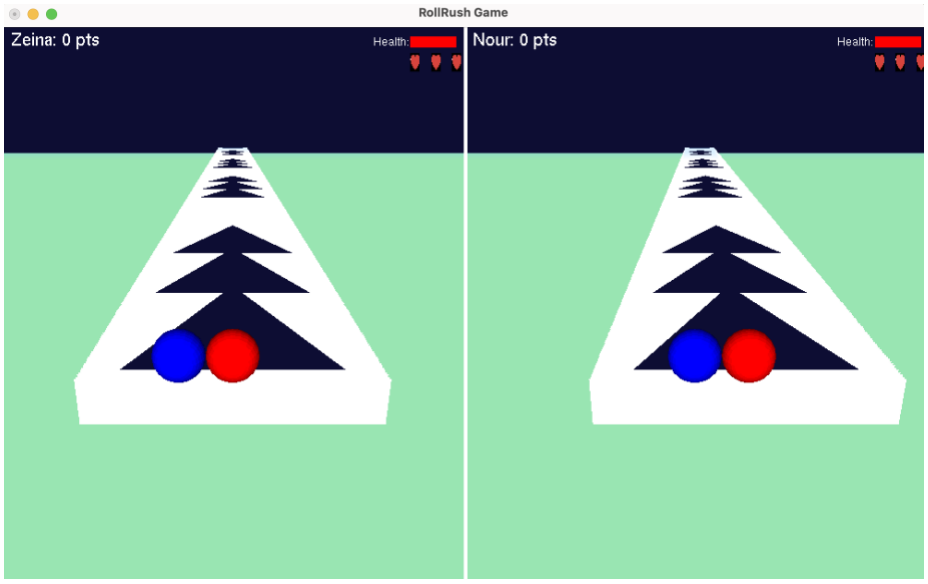

Figure 5 Race Mode Gameplay

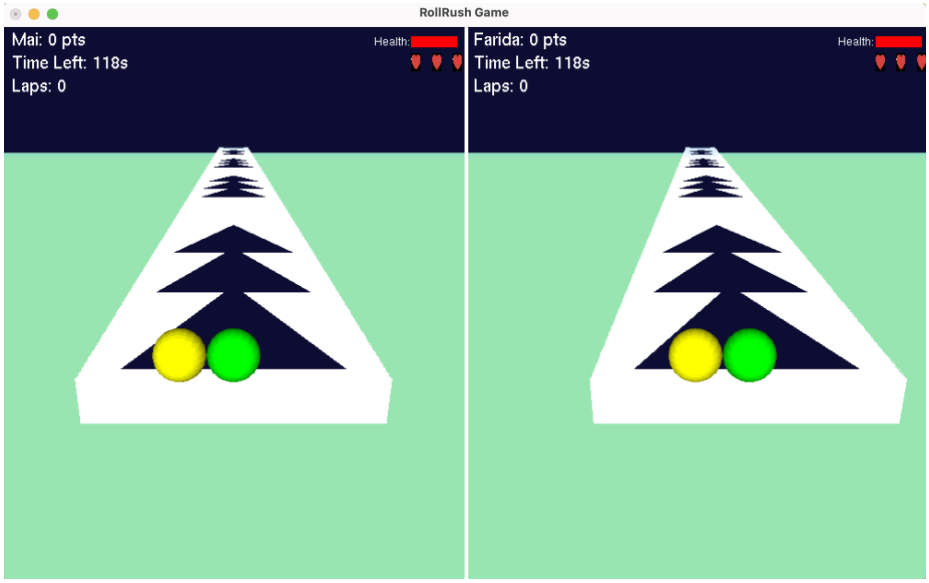

Figure 6 Timed Score Attack Mode

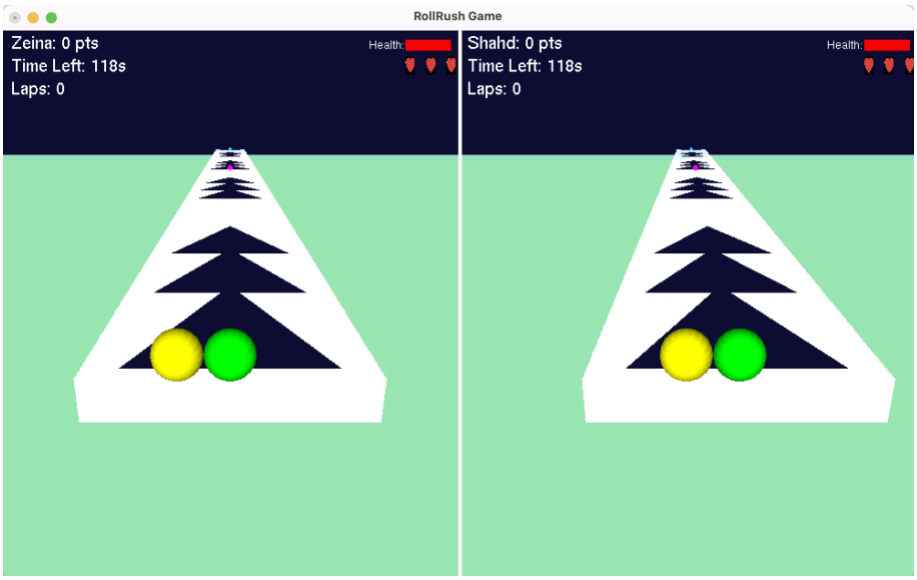

Figure 7 Treasure Hunt Gameplay

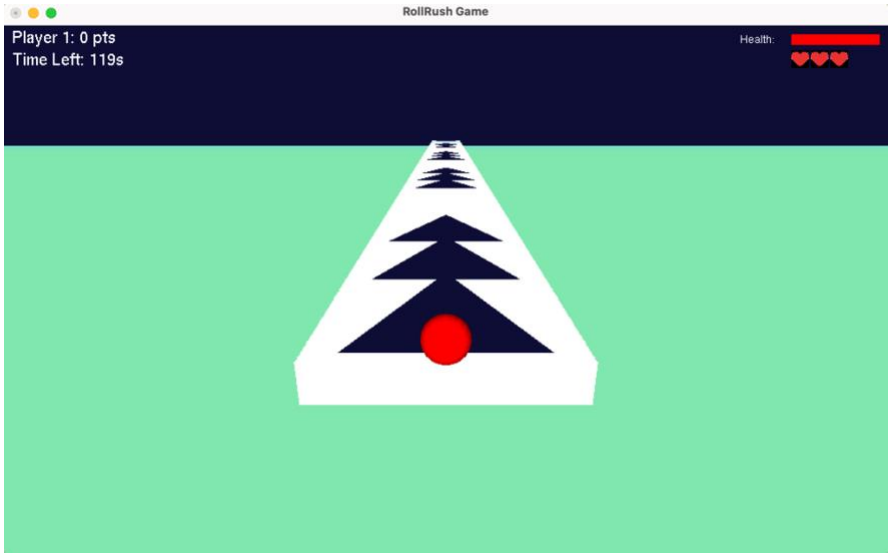

Figure 8 Single Player Mode

3.3 Visual and HUD Features :

- Split Screen Rendering : Both players have their own camera perspective.
- Health Bar : Red bar showing the player's current health.
- Lives Display : Heart textures for remaining lives (loaded using stb_image).
- Floating Text : "+20 pts!", "Too Late!" shown briefly after scoring or failing.
- Smooth Track Curves : Achieved using segment angle interpolation and tile joining.
- Track Arrows : Directional arrows are drawn on track tiles to guide players through the correct path.

3.4 Controls and Input :

Player 1 Controls (WASD) :

- W / S → Move Forward / Backward
- A / D → Turn Left / Right

Player 2 Controls (Arrow Keys) :

- UP / DOWN → Move Forward / Backward
- LEFT / RIGHT → Turn Left / Right

Global Controls :

- ENTER → Confirm menu selections / input
- BACKSPACE → Delete character during name input
- ESC → Return to previous menu
- Arrow keys → Navigate menu options

Each player controls their character independently, and the camera adjusts based on their direction and position.

4 Results :

The Race Challenge game was successfully implemented and tested across all four gameplay modes. The results reflect both the functional correctness of the game logic and the visual fidelity achieved through OpenGL rendering. Key outcomes are described below.

4.1 Functionality Testing :

- Main Menu Navigation : All menu transitions and player selections work without errors
- Multi-Player Split Screen View : Both camera views render simultaneously with correct separation
- Name and Color Selection : Inputs are handled correctly, with color conflict prevention enforced
- Game Logic per Mode : Scoring, winning conditions, respawns, and health logic work as intended
- Timer Based Modes : Countdown timers trigger appropriate end game conditions
- Checkpoints and Pickups : Points update in real time; exclusive claiming of checkpoints enforced

5 Conclusion :

The Race Challenge project demonstrates clearly how to employ OpenGL and FreeGLUT for developing an interactive, real time 3D car racing game with various game modes, sensitive input handling, and dynamic graphics rendering. Through the project, we implemented and reinforced significant computer graphics concepts such as perspective camera initialization, texture mapping, fog usage, and real time user feedback.

All the functionality that was aimed, including menu navigation, split screen multiplayer support, keeping the scores, health and lives system, and respawning mechanism were implemented and debugged. The game offers an enjoyable and addictive experience in its four modes with distinguishing mechanics and winning conditions.

Technically, modular design of code, optimized state management, and stb_image-based loading of texture guaranteed streamlined development and debugging. The project also focused on unambiguous user interface, error avoidance (e.g., disallowing the same color choice), and visual feedback during gameplay.

In total, Race Challenge not only met its design goals but was also an informative computer graphics programming practice. Improvements in the future can include sound effects, better physics, AI players, and more visual environment such as skyboxes or dynamic objects.

6 References :

- Sean Barrett, stb_image.h - Public Domain Image Loader, GitHub. https://github.com/nothings/stb

  (Used for loading PNG textures, specifically the heart icon texture used to represent player lives in the HUD)
- FreeGLUT Documentation, The FreeGLUT API Reference. https://freeglut.sourceforge.net/docs/api.php

  (Used for setting up the OpenGL window, managing input events, and handling rendering callbacks in our main game loop)
- Joey de Vries, LearnOpenGL.com - Modern OpenGL Tutorials. https://learnopengl.com/

  (Provided insights into real-time rendering, camera control, texture blending, and fog effect implementation)

- Abdelrahman192317, Race Car Game With OpenGL. GitHub. https://github.com/abdelrahman192317/Race-Car-Game-With-openGL

  (Inspired the tile-based racetrack implementation and 3D rendering of curved platform segments. Provided a useful reference for organizing game logic in OpenGL)
- Vaishnavi Killekar, Ball and Obstacle Game using OpenGL and C++. GitHub. https://github.com/VaishnaviKillekar/Ball-and-Obstacle-Game-using-OpenGL-and-C-

  (Helped conceptualize player control mechanics for a ball on a platform, especially fall detection and boundary handling)
- xinyu-evolutruster, 3D Racing Game. GitHub. https://github.com/xinyu-evolutruster/3D-Racing-Game

  (Served as visual and structural inspiration for implementing racing HUD elements like score overlays, directional guidance arrows, and split screen display)