Arab Academy for Science, Technology, and Maritime Transport
College of Computing and Information Technology
Smart Village

Project Title

Bonk game

By

Rola Khaled

Omar khaled

Zeina Ahmed

Nada Aymen

Malak Maher

College of Computing and Information Technology, AASTMT, 2025

Supervised By

Amr daba

Table of Contents

1. Introduction

Bonk is a 2D multiplayer game developed using C++ and OpenGL as part of a Computer Graphics course project. It combines physics-based gameplay mechanics with simple, competitive fun. Players can choose from three distinct maps, each providing a different environment and challenge. The game supports local multiplayer, and features intuitive menus for navigation.

2. Objective

- To demonstrate fundamental and advanced computer graphics concepts through a game.
- Implement real-time rendering, texture mapping, and interactive UI using OpenGL.
- Develop basic physics including gravity, collisions, and player movement.
- Integrate sound using OpenAL.
- Support modular map development with reusable components.

3. Technologies Used

Technology	Purpose
C++	Core game logic and structure
OpenGL	Rendering graphics (2D/3D)
GLUT	Handling windows, input, and context management
OpenAL	Sound and audio playback
STB Image	Loading textures (e.g., PNG, JPEG)
Docker	Optional containerization for easy build and deployment

4. System Overview

Main Components:

- Main Menu System – Entry point to select game mode.
- Game Engine – Handles rendering, physics, and game loop.
- Physics Engine – Controls gravity, movement, and object collision.
- UI Manager – Renders menus and in-game HUD.
- Sound Engine – Plays background music and effects.
- Network Modules – Present but currently not used in local play.

5. Menu Flow

1. Main Menu
   - Multiplayer (Not implemented)
   - Local Player

  - Options
  - Quit
2. Local Player → Map Selection
  - OneVsOne
  - GangGrounds
  - GravityOff
3. Map Loads → Gameplay Starts

6. Game Modes (Maps)

OneVsOne

Designed for two players. Simple platform layout. Regular gravity and straightforward collision.

GangGrounds

Larger arena with multiple players. Dynamic platforms and obstacles. Complex physics and interactive objects.

GravityOff

Gravity disabled or reversed. Movement becomes floatier and more challenging. Requires new strategy for jumping and dodging.

7. Code Structure

Folder hierarchy:

```
project-root/
├── src/
|   ├── core/          # Input, rendering, sound
|   ├── network/       # Networking (WIP)
|   ├── physics/       # Physics engine
|   └── ui/            # Menu & scene logic
├── include/           # Headers for all modules
├── assets/            # Textures, sounds, etc.
├── external/          # External libraries (OpenAL, stb_image)
├── build/             # Output binaries
├── .vscode/           # Editor configs
├── bonk.sh            # Launcher script
└── README.md
```

Important Files

- main.cpp – Entry point

- MenuManager.cpp/.h – Handles all menu interactions
- Player.cpp/.h – Core of player logic and physics
- GameScene.h – Scene controller for switching between menus/maps
- Renderer.cpp/.h – OpenGL drawing logic
- PhysicsEngine.cpp/.h – Handles motion, collisions, and gravity

8. Assets

Assets are stored under the assets/ directory and include:

- Player and map textures
- Background images
- Sound effects and music
- Fonts and UI elements

9. How to Build and Run

Requirements

- C++17 compiler (GCC, Clang, MSVC)
- OpenGL Development Libraries
- Make or Bash-compatible shell

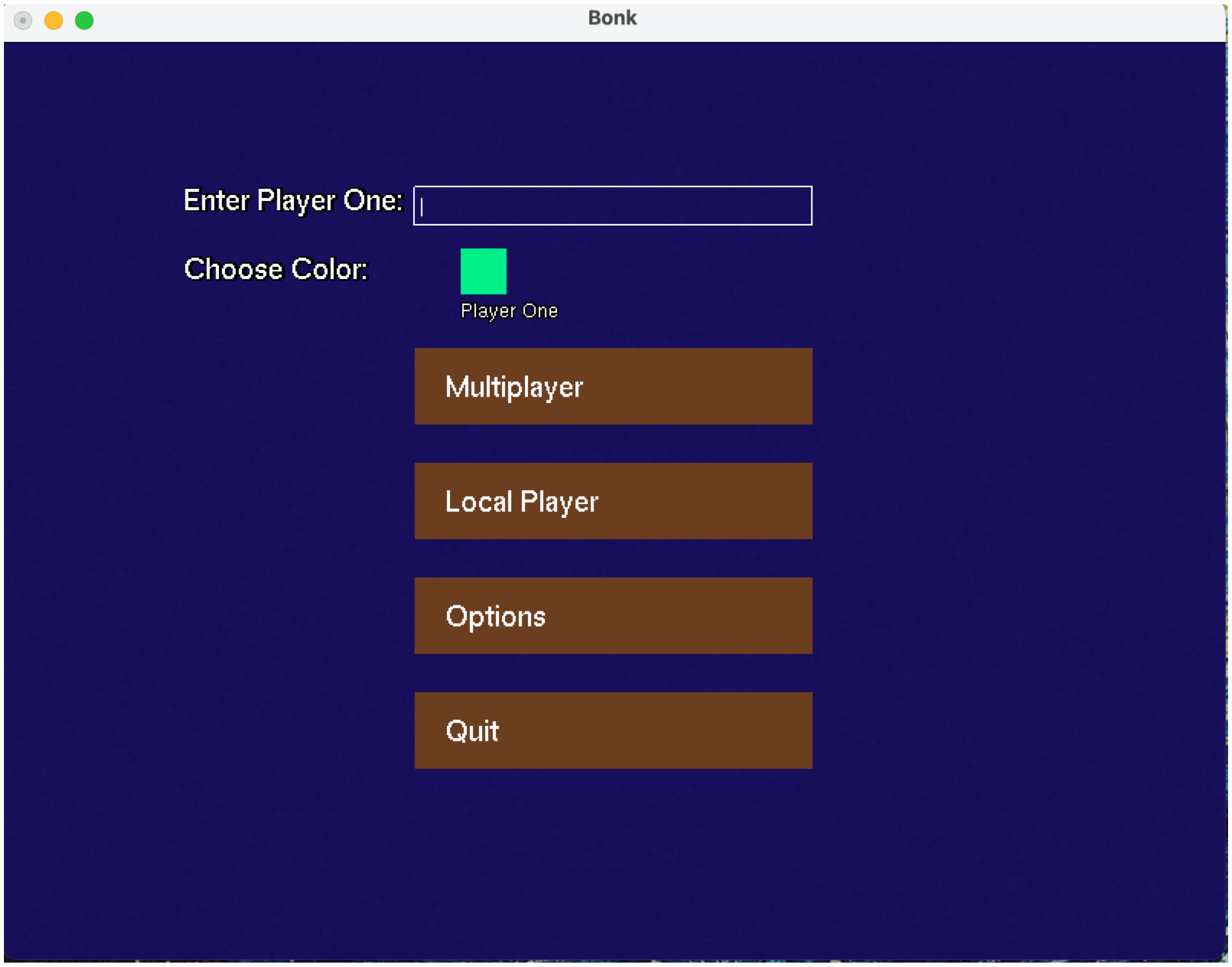
Build Steps

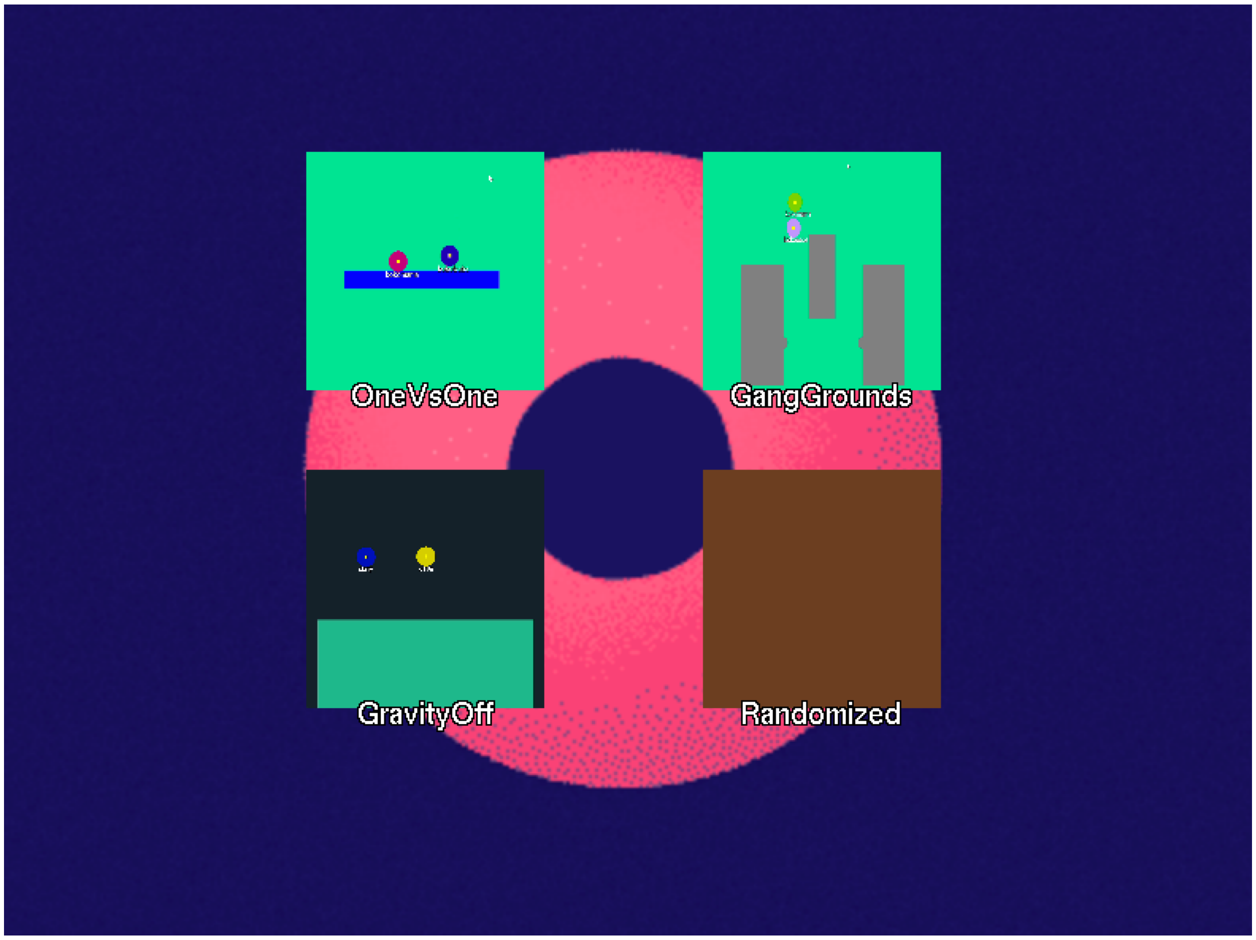
```
Chmod +x build_all.sh
```

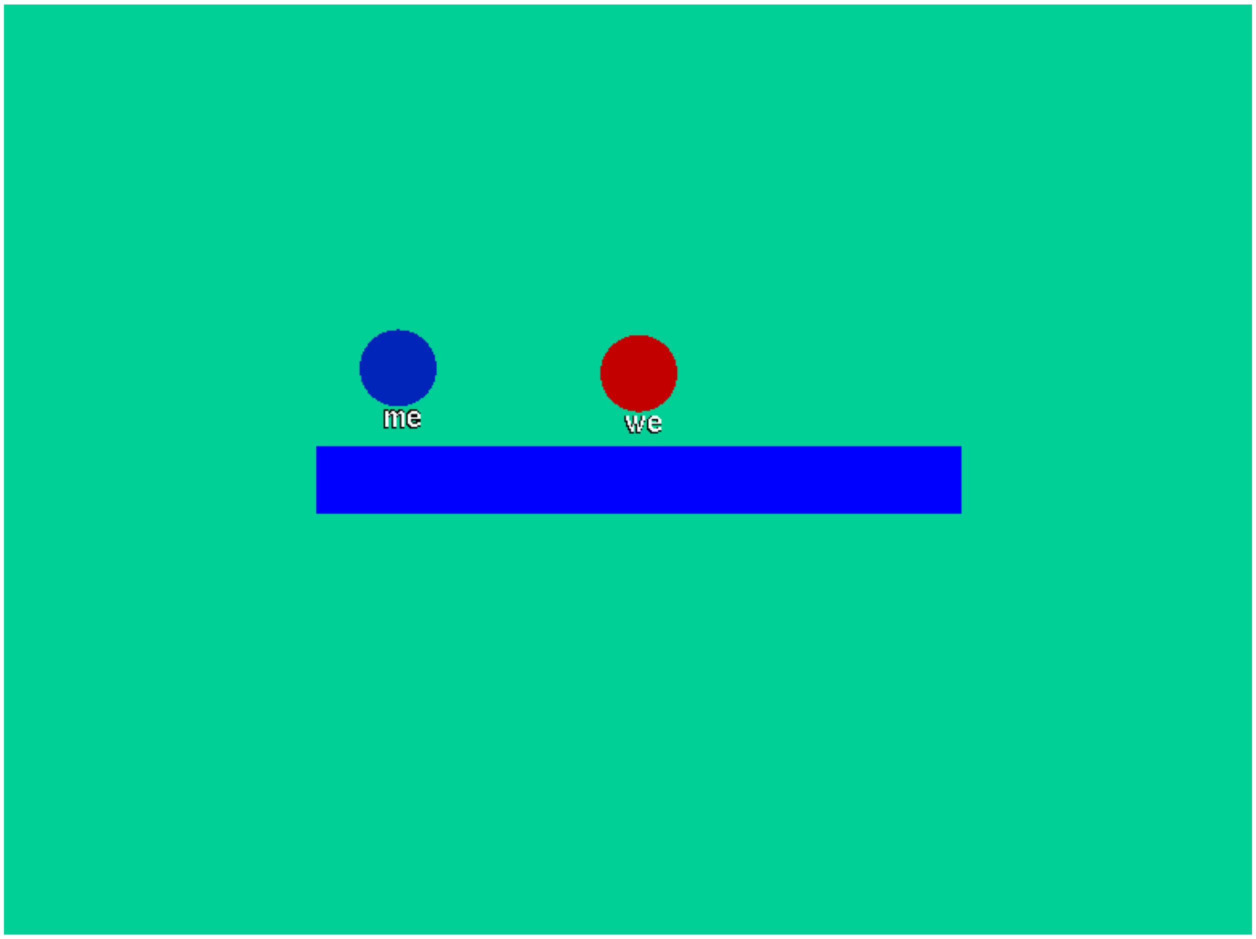
Run the Game

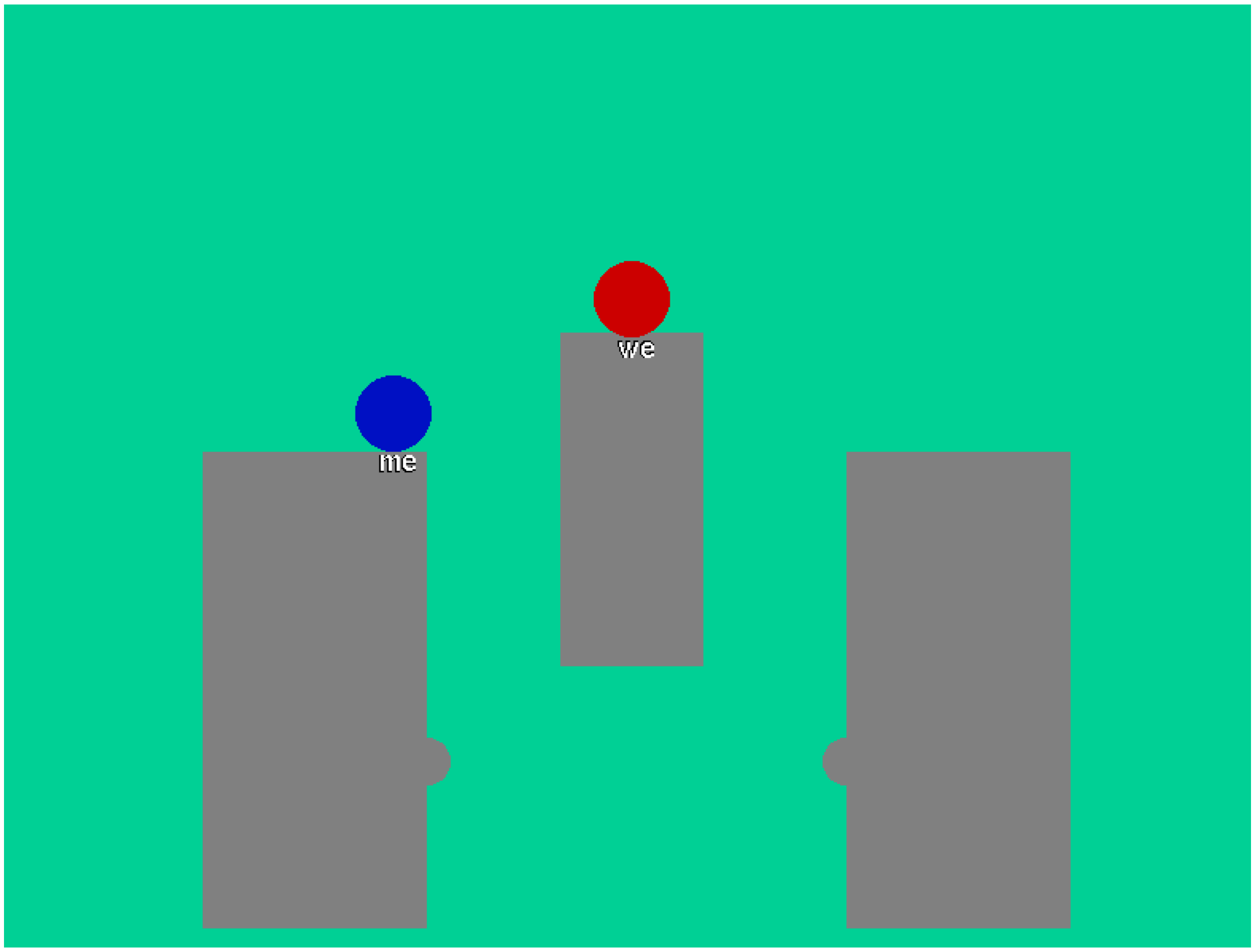
```
./build_all.sh
```

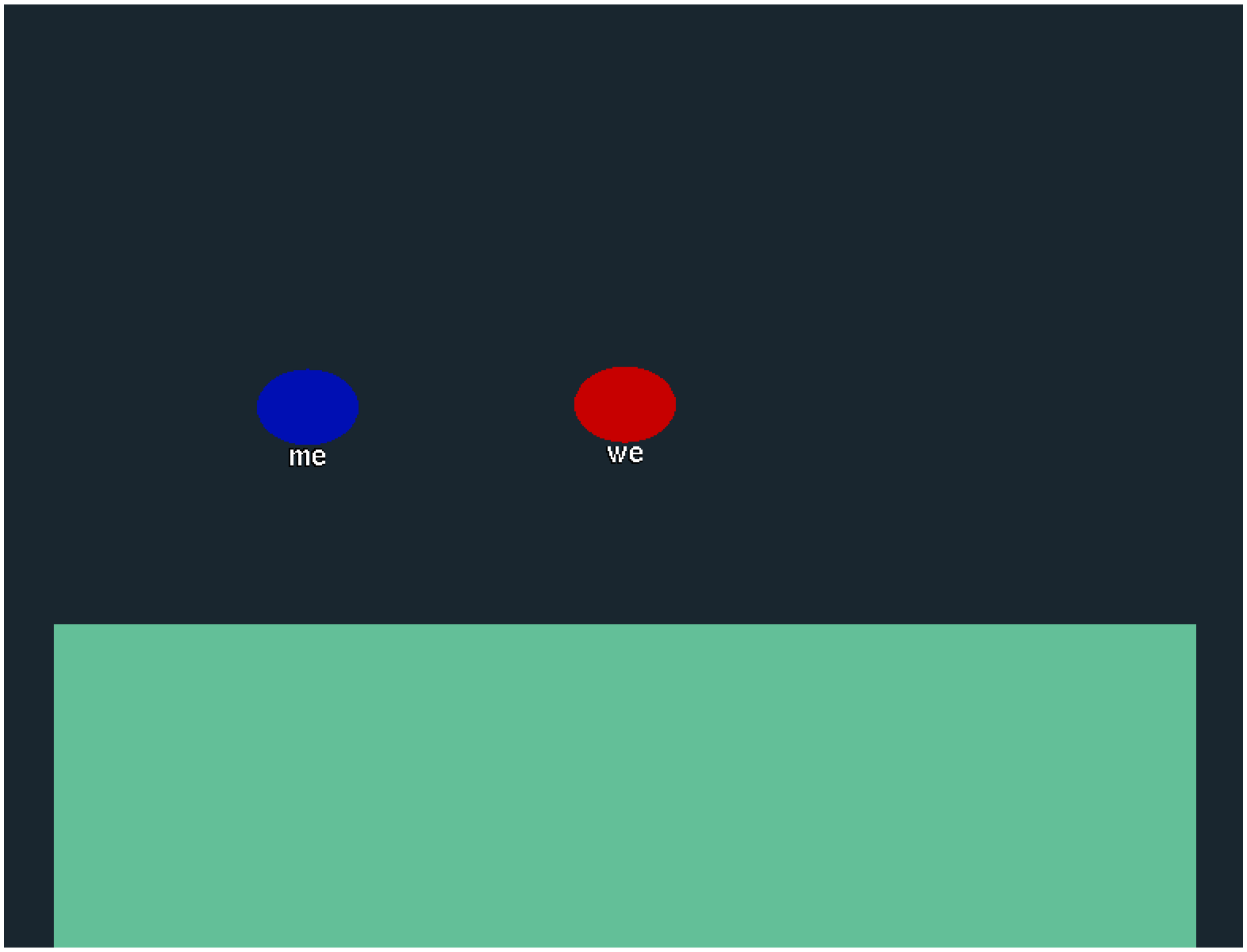
10. Screenshots




OneVsOne
GangGrounds
GravityOff
Randomized


me
we


we
me



11. Future Improvements

- Add working Multiplayer via sockets
- Implement AI-controlled bots
- Add character skins and customization
- More maps and power-ups
- Refine physics engine for edge cases

12. Conclusion

Bonk showcases the application of core computer graphics and game development principles, including real-time rendering, interactive UI, and basic physics simulation. It serves as a foundation for expanding into a larger multiplayer or networked experience in future iterations.