Retro Spaceship Game: A CLI-Based Space-Invaders Clone in C

Project group - 8
Course - CSE 115, Section – 4
Summer 2025

Radit Rifah Rahman (2521985042)
Seendeed Islam (2524611042)
Afrida Hossain Khan (2522096642)

Abstract

This document describes an Alien Invasion Command game developed in C for the CSE115 course as a retro-style command-line computer shooter game based on Space Invaders. It captures basic functions like moving, shooting bullets, destroying aliens, and scoring which are performed in simple two-dimensional grid space. The focus is mastery of basic console input control, rendering, as well as modular design and real-time systems.

I. INTRODUCTION

Developed in 1978, Space Invaders is regarded as one of the earliest and most impactful fixed shooter games. In relation to the CSE115 course, our group created a stripped down, text-based version of this game in C, displaying it in a console window. The purpose of this was to improve our grasp of programming concepts, control systems, arrays, and real-time input and output processes.

II. SYSTEM DESIGN

A. Game Structure

The game board is a 2D character array (grid[20][40]) which stores game entities: player (‘^’), invader (‘M’), bullet (‘.’), and empty space. The player starts at the bottom center and can move left/right using ‘A’ and ‘D’. Bullets are fired upward with the spacebar.

B. Game Logic

Invaders are arranged statically in a 15x2 grid. When a bullet reaches an invader, both are removed, and the player gains 10 points. The game ends when all invaders are destroyed. The program includes a simple menu, input validation, and score tracking.

C. Sound and Feedback

Upon successfully hitting an invader, a Beep() sound provides audio feedback. The screen is updated in real-time with system("cls") for rendering the grid.

III. IMPLEMENTATION

A. Input Handling

Real-time input is managed with _kbhit() and _getch() from conio.h. The player is restricted from moving out of bounds, and invalid moves do not disrupt the flow.

B. Modularity

Functions like initGrid(), drawGrid(), movePlayer(), and updateBullets() encapsulate logic for grid setup, player interaction, and bullet movement, promoting clean and maintainable code.

C. Portability

Although designed for Windows (due to conio.h, windows.h), the code is structured in a way that it could be adapted for Linux/macOS using equivalent libraries.

IV. RESULTS AND FEATURES

- Functional 2D shooter game in the console.
- Displays score in real time.
- User-friendly menu.
- Interactive feedback through sounds and win message.
- Infinite game loop with replay option.

V. CONCLUSION AND FUTURE WORK

This project highlights fundamental programming skills such as working with arrays, handling input and output, and applying modular design principles. Future works will include moving and shooting by invaders and graphical improvements to the system.

ACKNOWLEDGMENT

We extend our sincere gratitude to our course instructor, Mohammad Shifat-E-Rabbi, for his guidance and motivation throughout the CSE115.4 course. Additionally, we acknowledge the assistance of OpenAI's ChatGPT and Perplexity AI in researching and structuring coding techniques along with insights from Herbert Schildt’s book.

REFERENCES

[1] *Teach Yourself C* - Herbert Schildt

[2] OpenAI, "ChatGPT," https://chat.openai.com, accessed July 2025.

[3] Perplexity AI, https://www.perplexity.ai, accessed July 2025.

[4] GitHub Repository: https://github.com/limewolf6/Retro-spaceship-game