


Cat and Mouse - Chasing Game!

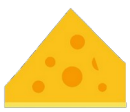
Computer Graphics Project Documentation

Mohamed Naeem Mohamed Mohamed Shousha

3rd Year Computer Science

Computer Graphics
Faculty of Computers and Informatics (FCI)
Suez Canal University
Computer Science Department

May 11, 2025

Contents

1 Introduction **1**

2 Game Features **1**

3 Game Design & Implementation Details **2**

3.1 Maze Design & Levels 2

3.2 Characters & Sprites 2

3.2.1 Player (Mouse) 2

3.2.2 Opponent (Cat) 2

3.3 Collectibles & Power-ups 3

3.3.1 Cheese 3

3.3.2 Cat Slowdown Power-up 3

3.4 Game Mechanics 3

3.4.1 Game States 3

3.4.2 Heads-Up Display (HUD) 4

3.4.3 Difficulty Scaling 4

3.5 Core OpenGL Concepts Used 4

4 Additional Screenshots **7**

1 Introduction

This document details the implementation of “Cat and Mouse - Chasing Game!”, a 2D maze game developed using C++ and the OpenGL Utility Toolkit (GLUT). The game features a player-controlled mouse navigating through various levels, collecting cheese pieces while being pursued by an AI-controlled cat. Key elements include dynamic difficulty scaling, collectible power-ups, multiple maze layouts, and a clear game progression system with distinct game states. The primary objective was to apply fundamental computer graphics concepts to create an interactive and engaging game experience.

2 Game Features

The game incorporates the following key features:

- **Core Gameplay:** Player controls a mouse to collect all cheese pieces on a level while avoiding a chasing cat.
- **Custom Sprites:** Unique, custom-designed sprites for the player (mouse), opponent (cat), cheese, and power-ups, drawn using OpenGL primitives. The initial visual design for characters was aided by an external shape-drawing tool that exported OpenGL primitive code, which was then integrated and transformed within the game.
- **Maze Navigation:** Movement within a 2D grid-based maze, including a tunnel feature for screen wrapping.
- **Rounded Wall Aesthetics:** Walls are rendered with a smooth, rounded appearance with outlines.
- **Multiple Levels:** Three distinct, pre-designed maze layouts providing varied challenges.
- **AI Opponent:** The cat utilizes a Breadth-First Search (BFS) algorithm to find and pursue the shortest path to the player.
- **Dynamic Difficulty:** The cat’s speed increases progressively as the player collects more cheese on each level, following a non-linear curve.
- **Power-up System:** A ”Cat Slowdown” power-up spawns randomly, which, when collected, temporarily reduces the cat’s speed.
- **Game States:** Comprehensive state management including a Start Menu, Playing, Paused, Game Over, Level Cleared, and Final Win screens.
- **On-Screen HUD:** Displays current level, total accumulated score, and the number of cheese pieces remaining on the current level.
- **User Controls:** Standard WASD for movement, ’P’ to pause/resume, ’R’ to reset the game, and ESC to quit.
- **Visual Feedback:** Includes a sparkling animation for power-ups and distinct screen overlays for different game states.


3 Game Design & Implementation Details

3.1 Maze Design & Levels

The game world is structured as a 23x23 grid. Three distinct maze layouts (`layout1`, `layout2`, `layout3`) are defined within the `initMaze` function. These layouts use integer codes: ‘1‘ for walls, ‘0‘ for traversable paths, and ‘3‘ for special path tiles where items like cheese or power-ups are not allowed to spawn (though they are visually identical to paths). A tunnel feature on row 11 allows horizontal wrapping. The `currentLevel` variable tracks progression, and `initMaze(currentLevel)` loads the appropriate layout.

3.2 Characters & Sprites

Custom sprites were designed for game entities and are rendered using OpenGL primitives. The initial visual design for characters was aided by an external shape-drawing tool that exported OpenGL primitive code. This code was then integrated into custom drawing functions (e.g., `drawCustomMouse`, `drawCustomCat`) where OpenGL transformations (`glPushMatrix`, `glPopMatrix`, `glTranslatef`, `glScalef`) are applied to position and scale them correctly within the game grid.

Note: An external tool was utilized for the initial design of character shapes, which exported basic OpenGL drawing commands. These commands were then integrated and transformed within the game’s C++ code.

3.2.1 Player (Mouse)

The player controls the mouse, aiming to collect all cheese pieces.

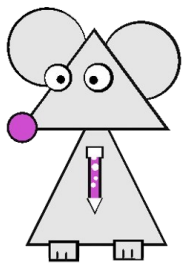


Figure 1: The player-controlled Mouse sprite, composed of several geometric primitives.

3.2.2 Opponent (Cat)

The cat is AI-controlled, using BFS to find the shortest path to the player.

Figure 2: The AI-controlled Cat sprite, also constructed from basic shapes.

3.3 Collectibles & Power-ups

3.3.1 Cheese

The primary objective: collect all cheese pieces to advance to the next level.

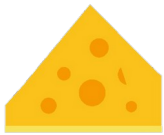


Figure 3: A cheese collectible. Collecting all cheese pieces completes the level.

3.3.2 Cat Slowdown Power-up

A blue, sparkling item that temporarily slows the cat down, giving the player an advantage.

Figure 4: The Cat Slowdown power-up item, visually indicated by a sparkling effect.

3.4 Game Mechanics

3.4.1 Game States

The game flow is managed by an `enum GameState`. The `display()` and `keyboard()` functions adapt their behavior based on the `currentGameState`.

3.4.2 Heads-Up Display (HUD)

Rendered using `drawText`, the HUD shows the current level, total accumulated score, and remaining cheese. It also displays ”CAT SLOWED!” when the power-up is active.

3.4.3 Difficulty Scaling

Cat speed (`currentCatDelay`) decreases (speeds up) as cheese is collected, following a non-linear curve.

3.5 Core OpenGL Concepts Used

- Windowing and Callbacks (GLUT)
- 2D Orthographic Projection (`gluOrtho2D`)
- Matrix Stack and Transformations (`glPushMatrix`, `glTranslate`, `glScale`)
- Drawing Primitives (`GL_QUADS`, `GL_TRIANGLES`, `GL_TRIANGLE_FAN`)
- Color Management (`glColor3f`)
- Double Buffering & Smooth Animation
- Blending for Transparency
- Anti-Aliasing
- Bitmap Text Rendering

534

5 Additional Screenshots

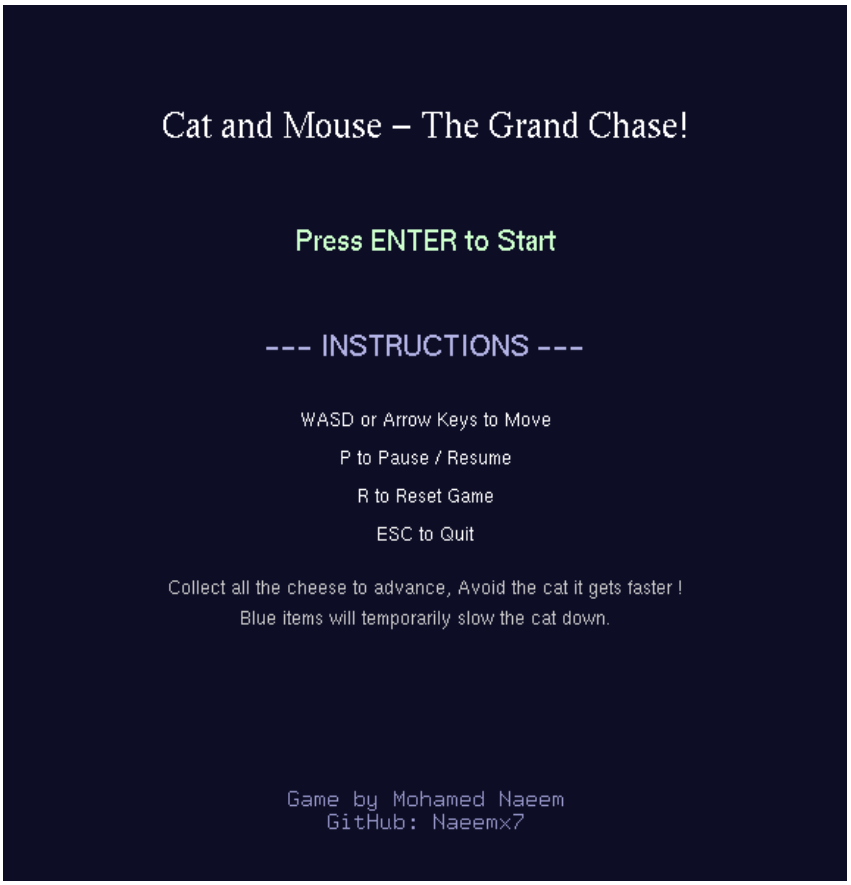


Figure 5: The initial Start Menu screen displaying the game title and control instructions.

Figure 6: Gameplay action on Level 1. The player (mouse) navigates the maze, collecting cheese (yellow triangle) while avoiding the AI-controlled cat.

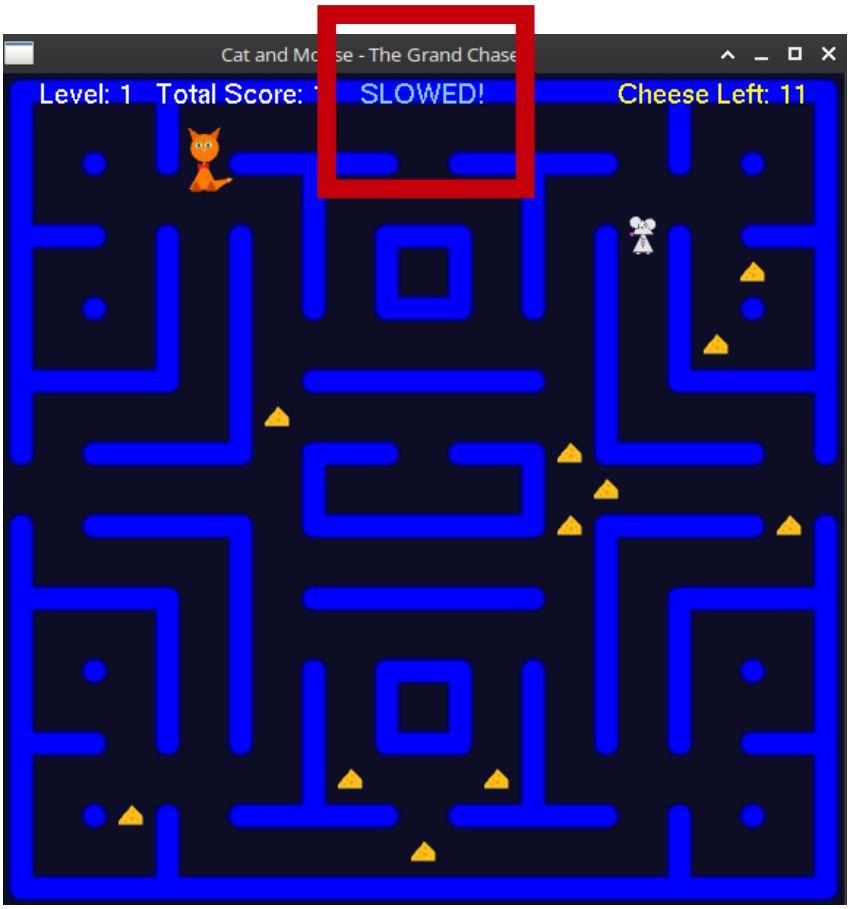


Figure 7: The Heads-Up Display (HUD) showing ”CAT SLOWED!” in the top center after the player collects a blue power-up item, indicating the cat’s movement is temporarily reduced.

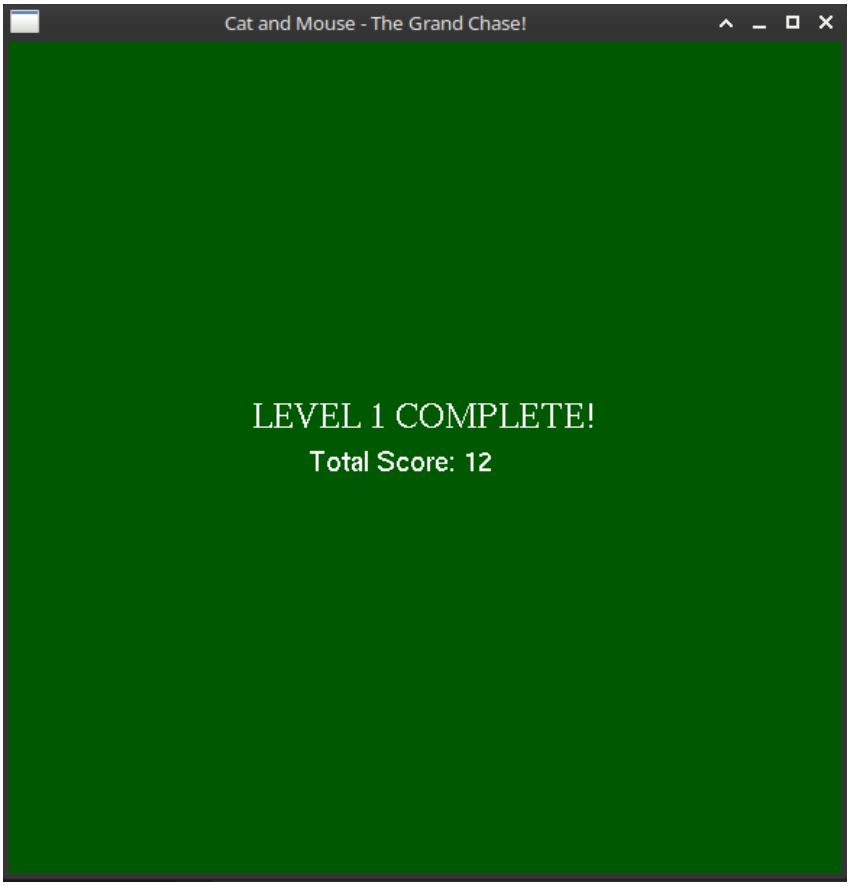


Figure 8: The ”Level Complete” screen, which appears after all cheese on a level has been collected, displaying the player’s total score.

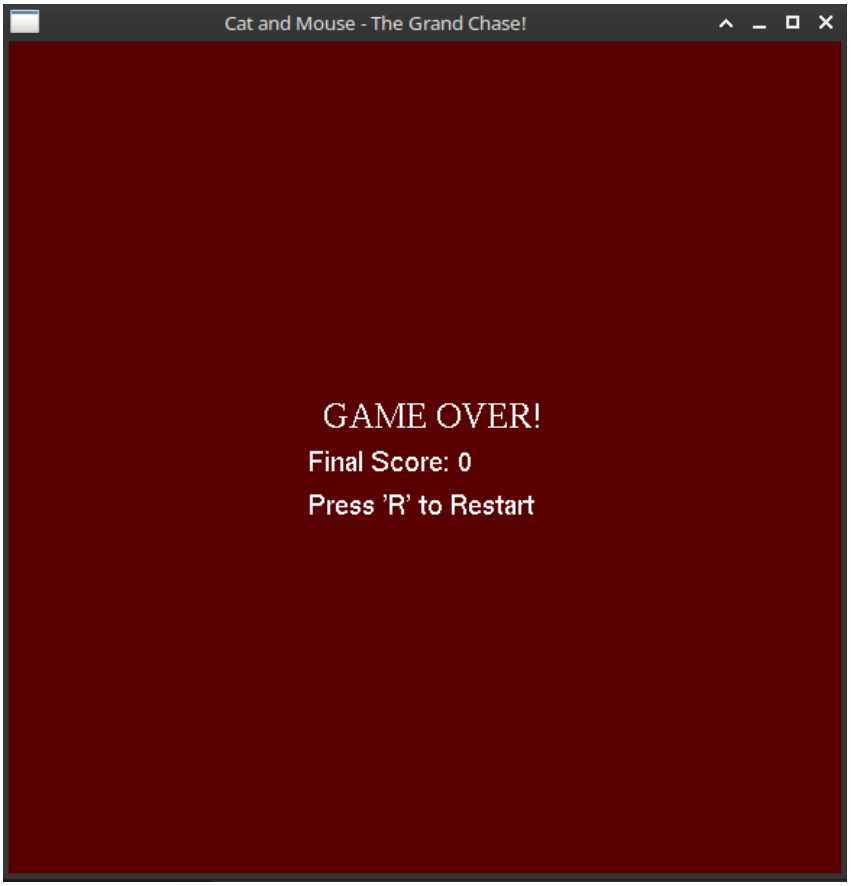


Figure 9: The ”Game Over” screen, shown when the cat catches the player, displaying the final score.