Tetris Game

Course Name: Data Structure and Algorithm II
Course Code: 0611CSE 232

By

Md. Mashakur Ruf Mueid
ID: CSE 07808403

Madhure Rani Mondal
ID: CSE 07808406

Shoisob Jahan Shaikat
ID: CSE 07808408

Ripa
ID: CSE 07808424

Course Teacher: Nusrat Jahan Farin
Lecturer

Department of Computer Science and Engineering

Stamford University Bangladesh

Submission Date: 08.11.23

Contributions:

No	Sector	Name	Contribution
01	Coding	• Madhure Rani Mondal • Shoisob Jahan Shaikat	• 80% • 20%
02	Report	• Md. Mashakur Ruf Mueid • Madhure Rani Mondal • Ripa	• 85% • 10% • 5%
03	Testing	• Madhure Rani Mondal • Ripa	• 60% • 40%

Abstract:

Tetris is a popular computer game that is enjoyed by people all over the world. It is a very common and most played game in the world. Previously, many developers have developed Tetris in many ways. With many modifications and features. The purpose of this project is to implement the Tetris concept with some modifications. We will try to make the game more attractive by doing some modifications. We implemented this project using C++ programming language. We used the Raylib library. The data is stored in two-dimensional format, and the two numbers are converted to tetromino shapes. The result is a game of Tetris with four different options for falling tetrominos , all of which meet in the center of the game board. And if there is a match in the row the whole row will disappear. This is the basic function of the game.

This game includes scoring system and collusion of tetrominos (blocks). 2D graphics is one of the most crucial parts of this project. We will explain how the game is build, the history and background of Tetris and working procedure throughout this report. The report will shortly describe how Raylib works in order to create 2D games. Testing and our limitations will also be a part of this report as well as the future aspect of this project.

Table of Contents

List of Figures:

1: Introduction:

1.1: Motivation:

There are many arcade games in the world. But we choose Tetris because of its simple gameplay and game design. As it is one of the most played games in the world. We have also played this game many times. But we didn't know how the game works and built. As a result, we chose Tetris to discover the working procedure of the game.

1.2: Objectives:

1. To develop the game using simple C++ and Raylib library.
2. To develop a fully functional Tetris game.
3. Add something new to the game.
4. Make the game attractive and enjoyable.

1.3: Tetris:

Tetris is a classic and popular puzzle video game. It was firstly created by Alexey Pajitnov in 1984. Pajitnov was a computer engineer of Russia. It has become the most played and addicted game throughout the years, because of its simple gameplay and user interface. This Tetris game has been developed in many ways by many developers throughout the years. The primary objective of this project is to develop a Tetris game using C++ programming language and Raylib library.

1.4: C++ Programming Language :

C++ is one of the world's most popular programming languages. It is an object-oriented programming language which gives a clear structure to programs and allows code to be reused, lowering development costs. C++ is a cross-platform language that can be used to create high-performance applications. C++ was developed as an extension of the C programming language and both languages have almost the same syntax.

1.5: Raylib Library:

Raylib is a programming library. It is used for enjoying video games programming. It is a free, open source and easy to use platform. Raylib is used for real-time 2D and 3D graphics rendering and game development. It has no fancy interface, no visual helpers and no guide tools or editors. The liberty is just for rendering code into visual gameplay.

2: Literature Review:

2.1: History of Tetris:

Tetris is a tile-matching video game developed in 1984 by Russian software engineer Alexey Pajitnov **(1).** It quickly turned into a popular culture icon. It has become the most played game in the world. Tetris quickly gained popularity among Pajitnov's colleagues and friends. The game began to spread through the Soviet Union and eventually to other Eastern European countries. Tetris made its way to the West when it was discovered by Hungarian game developer Andromeda Software. Andromeda began distributing Tetris on a variety of platforms. Tetris gained enormous popularity when it was bundled with the Nintendo Game Boy **(2)** , which was released in 1989.The rights were reverted to Pajitnov in 1996, who co-founded The Tetris Company with Henk Rogers to oversee licensing, after a significant period of publication by Nintendo.

2.2: Current Situation:

Tetris is the most ported video game in history. It has been played and published on almost over 65 different platforms. It has sold over 495 million copies worldwide across various platforms. Tetris has been recognized by Guinness World Records for various achievements, including being the "Most Ported Video Game" and the "Best-Selling Video Game." (3) In a recent review of the literature (Mayer, 2014), students who were assigned to play Tetris did not show substantially better gains than students who did not play Tetris on an array of tests of spatial cognition (median d = .04 based on 15 experimental comparisons) (4)

3: Implementation:

3.1: Steps:

For develop Tetris Game, we need follow some step such as-

1. Set up the Game Loop.

2. Creating the Grid.

3. Create the Blocks.

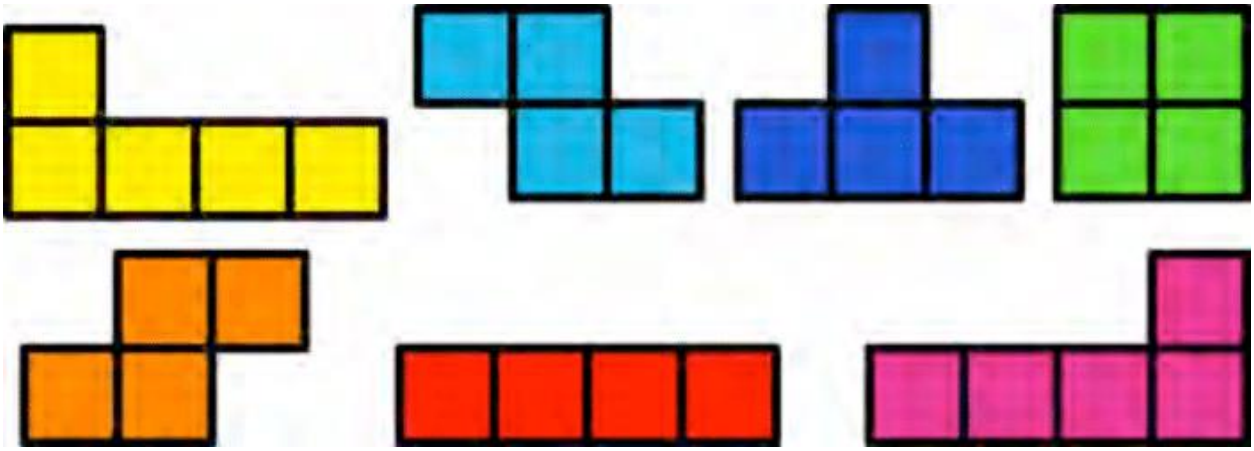

Figure-01

4. Move the Blocks.

5. Rotate the Blocks .

6.Checking for collisions.

7. Checking for completed rows.

8. Game over.

9. Create User Interface.

10. Ad Score.

11. Add Next Block .

12.Add Sounds.

If we follow this procedure creating the game will be fairly simple.

3.2: Configuration Code:

Here, is the algorithm for our Tetris game that was implemented using C++:

1. Initialization:
   - Initialize Raylib and set up the game window.
   - Load fonts, music, and sound effects.
   - Create a game instance (Game object).

- Set the game variables, including the game board, current and next Tetriminos, and score.

- Start playing the background music.

2. Game Loop:

- Enter a game loop that runs until the game window is closed.

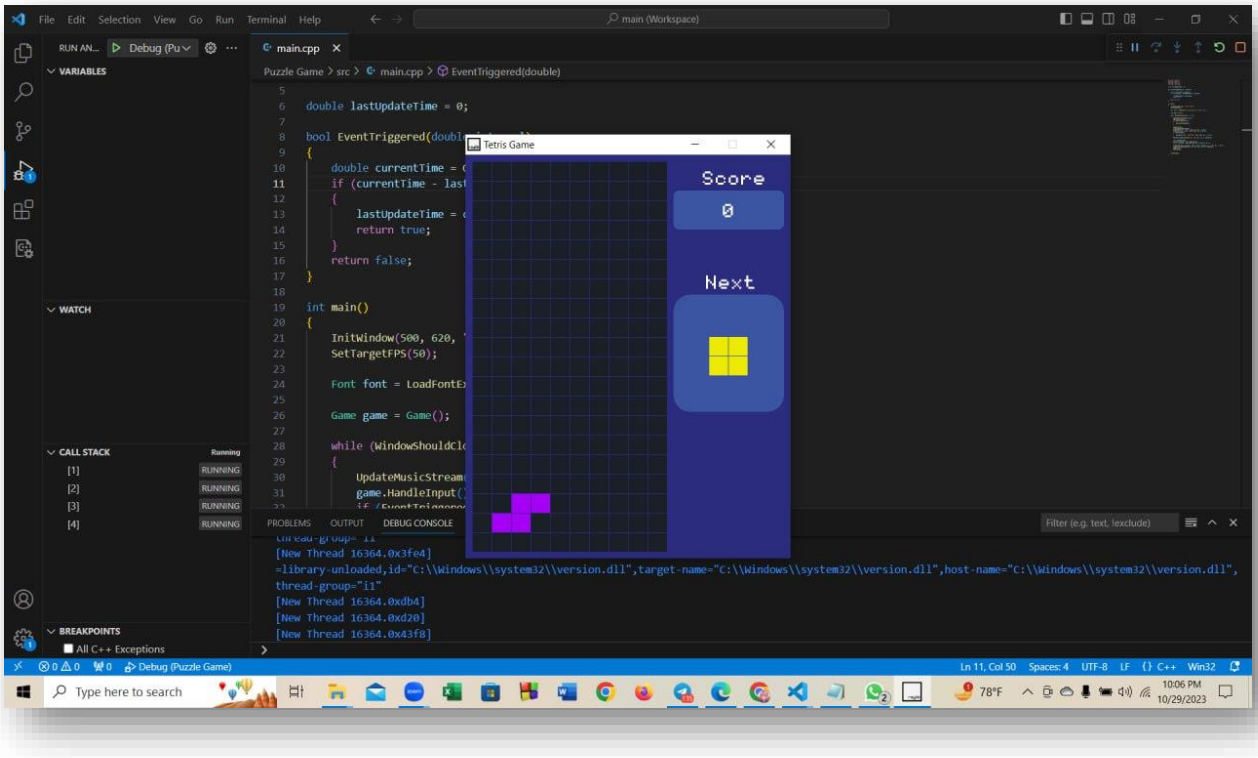

Figure:02

3. Event Trigger:

- Check if it's time to trigger an event (e.g., moving the Tetrimino down) based on a time interval.

4. Input Handling:

- Poll for player input, and handle the following key events:

- Move the current Tetrimino left.

- Move the current Tetrimino right.

- Move the current Tetrimino down and lock it if it reaches the bottom.

- Rotate the current Tetrimino.

5. Update Current Tetrimino:

- Update the position of the current Tetrimino on the game board.

- Check for collisions with the walls, other Tetriminos, or the bottom of the game board.

- If the Tetrimino can't move down further, add it to the game board and check for completed lines.

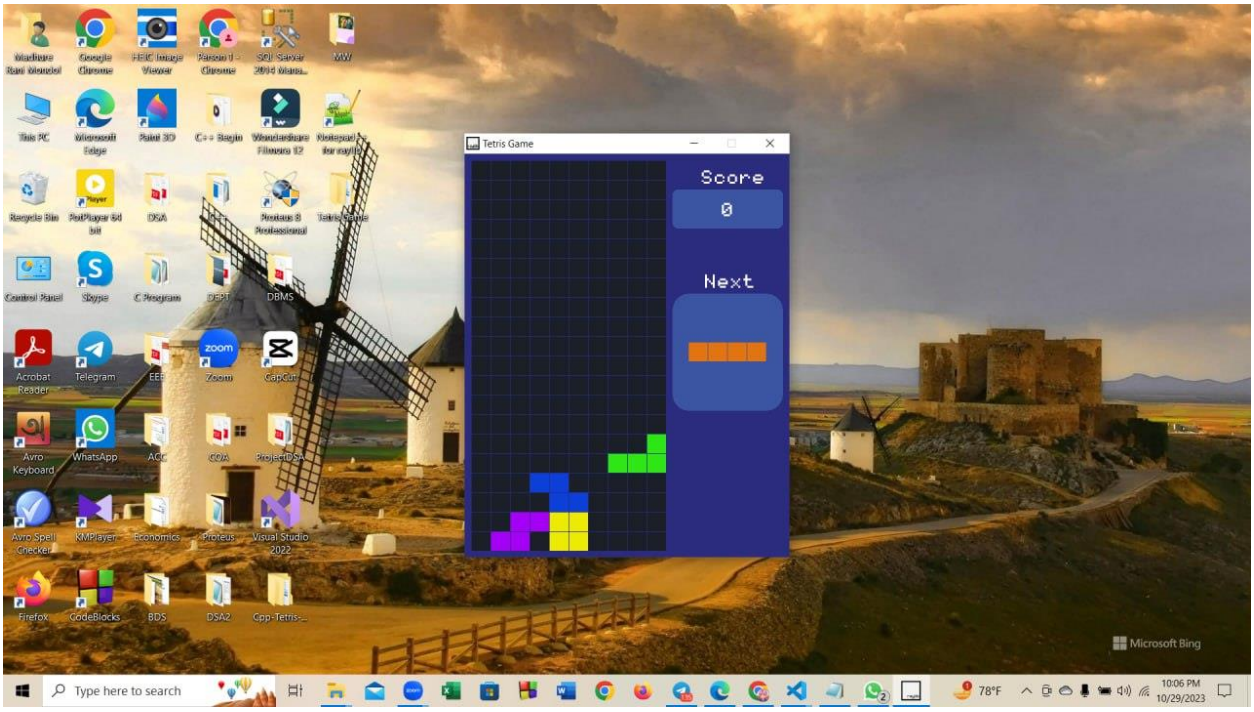

Figure: 03

6. Generate New Tetrimino:

- Create a new random Tetrimino and set its initial position at the top of the game board.

7. Check for Game Over:

- If a newly generated Tetrimino collides with existing blocks at the top of the game board, the game is over.

8. Scoring:

- Keep track of the player's score based on the number of lines cleared.

- Play sound effects when lines are cleared.

9. Rendering:

- Render the game board, current and next Tetriminos, score, and other game elements using raylib's functions.

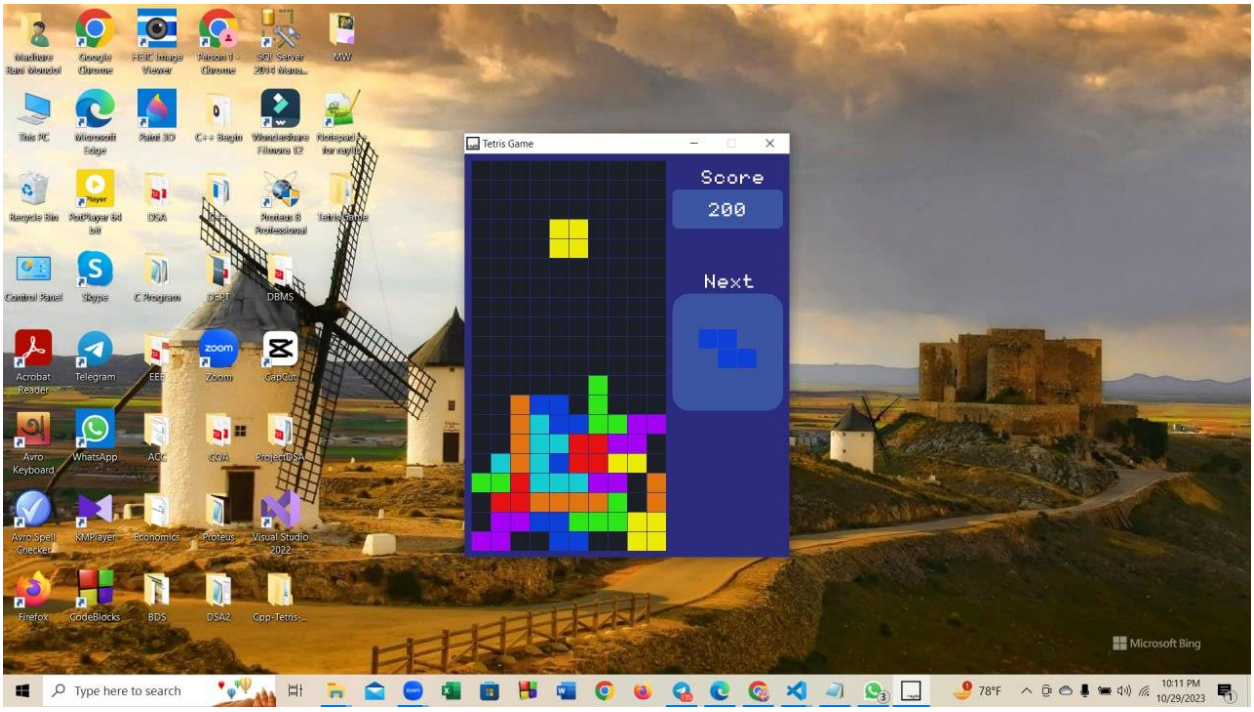

Figure: 04

10. Game Over Handling:

- When the game is over, display a "GAME OVER" message and the player's final score.

- Provide an option to restart the game if a key is pressed.

11. Clean Up:
    - Unload resources, such as fonts, music, and sound effects.
    - Close the audio device.

12. Main Function:
    - In the main function, initialize the game, create the game window, and start the game loop.

3.3: Pseudocode:

```
# Import necessary libraries and headers
include "raylib.h"
include "game.h"
include "colors.h"
include "position.h"
include "grid.h"
include "block.h"
include <iostream>

# Initialize variables
double lastUpdateTime = 0
bool EventTriggered(double interval)

# Initialize the game window and fonts
InitWindow(500, 620, "raylib Tetris")
SetTargetFPS(60)
Font font = LoadFontEx("Font/monogram.ttf", 64, 0, 0)

# Create a Game instance
Game game = Game()
```

```
# Main game loop
while (WindowShouldClose() == false)
{
    UpdateMusicStream(game.music)
    game.HandleInput()

    # Update the game state
    if (EventTriggered(0.2))
    {
        game.MoveBlockDown()
    }

    # Render the game elements
    BeginDrawing()
    ClearBackground(darkBlue)
    DrawTextEx(font, "Score", {365, 15}, 38, 2, WHITE)
    DrawTextEx(font, "Next", {370, 175}, 38, 2, WHITE)

    if (game.gameOver)
    {
        DrawTextEx(font, "GAME OVER", {320, 450}, 38, 2, WHITE)
    }

    DrawRectangleRounded({320, 55, 170, 60}, 0.3, 6, lightBlue)
    char scoreText[10]
    sprintf(scoreText, "%d", game.score)
    Vector2 textSize = MeasureTextEx(font, scoreText, 38, 2)
    DrawTextEx(font, scoreText, {320 + (170 - textSize.x) / 2, 65}, 38, 2, WHITE)
    DrawRectangleRounded({320, 215, 170, 180}, 0.3, 6, lightBlue)
    game.Draw()
```

```
    EndDrawing()
}
# Clean up resources and close the game window
CloseWindow()
```

3.4: Time Complexities:

1. Tetromino Movement/ Falling Blocks movement: 0(1)
2. Line Clearing: O(R*C); here, R = number of rows and C = number of columns
3. User Input Handling/Rotating Tetrominoes: O (1)
4. Game Screen Rendering: O(R*C); here, R = number of rows and C = number of columns
5. Generating random Tetrominoes: O (1)

3.5: Space Complexities:

1. Game Grid: O(R*C); here R = number of rows and C = number of columns
2. Storing the Tetrominoes: O (1)
3. Storing Colors and Graphics: O (1)
4. Storing the game state: O (1)
5. User Interface: O (1)

4: Result Analysis:

01. A fully functional Tetris game implemented in C++ with Raylib.
02. Smooth and responsive gameplay.
03. A scoring system that player will find attractive.
04. An easy-to-use interface.

5: Conclusion:

5.1: Limitations:

The project was a success. We have successfully completed the game. We were able to run the game on Raylib. There were some difficulties while implementing the game code. We were able to get help from various source. There are some limitations of out project. It is not a fully developed game but rather a prototype. As a result, the procedure of the game is simple. There are no highly developed graphics and user interface. We can't load a new game without completing the current round.

5.2: Future Work:

Despite the limitations of this game, we are planning to do more development on this game in the future. We will add some features like a high and different level system, game difficulty management, higher graphics, coin/token gaining system and extra event feature. Also, we are planning to turn this game into a fully functional app for different offline and online platforms.

6: Reference:

01. Demaine, E. D., Hohenberger, S., & Liben-Nowell, D. (2003). Tetris is hard, even to approximate. In *Computing and Combinatorics: 9th Annual International Conference, COCOON 2003 Big Sky, MT, USA, July 25–28, 2003 Proceedings 9* (pp. 351-363). Springer Berlin Heidelberg.

02. (https://tetris.com/history).

03. (https://www.guinnessworldrecords.com/world-records/62522-tetris)

04. Pilegard, Celeste, and Richard E. Mayer. "Game over for Tetris as a platform for cognitive skill training." *Contemporary Educational Psychology* 54 (2018): 29-41.