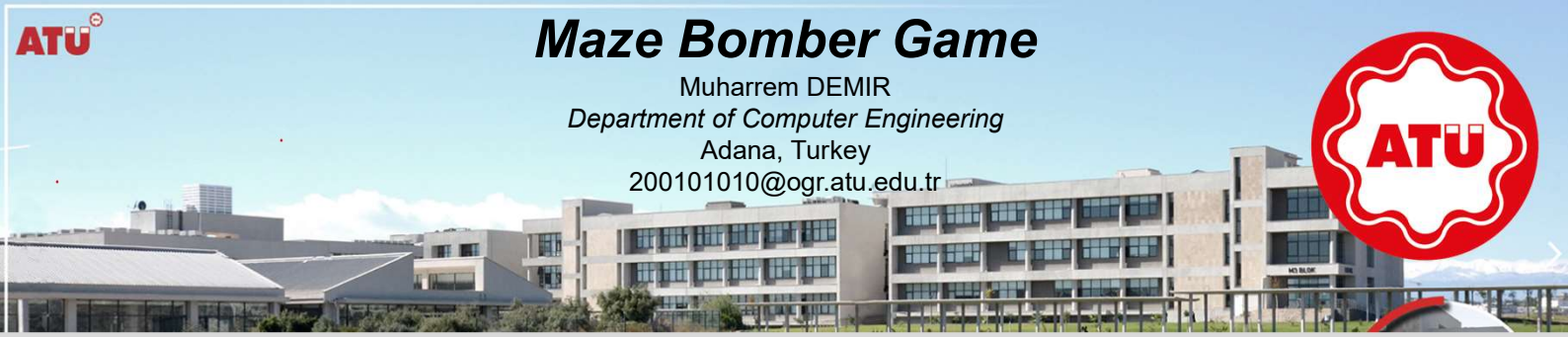
Maze Bomber Game

Muharrem DEMIR
Department of Computer Engineering
Adana, Turkey
200101010@ogr.atu.edu.tr



Introduction

The primary objective of this study is to develop a game called **Maze Bomber**, inspired by Bomberman. The project aims to provide a gaming experience that supports both single-player and multiplayer modes. Throughout the development process, elements such as **strategic AI bots, dynamic map generation mechanisms, and power-up systems** have been integrated to offer players a competitive and interactive experience.
This study explores the **practical applications of algorithmic problem-solving techniques in game development** by utilizing advanced **pathfinding algorithms and real-time event management**.
It has been observed that **classic arcade games like Bomberman** have significantly influenced AI-driven gaming environments. In modern implementations, *pathfinding algorithms such as A (A-Star)** play a crucial role in improving decision-making in grid-based games.
For this project, the **I-See-Bytes library** has been used for graphics processing. Developed as an alternative to OpenCV, this library combines Python's ease of use with C++ performance to offer a robust graphical user interface. Unlike OpenCV, which lacks comprehensive GUI support, ICB provides both beginner-friendly accessibility and enhanced efficiency for professional developers. Additionally, it simplifies **runtime-determined matrix structures and optimized data processing interfaces**, reducing complexity in graphics-based game development.
In conclusion, this project aims to develop **modern approaches for Bomberman-style games** while enhancing game mechanics with AI-driven strategies. The system represents a game model with an optimized **AI infrastructure, improved player interactions, and multiplayer support**, providing an engaging and dynamic gaming experience.

Core Technologies

C++

The game is developed using **C++**, which offers high performance, efficient memory management, and real-time processing capabilities. C++ is widely used in game development due to its ability to handle low-level system operations while maintaining flexibility for complex game mechanics.

I-See-Bytes Library

For graphics processing, the **I-See-Bytes library** is utilized as an alternative to OpenCV. Unlike OpenCV, which relies on predefined matrix structures and complex macro definitions, ICBYTES simplifies matrix operations by introducing a fluid data structure that dynamically adapts to different types of data. This feature eliminates the need for compile-time data type specifications, making it easier to manipulate matrices in real time. Additionally, ICBYTES integrates an intuitive GUI system, enabling seamless window management, button creation, text input handling, and real-time image rendering without requiring external dependencies. The library is designed for cross-platform compatibility, ensuring smooth execution on both Windows and Linux systems. Its efficient memory usage and low-latency processing make it particularly suitable for real-time applications such as game development.

Windows Threads API

To manage **asynchronous tasks** such as player movements, explosion effects, and AI bot decision-making, the **Windows Threads API** is employed. This allows for parallel execution of multiple game components, ensuring a smooth and responsive gameplay experience. By utilizing multithreading, the game can handle simultaneous events without performance bottlenecks, improving real-time interactions and reducing input lag.

Crono Library

The **Chrono library** is used for precise time-based operations, ensuring accurate scheduling of **power-up activations, trap delays, and explosion effects**. This guarantees smooth gameplay mechanics and balanced event timing, preventing inconsistencies in real-time interactions.

Algorithms

The game employs a *custom-built A pathfinding algorithm** designed specifically for dynamic, grid-based navigation. Unlike standard A* implementations, this version continuously adapts to changing game conditions, such as **bomb placements, destructible walls, power-up spawns, and opponent movements**. AI bots utilize real-time recalculations to avoid hazards, ensuring they do not move into explosion zones. The algorithm assigns **danger values** to tiles at risk, prioritizing escape routes and safe paths while also optimizing for strategic positioning.
AI bots operate based on a **multi-objective decision-making system**, adjusting their behavior according to the game state. If a nearby explosion is detected, they switch to a **defensive strategy**, finding the quickest escape route. Conversely, in offensive mode, the AI places bombs in locations that restrict an opponent's movement, **effectively trapping them**. The AI also prioritizes **power-up collection**, selecting upgrades based on its current needs, such as speed boosts for evasive play or increased bomb range for aggressive tactics. Additionally, it tracks previously triggered traps, avoiding areas that may hinder movement.

The **heuristic function** of the algorithm is enhanced beyond basic distance calculations by incorporating **risk evaluation, strategic positioning, and escape potential**. Instead of simply finding the shortest path, the AI evaluates multiple factors, such as bomb detonation timing, opponent behavior, and available safe zones, making its movement more intelligent and unpredictable. This **adaptive AI system** creates a **challenging and competitive** gameplay experience, ensuring that bots respond dynamically to player actions, enhancing overall game strategy and engagement.

Conclusion and Future Work

Conclusion
Maze Bomber modernizes the classic Bomberman-style gameplay with strategic elements, advanced AI, dynamic map generation, and diverse power-up systems. It offers both single-player and local multiplayer modes, providing a competitive gaming experience. Future plans include online multiplayer support using ENet, high-quality audio effects with FMOD, and continuous performance improvements. Cross-platform support and enhanced animations are also part of the roadmap.
Future of Works
The game will be expanded with online multiplayer functionality using ENet for global connectivity, along with matchmaking and ranking systems for balanced competition. An in-game voice chat system will enhance multiplayer communication, and cross-platform support will broaden player accessibility. Visual and sound quality will be improved through refined animations and FMOD audio engine integration. Continuous performance optimization will focus on memory management, latency reduction, and resource usage refinement to maintain a polished and competitive gaming experience.

References

- Otoidrak. (n.d.). *I-See-Bytes: A Simplified C++ Library* (Assoc. Prof. Dr. Ibrahim Cem Baykal, Author). https://otoidrak.com/
- Yılmaz, M. (2020). *Game Development Technologies*. Oyun Yayıncılığı Publishing.