

Project Notebook

Rebomb

Yaxuan Dai
Mahdis Sabzevarzadeh
Miguel Trasobares
Jialin Yang

Master Practical Course Games Engineering
Chair of Computer Graphics and Visualization
Technical University of Munich
Munich, Germany

December 3, 2024

Contents

Milestone 1: Formal Game Proposal	4
1 Game Description	4
1.1 Gameplay	4
1.2 Game Mechanics	5
2 Technical Achievement	6
2.1 Game Snapshots Recording	6
2.2 Multiplayer Functionality	6
2.3 Procedural Generation	7
2.4 Explosions via Particle Simulation	7
2.5 Environment Change	7
3 "Big Idea" Bullseye	8
4 Development Schedule	8
4.1 Layered development Description	8
4.1.1 Functional minimum	8
4.1.2 Low target	8
4.1.3 Desirable target	9
4.1.4 High target	9
4.1.5 Extras	9
4.2 Timeline	10
5 Assessment	11
Milestone 2: Game Prototype	13
6 Description	13
7 Experience	15
7.1 Strategy and Interaction	15
7.2 Weapons and Items	15
8 Revisions to Game Idea	15
Milestone 3: Interim Demo	17
9 Task Progression	17
10 Challenges	18
10.1 Input System: Keyboard Splitting	18
10.2 Time Travel Feature: Snapshots and Current States	18
10.2.1 Data Structure Definition	18
10.2.2 Taking Snapshots	19
10.2.3 Loading Snapshots	19
10.3 Item Interaction	19
10.4 Chain Reaction	19

11 Design Revision **20**
 11.1 Scalability Considerations 20
12 Progress in Photos **20**

Milestone 1: Formal Game Proposal

1 Game Description

We are planning to create a multiplayer turn-based game that contains time travel features, procedurally generated maps, and explosions via particle simulations. We draw inspiration from popular titles such as *Bomberman* for showing blast in maze, *Quantum League* or *Life is Strange* for their time travel features and *Candy Crush Saga* for the cascaded effects. *Rebomb* demonstrates the theme "Chain Reaction" in its mechanics by explosions of bombs. An active bomb affects nearby passive bombs, causing them to trigger and explode. Additionally, flammable objects such as oil buckets, haystacks, wooden fences and trees will be available on the map and players can use these items to their advantage. The game will be designed on several levels. In each round the player has resources that will be carried in the next rounds. The player gains progressively more resources for the chain reactions. Later rounds players can unlock powerful and more costly bombs with their survival bonus.

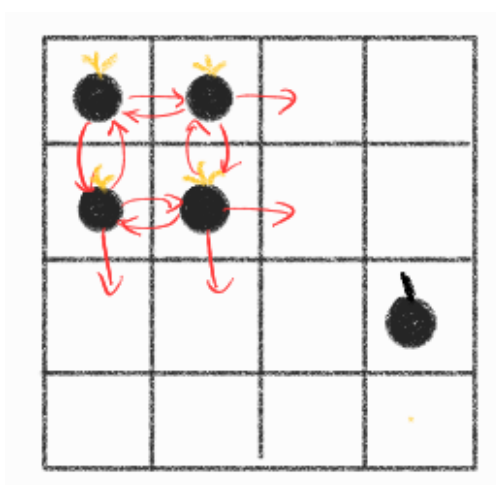


Figure 1: Chain Reaction.

1.1 Gameplay

Rebomb is a turn-based, maze-based multiplayer game that can be played with 2 or 4 players. The players can only move horizontally or vertically in a 2.5D map containing obstacles and collectibles. To win the game, players must eliminate all opponents and remain the last one standing. Players can get killed once they get caught up in a bomb's explosion, including their own. The bombs have to be strategically placed to destroy obstacles or kill other players. They explode in horizontal and vertical directions after some time causing a cascaded effect. Different types of bombs vary in their impact radius, with each type affecting the explosion distance to a greater or lesser extent.

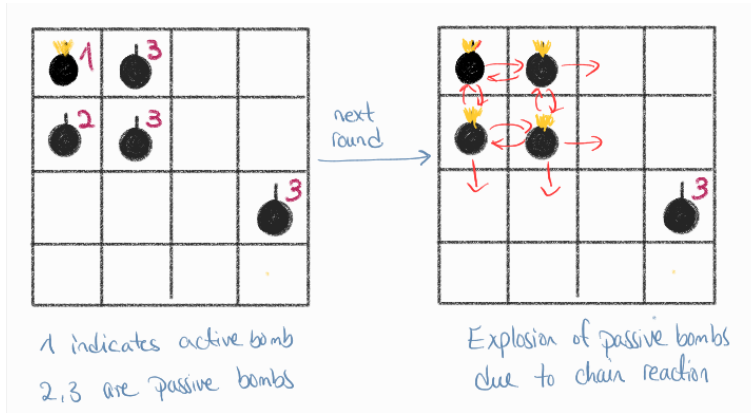


Figure 2: Passive and Active Bomb differentiation.

1.2 Game Mechanics

The player starts the game with fixed and limited resources. Resources can be measured by the number of coins a player holds. Placing the bombs costs coins. Various bombs have different costs. In each round, the player can move only n tiles in horizontal or vertical directions and place their bombs. Resources can be gathered by items on the map, by killing other players, or by designing an in-game interest system, so that the player gains more resources in the upcoming rounds, once they survive the round. The map consists of collectibles, such as hourglasses for time traveling, power-ups for stronger bombs, or extra coins. Once an hourglass is collected the player may choose to time travel back k rounds. In the context of this game, time travel means rewinding to a previous state of the map, where the position of the players, the resources, and all bombs, except for the last placed bombs by each player, will be reverted for all the players. This way the player with time-traveling ability has a chance to strategically change the course of the game in their own favour. Figure 3 demonstrates how time travel can be in a 1 versus 1 gameplay. Note that the same mechanic can also be applied when 4 players are in the game. In this example one bomb costs two coins and the players can move up to 3 tiles in each round.

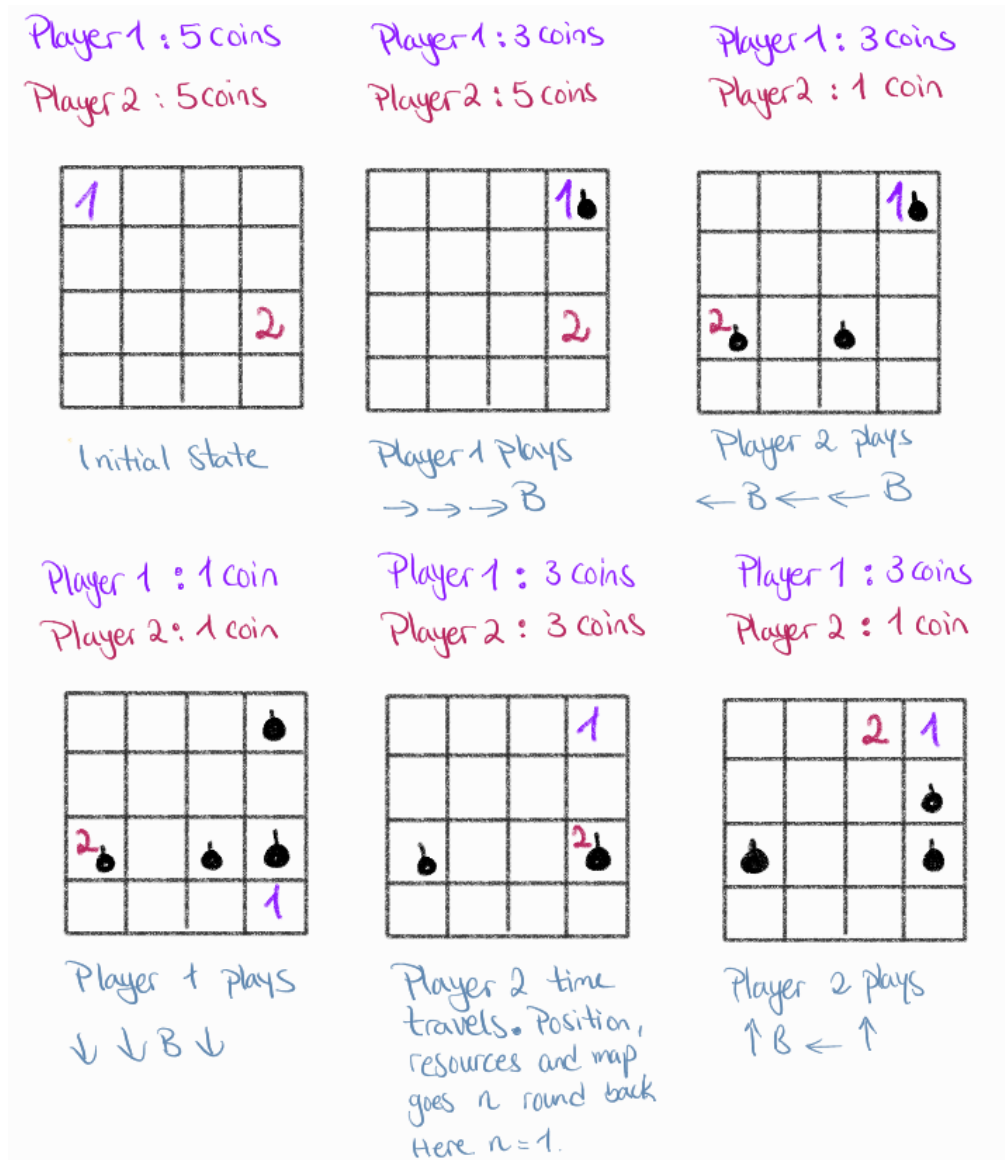


Figure 3: Time Travel Example.

2 Technical Achievement

2.1 Game Snapshots Recording

Our main technical achievement will be the implementation of a game state recording system to enable time travel mechanics. The system will save snapshots of key states including all necessary information to replay the game (movements, actions, objects, environment, etc). This will allow players to interact with previous game experiences, allowing them to explore alternative strategies and making time travel a functional and engaging game mechanic.

2.2 Multiplayer Functionality

Our game will support a multiplayer mode, allowing players to compete in the same scenario in real-time. This feature will require careful management of game states to ensure a minimal latency between devices so



Figure 4: Procedural generation in Minecraft.



Figure 5: Different particle effects for explosions.

that game states are synchronized in every device. The multiplayer setup will also integrate seamlessly with the time travel mechanics to offer unique multiplayer interactions and experiences.

2.3 Procedural Generation

Using seed-based procedural generation, we will generate scenarios with unique configurations at each gameplay. This method provides players with new unseen environments each session while ensuring playability with a set of predesigned rules that control randomness. This simple algorithm also allows for the scalability of the map generation. The seed value will allow players to replay specific map setups that they consider interesting and share them with other players, encouraging game replayability.

2.4 Explosions via Particle Simulation

We will pay special attention to visual dynamic effects from explosions. Each explosion will be generated using particle effects and based on parameters such as brightness, color, shape, or velocity, allowing for distinct patterns and range effects. The particle properties will be tied to different types of explosives and objects, allowing the player to associate the effects with the objects used and better visualize the game state.

2.5 Environment Change

To visually enhance the passage of time within a game session, we will incorporate environmental changes, such as shifting sunlight, moving shadows, or changes in lighting as cues for time progression. This element will act as a way finder, helping players to orient themselves in the temporal dimension while also improving immersion to reflect player decisions. These changes will be achieved through dynamic lighting adjustments based on the game's internal clock and scene configurations.

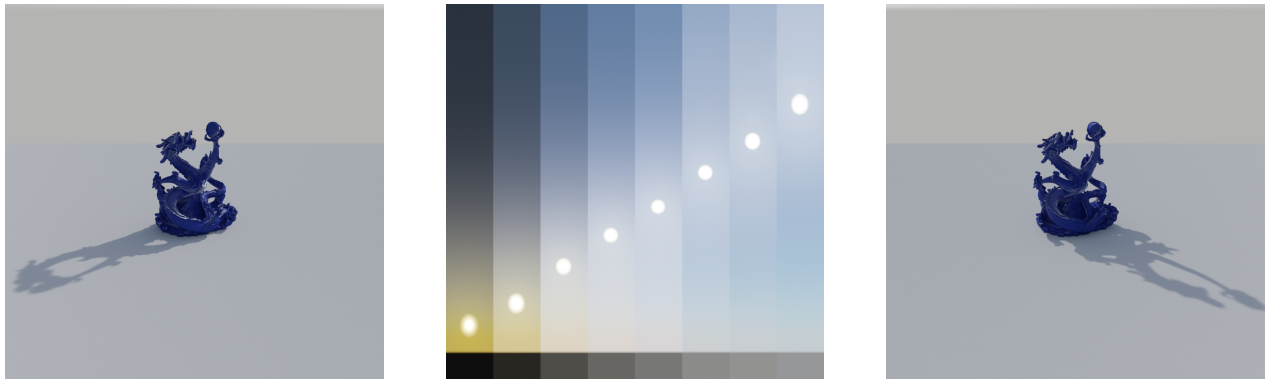
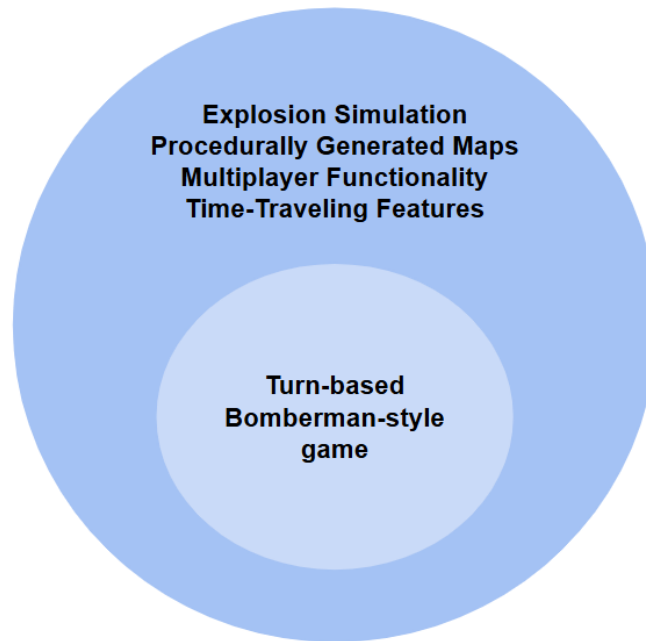


Figure 6: Change on shadows due to different solar azimuth.

3 "Big Idea" Bullseye



4 Development Schedule

4.1 Layered development Description

4.1.1 Functional minimum

Our minimum goal is to deliver a playable turn-based Bomberman-style game with simple default maps and limited interactive elements. The initial focus will be on core mechanics to ensure a functional foundation.

- Player movement and item placement logic.
- Basic environment objects.
- A fundamental weapon system that includes active and passive bombs.
- A basic resource system for initial resources and survival bonus.
- Essential assets.

4.1.2 Low target

At this stage, we aim to enhance gameplay mechanics and introduce additional features to create a richer experience.

- Time travel mechanics.
- Weapon system: sequentially unlocks more powerful bombs and special items.
- A basic GUI with menus and gameplay options.
- Assets specifically for special items.

4.1.3 Desirable target

Building on previous stages, we aim to create a more dynamic and engaging game environment.

- Procedural map generation to increase replayability.
- Local multiplayer for competitive play.
- Enrich interactive elements in environments.
- Assets and GUI for a better user experience.

4.1.4 High target

- Character personalization and animation.
- Multiplayer functionality across multiple machines.
- Additional game mode (e.g. round-based).
- Fine-tune numerical settings to balance gameplay and improve mechanics.

4.1.5 Extras

- Explore more interesting time travel feature possibilities and realtime gameplay.
- Online multiplayer gameplay.

4.2 Timeline

According to our layered development description and milestones in this semester, a initial time schedule is illustrated in Figure 11 and Figure 8.

Date	Milestone	Week	Layer	Task	Expect Hours	Actual Hours
Nov 06-12	Prototype	1	Prototype	physical Prototype	4 * 5	
			Minimum	simple assets	5	
				simple map & static object	5	
				player move & item place	5	
				simple GUI	5	
Nov 13-19		2	Minimum	active & negative bomb	6	
				resourse system	6	
				interactive map & object	6	
				turn-based gameplay	6	
				version integration	4 * 4	
Nov 20-26		3	Low	specifical assets	8	
				time travel mechanism 1/2	8	
				weapon & interactive obje	8	
				full GUI	8	
				version integration	4 * 2	
Nov 27-Dec 03	Interim demo	4	Low	time travel mechanism 2/2	8	
				cascaded explosion refine	8	
				map generation 1/3	8	
			Desirable	local multiplayer 1/3	8	
				version integration	4 * 2	

Figure 7: Time schedule [1/2].

Dec 04-10		5	Desirable	map generation 2/3	8
				local multiplayer 2/3	8
				explosion effects 1/2	8
				more weapon and objects	8
				version integration	4 * 2
Dec 11-17		6	Desirable	map generation 3/3	6
				local multiplayer 3/3	6
				explosion effects 2/2	6
				numerical refine	6
				version integration	4 * 4
Dec 18-24		7	holiday(Dec 24-Jan 06)		
Dec 25-31		8			0
Jan 01-07	Alpha release	9			0
Jan 08-14		10	High	remote multiplayer	2 * 8
				assesment & bugfix	2 * 8
				version integration	4 * 4
Jan 15-21	Playtesting	11	High	remote multiplayer	2 * 8
				improve mechanics	8
				refine animation/character:	8
				version integration	4 * 4
Jan 22-28		12		remaining tasks & test	4 * 10
Jan 29-Feb 04	Final release	13	Extra	extra tasks & test	4 * 10

Figure 8: Time schedule [2/2].

5 Assessment

The game will combine strategic decision-making with explosive chain reactions, creating a dynamic and suspenseful experience. The time travel feature will add a unique twist, allowing players to strategically undo their moves, heightening both strategic depth and unpredictability.

The game is designed for players who enjoy competitive strategy games, especially fans of Bomberman, turn-based tactics, and games with elements of mind games or puzzle-solving. Players seeking multiplayer experiences with strategic depth and unique gameplay mechanics are expected to be particularly drawn to this game.

In this game:

- Players place bombs with an option to activate their fuses. Activated bombs explode after a delay of a few turns, while unlit bombs remain dormant and can be triggered by chain reactions from other explosions.
- The time travel feature allows players to revert all players and the environment to their state from some turns ago. However, the last bomb each player placed before turning back time will remain and explode according to its original timing.

This gameplay will introduce both offensive and defensive layers, encouraging strategic bomb placement and reactionary tactics.

1. **Engagement and Replayability:** Players should feel compelled to play multiple rounds and experiment with new strategies, especially utilizing the time travel mechanic.

2. **Balanced Mechanics:** A successful design should make the time travel and bomb chain reactions feel fair, with strategy prioritized over luck.
3. **Strategic Depth:** Players should have opportunities to improve with experience, discovering advanced tactics and evolving their strategies.
4. **Player Retention:** The game's success can be measured by a strong player base with high retention, indicating enjoyment and multiplayer appeal.

Milestone 2: Game Prototype

6 Description

We create a physical prototype in a 2D board that fully reflects the gameplay functionality of our game. Initially we propose the following rules for playing the game:

1. The board is set arbitrarily to a 8x8 grid of tiles. The tiles will contain an unbreakable walls, breakable walls or be empty.
2. Every tile can be occupied by a single element at the same time.
3. The game is played via sequential turns as in chess. First a player decides and plays his actions, then the next player plays his turn until every player has completed their actions.
4. In his turn, a player can move up to 3 tiles horizontally and vertically in any possible trajectory. A player cannot move into a tile occupied by any other element or player. Additionally, in his turn, a player can place a single bomb in any of the tiles it has visited. Every bomb is either active or passive and this is decided by the player at the moment he places on the board.
5. Active bombs have attached a counter that begins at 3 when placed. At the beginning of the player who originally placed the bomb, the counter is decreased by 1. When the counter reaches 0 the bomb explodes. The explosion reaches all the horizontal and vertical tiles that are at a distance of 1 tile from the bomb and the tile the bomb is at. All other bombs that are in tiles affected by the explosion also explode causing a chain reaction that get resolved at that same moment. The breakable walls that are affected become empty tiles. The players in affected tiles are eliminated and lose the game.
6. Passive bombs do not have a counter but when affected by an explosion explode as an active bomb.
7. One breakable wall will have hidden an hourglass item. When the tile is affected by an explosion, the hidden item will appear on that tile. The first player that moves over the item will acquire it and the item will disappear the board. At the beginning of his turn, a player can use the item, which is consumed, to perform a time travel of 3 turns in the past to the beginning of that turn. When this happens the board is reset to the exact state it was 3 turns before with the exception of the bomb both players might have played in the turn before performing the time travel, which are also carried in the state they are to the new board.
8. The game ends when a single player is alive and all the other players have been eliminated.

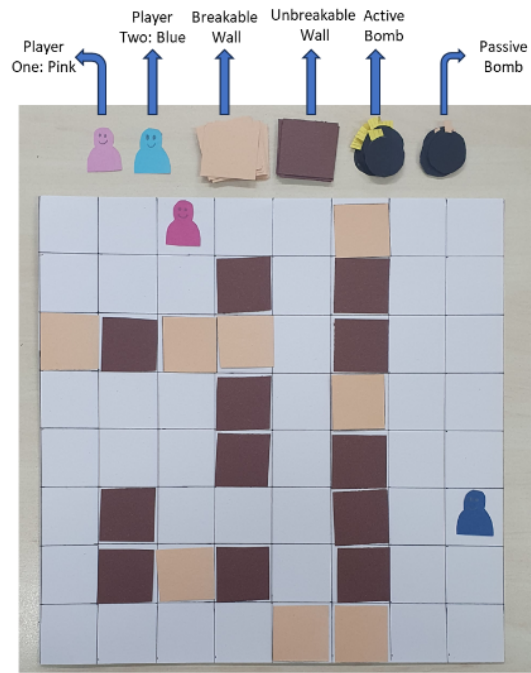


Figure 9: Paper Prototype - Round 0

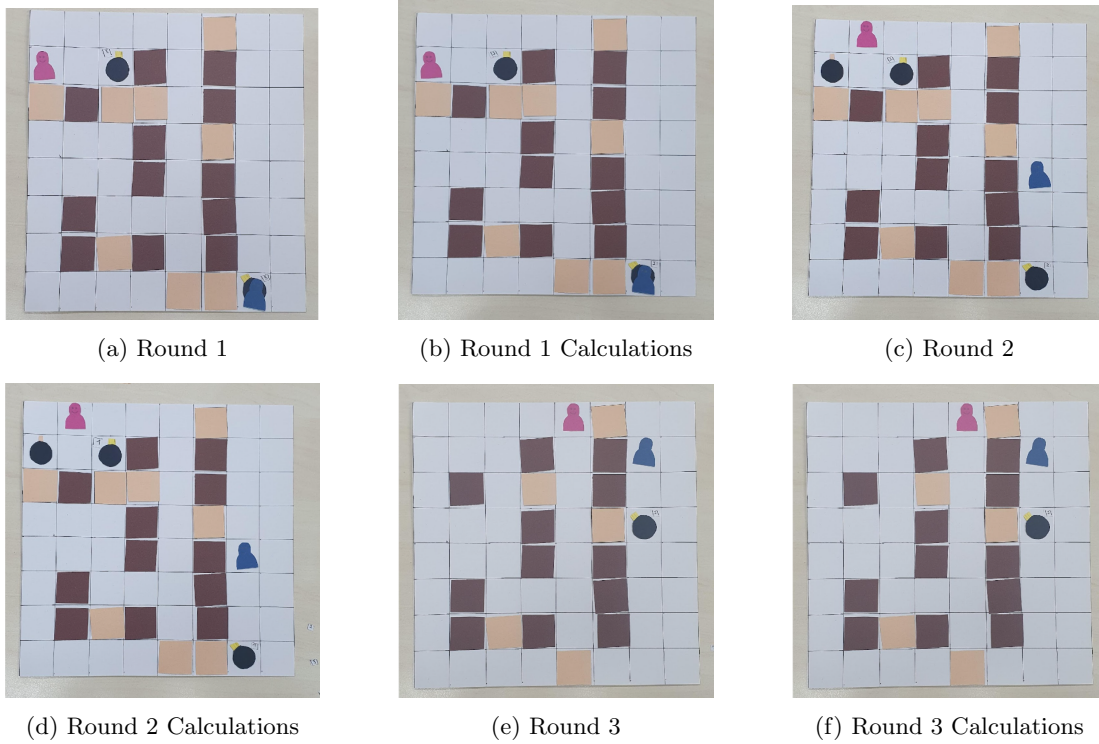


Figure 10: Gameplay demonstration of prototype

7 Experience

7.1 Strategy and Interaction

Throughout our play sessions, we observed that players are motivated to explore and develop various strategies based on our interaction and mechanics. These mechanics distinguished our game from the traditional action game Bomberman, leaning more towards a strategy-driven experience. Rather than relying on quick reflexes, players are encouraged to plan, anticipate, and outmaneuver each other.

- **Timing with Bomb Detonations:** the combination of explosion range, movement step size, and bomb countdown encourages players to carefully time their moves. Planning ahead becomes essential, as players must anticipate both their own and their opponents' actions to avoid explosions and capitalize on positioning.
- **Early Bomb Triggering:** players tend to place bombs early in the game to remove breakable walls and expand the battlefield. This strategy creates a dynamic environment where players have more options and tactical opportunities.
- **Trap Setting:** an integral aspect of gameplay is players attempting to trap each other by strategically placing bombs. By predicting and limiting opponents' movement options, players can increase the likelihood of an opponent getting caught in an explosion, making for tense, engaging moments.
- **Chain Reactions:** to maximize impact and maintain safe positions, players frequently set up bomb chains that trigger successive explosions. This strategy allows them to target opponents from a distance, adding a rewarding level of risk management.

7.2 Weapons and Items

The integration of our core items—the active and passive bombs, as well as the hourglass for time travel—adds both depth and breadth to the strategic possibilities. Each item introduces new layers of decision-making.

- **Impact of Bomb Collision:** when bombs are placed with collision volumes, they can act as barriers, restricting movement options and enhancing the trapping mechanics. Players are motivated to use bombs as obstacles to manipulate opponent movement.
- **Hourglass to Travel Back:** the hourglass item brings a unique twist to gameplay. When used, it rewinds recent turns while keeping the player's last bomb in place. This mechanism allows for surprise strategies, such as reversing a near-loss situation or enhancing a bomb trap.

8 Revisions to Game Idea

By creating the prototype, we are able to play the game, spot the advantages and disadvantages of our proposed design, and make modifications to our design.

The basic winning strategy of classical Bomberman is trapping the enemies in the bombs' explosion region. Our game should also take it as the key to winning, in ways of placing the bombs as obstacles, creating chained explosions, and using time travels or other potential items.

- **Map Design:** the map design appears to be harder than expected. During our test plays on the prototype, we found adding the number of breakable walls can help increase the suspense of the possession of the time travel feature but at the same time increase the length of the game. The unbreakable walls should be of a larger number but not continuous in a large size, to provide enough possibility of trapping enemies and encourage the use of chain reactions simultaneously.
- **Numerical Design:** as expected, the numerical design is crucial to tuning the desired player behavior. For example, a small explosion range encourages more blocking opponents using bombs which is not what we want from a chain-reaction-centered game, while a large one also kills the value of cascaded

explosions because single explosions are already powerful enough. On a current 8x8 prototype map with max. 3 tiles movement and max. place 1 bomb per player each round, we tested and found the sweet point explosion range of 2 that can hopefully highlight the potential of chain reactions.

- **Game Mode:** we proposed the game as a turn-based one available for local multiplayer, which means one player has to wait for the others' turns to finish the round. In our tests on the prototype, we found it possible to allow simultaneous actions for all players within a round. To solve edge conditions, the bombs placed in one round take effect (block players and being triggered by other bombs) only after the next explosion calculation (i.e. the start of the next round). At the same time, local multiplayer will not be possible because the players should act at the same time and should not know the other player's action in the round.
- **Time Travel Feature:** the travel back feature has been proven to successfully deepen strategy depth and to be a powerful resource to compete for. It can be used as a redo to save the user's life when trapped, or as an aggressive item by taking advantage of the remained last bomb. However, players feel the consequence of using time travel on our paper-based prototype is unclear, which affects user experience. In our game development, we should provide previews of travel-back to the players.
- **Game Concept:** we are happy to find that all 4 members of our team feel it's fun to play with the prototype. Since we are different types of gamers, we are more confident in having the game attract a large range of players.

Milestone 3: Interim Demo

9 Task Progression

We have successfully completed all tasks within our minimum and low target objectives. We have focused mainly on the game logic and its functionality until now. For features in our desired target, we have made significant progress. The local multiplayer functionality is now fully implemented, allowing two players to share one keyboard (each using either the left or right half). Our game is now fully playable locally in a two-player mode on a single level, featuring active and passive bomb explosions, chained reaction mechanics, and time travel functionality.

Date	Milestone	Week	Layer	Task	Time		Owner	State
					Expect	Actual		
Nov 06-12	Prototype	1	Prototype	physical Prototype	4 * 5	4 * 5	All	DONE
				simple assets	5	5	All	DONE
				simple map & static object	5	3	Jialin	DONE
				player move & item place	5	1	Jialin	DONE
				simple GUI	5	2	Mahdis	DONE
Nov 13-19		2	Minimum	active & passive bomb	6	7	Miguel	DONE
				resource system	6	1	Yaxuan	DONE
				interactive map & object	6	6	All	DONE
				turn-based gameplay	6	10	Yaxuan	DONE
				version integration	4 * 4	4 * 2	All	DONE
Nov 20-26		3	Low	specific assets	8	2	Miguel	TODO
				time travel mechanism 1/2	8	6	Jialin	DONE
				weapon & interactive object	8	8	Mahdis	DONE
				full GUI	8	8	Yaxuan	DONE
				version integration	4 * 2	4 * 2	All	DONE
Nov 27-Dec 03	Interim demo	4		time travel mechanism 2/2	8	10	Jialin	DONE
				cascaded explosion refine	8	2	Miguel	TODO
				Desirable	map generation 1/3	8		
local multiplayer 1/3	8	10	Yaxuan		DONE			
Dec 04-10		5		version integration	4 * 2	4 * 2	All	TODO
				map generation 2/3	8			
				local multiplayer 2/3	8		Yaxuan	DONE
				explosion effects 1/2	8			
				more weapon and objects	8			
			Desirable	version integration	4 * 2		All	
				map generation 3/3	6			
				local multiplayer 3/3	6		Yaxuan	DONE

Figure 11: Time schedule

10 Challenges

10.1 Input System: Keyboard Splitting

Implementing local multiplayer using Unity's modern input manager presented unexpected challenges. While the system is designed to simplify input handling, configuring two players to share a single keyboard proved non-trivial. The default player-joining behavior assigns players to separate devices, which conflicted with our keyboard-splitting setup.

To resolve this issue, we:

- Switched the player join behavior from automatic to manual, and manually join the players in Game-Manager.
- Assigned the same keyboard device to both players.
- Configured distinct key schemas for each player, ensuring that the shared keyboard functionality operates as intended.

10.2 Time Travel Feature: Snapshots and Current States

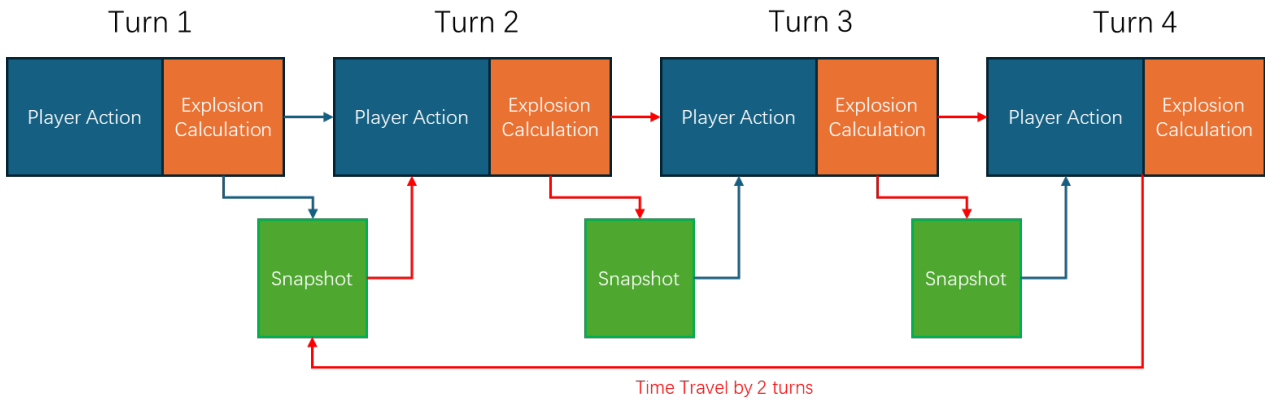


Figure 12: Overview of Time Travel Feature. The red arrows denote the pipeline of Time Travel.

The time travel feature requires taking snapshots of each round and properly rewinding to the related previous snapshots. So the main task of the time travel feature is separated into three subtasks:

- Definition of the data structure of snapshots.
- Taking the snapshots to save the current state.
- Loading snapshots to rewind to a previous state.

10.2.1 Data Structure Definition

Because the Player, Bomb, and Resource are Monobehavior classes, we need to define information classes as a record to take a copy of their current states, otherwise the information in snapshots will be changed due to reference. So we created PlayerData, BombData, and ResourceData classes to store their core data in the snapshots. The wall data is stored as a list of position, and the map items are in a list of GameObjects for convenient placing and enabling/disabling. To allow remaining the last bomb placed by each player, we recorded the last placed bomb in Player class and stored it in PlayerInfo class.

10.2.2 Taking Snapshots

At the end of every turn, a snapshot of the whole game state will be taken, by storing data to the defined data structure.

10.2.3 Loading Snapshots

When a player calls time travel, the current turn will not be taken into account and will rewind back to a previous state by loading a snapshot.

The Loading of snapshots contains player position placement, player resource management, map wall placement, map item recovery, past bomb and last bomb placement, bomb state (e.g. turns until explosion) loading, game manager state loading.

There are some edge conditions here that can bring up unexpected conditions, such as:

- Calling a 3-turn time travel in the 2nd turn.
- Time Travel back to a turn when the hourglass hasn't been picked up.

To deal with the issues, we:

- Revert to initialized state when calling an early time travel.
- Make the hourglass an exception in recovery, i.e. the hourglass is a one-time item.

10.3 Item Interaction

Until now we display the picked-up items in the game world such as hourglass, coins and the weapons in a UI panel. For the coins we use yellow sphere and for the hourglass we use a blue sphere for now. We have defined an Item class to demonstrate different existing item types that appear as in-game world items. The user picks up the items by collision detection with the items. The resource management system that we have implemented facilitates the storage of them in our inventory and is easily scalable.

We differentiate between stackable and unstack-able items. Items such as coins and various types of weapons are considered stackable. However, hourglass item is a unique and unstackable item, which the user can hold a use one time. This way there is a fair chance for all players to be able to pick up at least one hourglass per game, once it appears.

Items for now are placed manually by us, for further improvement we plan to design a respawn system of the in-world items. The respawn system has to be randomized but also calibrated with the generated map per level. Additionally, one further improvement will be creating a visual inventory to display the items as icons and reduce the amount of text displayed. This will improve the game experience for the players.

10.4 Chain Reaction

We implement the logic for placing active and passive bombs on the current player position. Active bombs have a counter that is decreased at the end of each turn. Once the counter reaches 0 the bomb explodes and affects the tiles in all 4 directions (up, left, down, right). We check for collisions with any object tracing a ray from the bomb position and retrieve the first hit object (if any). If the object lies within the bomb radius, then it will be affected in the following way: breakable walls will be broken, players will be eliminated and other bombs will explode. We add a small explosion animation at every affected tile to keep visual coherence and help the development testing.

To prevent any undesired behaviour we place every object center at round number positions and assume their position as the tiles for our game with a tile step of 1. This allows us to perform all the distance checks using the objects' positions directly and avoids the additional logic for a matrix based tile manager.

In order to maintain the logic proposed in our prototype, we want to perform all explosions at the same time, and afterwards compute the result on the breakable walls. This prevents some edge cases where the order of the bomb explosion changes the final result, e.g. propagating an explosion through a breakable wall after it has been affected by another bomb that same turn. We consider this specially important for obtaining a deterministic output from the same initial situation in a strategy game.

11 Design Revision

11.1 Scalability Considerations

During the implementation we learnt that it is important to focus on the scalability of the code to facilitate the extension of mechanisms, if needed. We have not made any design revision compared to our presented prototype up until now. Instead, we focused on not only achieving the current targets but also ensuring scalability for future versions. Below we focus on some of them:

- **Turn Mechanism:** instead of managing turns solely within a single game, we implemented a hierarchical structure that organizes gameplay into games, rounds, and turns. This approach eliminates the need for major refactoring when introducing a multiple-round setting, enabling seamless winner calculation and turn management adjustments.
- **Resource Management:** all user resources (including steps, coins, and time-travel items) are managed cohesively. This structure simplifies the addition of new items and facilitates future numeric or categorical extensions.
- **Input Handling:** by adopting Unity's event-driven input system, we ensured that our local multi-player requirements are met. This approach also lays a strong foundation for transitioning to remote multiplayer in the future.

12 Progress in Photos

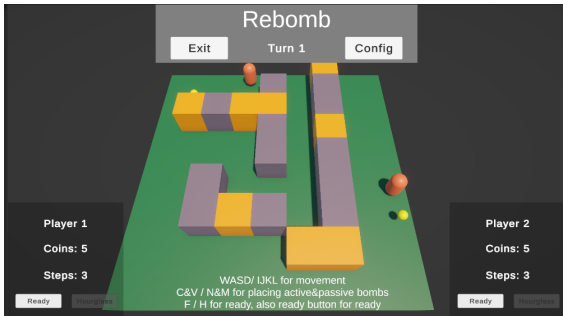


Figure 13: Initial state of the game



Figure 14: Hourglass spawns under breakable wall.

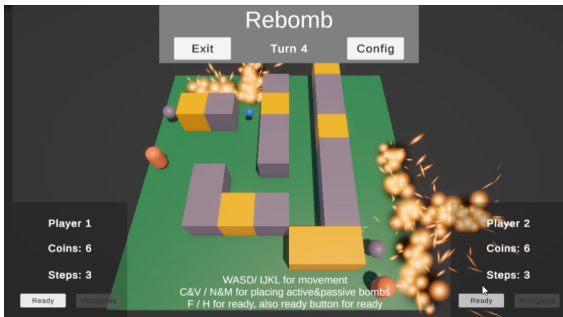


Figure 15: Active and Passive Bomb Explosion

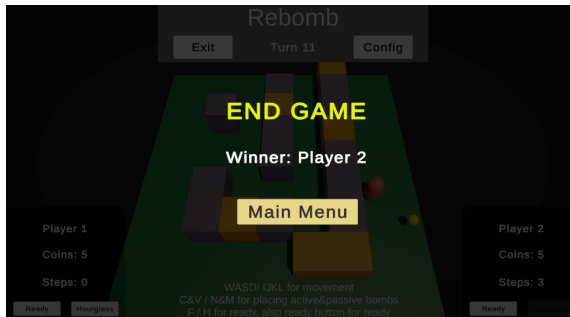
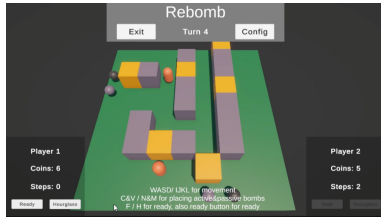
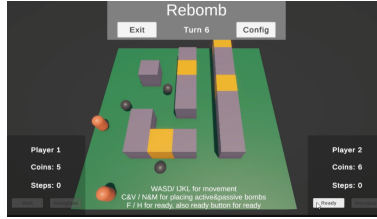


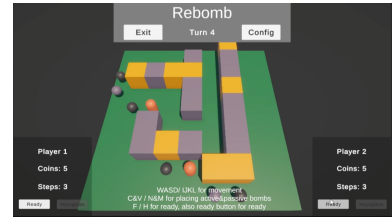
Figure 16: Endgame Result



Turn 4 Status



Before the Usage of Hourglass



After the Usage of Hourglass