Game Programming Laboratory

Team 4

Patrick Schmid - Producer, Programmer, Designer
Dirk Hüttig - Programmer, Designer, QA Manager
Sven Kellenberger - Programmer, Designer, Visual Artist
Christian Knieling - Programmer, Designer, Visual Artist

Contents

1 Formal Project Proposal

1.1 Game Description

1.1.1 Overview

The Dark Matters is a collaborative multiplayer game where players have to communicate, allocate resources and make decisions to reach as high a score as possible. The game can be played locally by two to four players, the game difficulty adjusting to the number of players.

In The Dark Matters, the players are aboard a spacecraft and try to steer it through outer space with the aim of flying as far as possible through a procedurally generated map. While the spacecraft flies with a constant speed after an initial boost, it might be slowed down due to space scrap, asteroids and so on. Also, enemies try to attack and damage the spacecraft. To accelerate the spacecraft and to repair it the players need to collect iron and an all-purpose fuel which they can get from enemies and bypassing asteroids.

Players can either perform actions on the spacecraft or board smaller spacecrafts to defeat enemies and collect resources. On board of the main spacecraft players can steer the ship, repair the shields, control weapons, process resources or perform upgrades.

The players will have to work together and agree on a common strategy with the goal of surviving as long as possible. The game ends when the spaceship has lost all its health, either from enemy attacks or from asteroid impacts.

1.1.2 Background Story

The year is 2099 A.D. and mankind has finally conquered space. A jump in space technology has allowed for an entrepreneur formerly known for building electrical cars to send the first manned mission to Mars in 2030. Since then, Mars has been successfully populated and as the effects of climate change made it more and more unpleasant to stay on earth, a lot of people have decided to emigrate to the Red Planet.

This caused the remaining population on earth to radicalise. Feeling bitter because of being left behind on the planet their ancestors helped destroy, the earth dwellers turned to a wigged, orange skinned man to lead them. All countries of the earth have since united under the banner of the United Countries of the World (UCW) and their leader is openly pondering war against Mars. For this, the United Countries Space Force (UCSF) has been created and is equipped and trained as we speak.

However, there remain still a few reasonable people on earth who can see the catastrophic consequences of a war with Mars but they are intimidated and muzzled by the new regime. Because of this, they have joined forces and built a spacecraft in silence to try to get to Mars in time and warn the Martians.

This is where we join in, as our heroes try to reach Mars while being attacked by the troops of the UCSF.

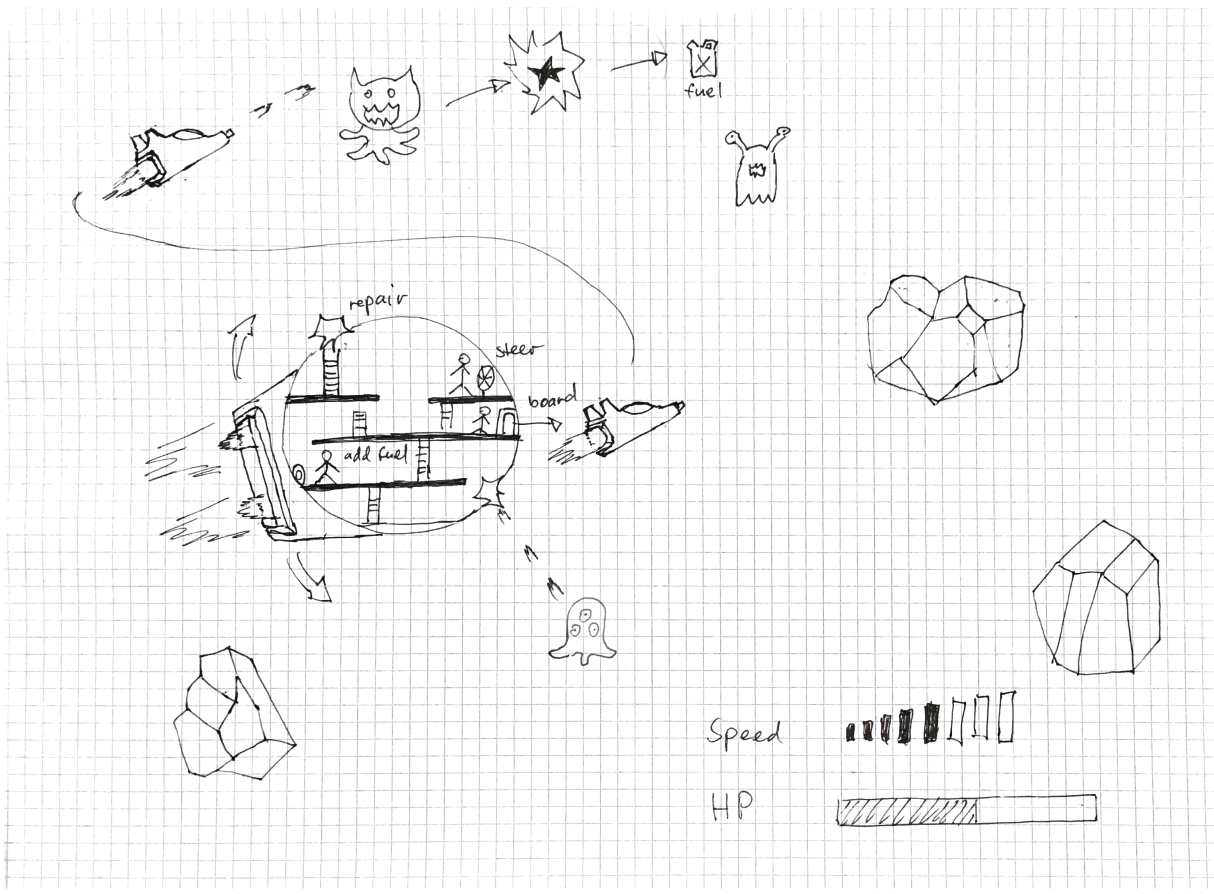
1.1.3 Design Decisions

The game will be in flat design 2D or 2.5-D voxel or low poly art. We will make the final decision after a first design phase since there are no artists in the team.

The players always need to have enough iron and fuel for different purposes. They can use smaller spacecrafts to get resources and attack enemies but the small spacecrafts also need iron to be built (after they are destroyed) and fuel to fly. Inside the main spacecraft players need iron to repair the shields and fuel to accelerate and steer the ship. Players can steer the spacecraft within a limited range to evade obstacles such as meteoroids and asteroids. Also, if players die outside of the ship, resources are needed to create a clone (i.e. to respawn the player). A research system could provide additional depth by providing an opportunity to invest resources into improved equipment.

The game is over if either the main spacecraft explodes because its health reaches zero or if it doesn't move for a certain amount of time. The former will eventually happen if no fuel is available since enemies will attack the ship.

The challenging part of the game is to attend to the tasks that are most needed in that moment. As an example, we consider that three players are attacking enemies from their smaller spacecrafts and the main ship needs to be repaired at three different locations. If all three players stay to fight the enemies, time might run out for the fourth player on the main ship to fix the damage and the game would be over. Thus, another player has to return to the main vessel to help his teammate. Furthermore, there might be obstacles which require one player to steer the ship and prevent it from further damage. The game always tries to create more tasks than players

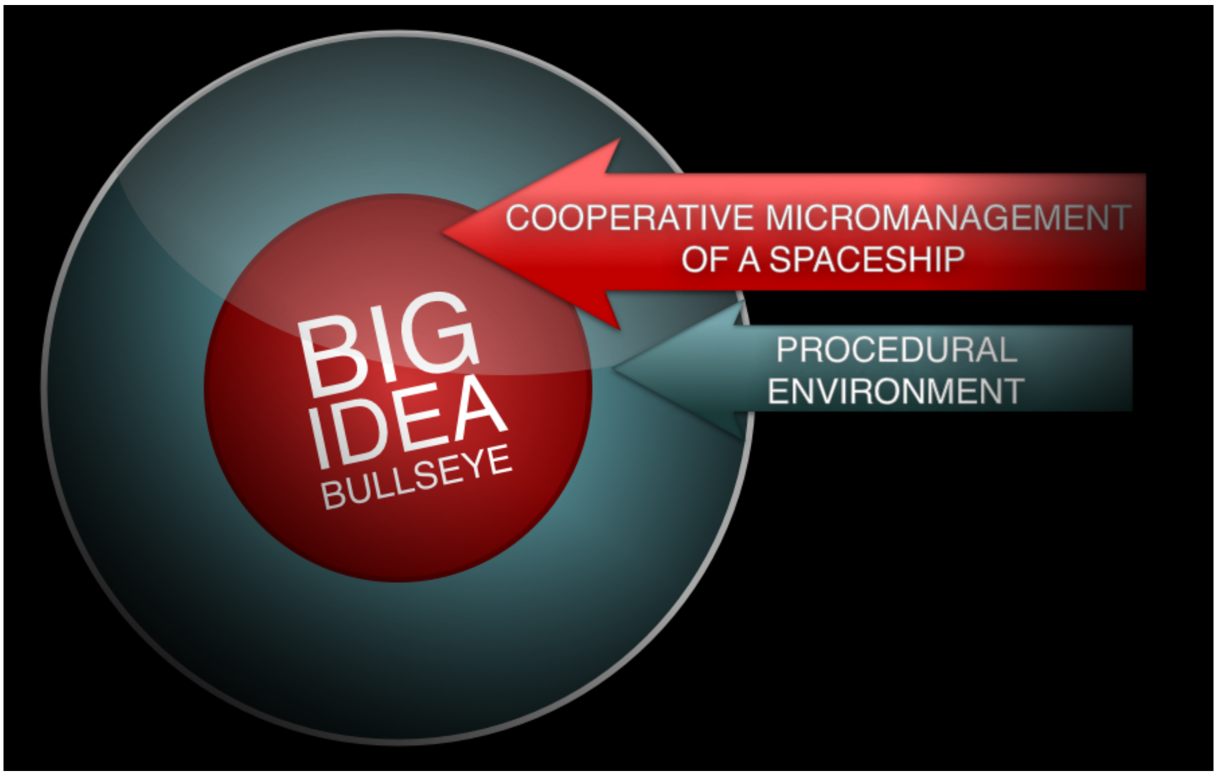


available while each scenario should still be beatable.

Controls will be kept as simple as possible to not get confused by the different tasks. Most of the time players only need to use one button for actions and the joystick for directions. Enemies will have a simple AI whose goal it is to damage the main ship. The main spacecraft will be shown from a top-down view and the interior is always visible.

1.2 "Big Idea" Bullseye

Big Idea: Cooperative micromanagement of a small spaceship in order to advance in space.
Technical Achievement: Procedural world generation and procedural enemy and resource spawning.



1.3 Technical Achievement

1.3.1 Procedural Environment

All players together control different parts of a spaceship, in order to navigate through a 2d procedurally generated space world, where enemies and resources spawn over time. One challenge will be to increase the difficulty of the map over time, while it should still remain beatable.

1.3.2 Resource Collection

Players need to collect resources in order to repair, upgrade and refill the ship. In order to do so, they can leave the mother ship in a small shuttle and collect resources from destroyed asteroids and enemies.

1.4 Development Schedule

This section is divided into two parts. First, we show all tasks layered into five categories depending on their importance of the final product. Secondly, we provide a detailed timeline of the development schedule.

1.4.1 Layered Task Breakdown

Functional Minimum

- Controllable player entities & small ships
- Main ship with attacking station(s)
- Small ships controlled by player
- Basic static map
- Enemies attacking main ship with simple behavior
- Collectable resources
- Collision detection
- Interaction between objects/player

Low Target

- Map generation
- Simple assets
- Improved enemy behavior
- Game menu

Desired Target

- Balancing of game mechanic
- Sound effects
- Soundtrack
- Improved assets and animations

High Target

- Station for upgrading ship

- Resources for upgrading ship
- Hints to help player understand gameplay
- Different types of enemies

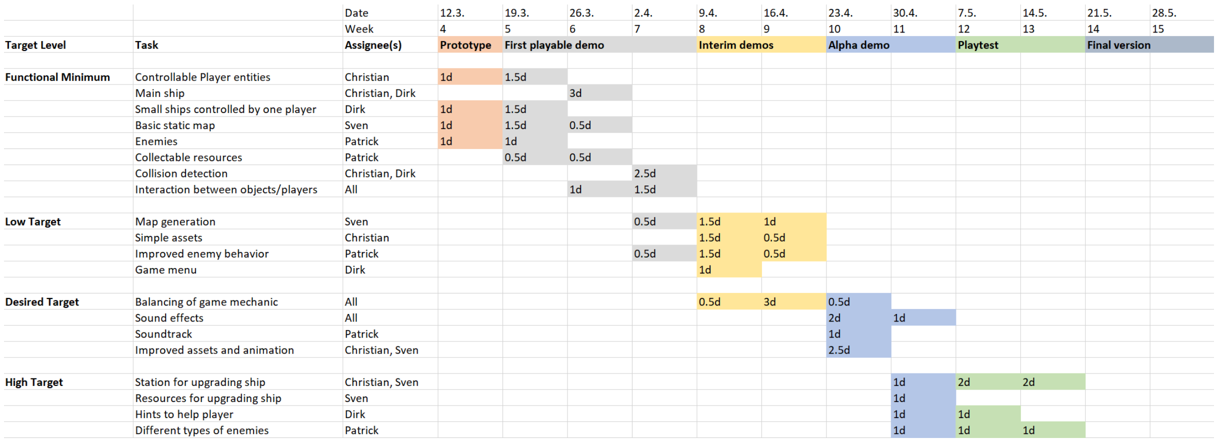
Extras

- Online mode
- Competition mode (2 vs. 2)
- Different types of ships
- Big boss fights

1.4.2 Task List

The task list with assignee(s) and estimated workload can be checked in the timeline.

1.4.3 Timeline



		Date	12.3.	19.3.	26.3.	2.4.	9.4.	16.4.	23.4.	30.4.	7.5.	14.5.	21.5.	28.5.
		Week	4	5	6	7	8	9	10	11	12	13	14	15
Target Level	**Task**	**Assignee(s)**	**Prototype**	**First playable demo**			**Interim demos**		**Alpha demo**		**Playtest**		**Final version**	
Functional Minimum	Controllable Player entities	Christian	1d	1.5d										
	Main ship	Christian, Dirk			3d									
	Small ships controlled by one player	Dirk	1d	1.5d										
	Basic static map	Sven	1d	1.5d	0.5d									
	Enemies	Patrick	1d	1d										
	Collectable resources	Patrick		0.5d	0.5d									
	Collision detection	Christian, Dirk				2.5d								
	Interaction between objects/players	All			1d	1.5d								
Low Target	Map generation	Sven				0.5d	1.5d	1d						
	Simple assets	Christian					1.5d	0.5d						
	Improved enemy behavior	Patrick				0.5d	1.5d	0.5d						
	Game menu	Dirk					1d							
Desired Target	Balancing of game mechanic	All					0.5d	3d	0.5d					
	Sound effects	All							2d	1d				
	Soundtrack	Patrick							1d					
	Improved assets and animation	Christian, Sven							2.5d					
High Target	Station for upgrading ship	Christian, Sven								1d	2d	2d		
	Resources for upgrading ship	Sven								1d				
	Hints to help player	Dirk								1d	1d			
	Different types of enemies	Patrick								1d	1d	1d		

1.5 Assessment

The main strength of our game is that all players always have something to do and are not constrained to one thing. People interested in arcade like space games and friends of local multilayer games are our target audience. The core gameplay features are:

- Swapping and prioritizing different tasks in and outside of the spaceship.
- Manoeuvring and upgrading the spaceship in order to advance in space.
- Defending the ship from incoming enemies.
- Managing available resources.

Overall, the game should in principle always be beatable but still pose a reasonable challenge for the players. The game should also motivate team coordination, be fun to play and easy to pick up.

2

Prototype

2.1 Prototype Setup

The game board of our prototype consists of a 10x16 tile grid map, which simulates a screen, and a counter bar to simulate the progress of the main spacecraft on its way from Earth to Mars. Next to the map, there are indicators for resources, shield and current velocity of the main spacecraft.

We use a mix of wooden pieces and pieces made out of cardboard to represent the characters, the main ship, the smaller spacecrafts, the enemies, the asteroids and the resources. During the game, the players can place their character in a room inside the main spacecraft or in one of the smaller spaceships.

The game is played by three players trying to get to Mars against one player who simulates the movement of the computer. The goal of the three players is to move the piece on the counter bar to field 25. The players lose either if their main spaceship's shield indicator reaches zero or if the computer's piece surpasses the players' piece on the counter bar. Each round the counter of the players is increased by their current speed while the counter of the computer is always increased by three.

Initially the spaceship has five resources, full shield bar (ten) and the speed is two. Before the game begins, there are also two meteoroids spawned on the right side of the board and three enemies are spawned on the left side. The exact spawn position is determined randomly by a W20 die.

The procedure of each turn is always the same. First the players take turns in a predetermined order. After each player has finished, all meteoroids on the board move one tile to left, i.e. each meteoroid moves three tiles in each round. Next, the computer can perform actions for all enemy spaceships. If the players were on or passed squares 5, 10, 15, or 20 on the counter in

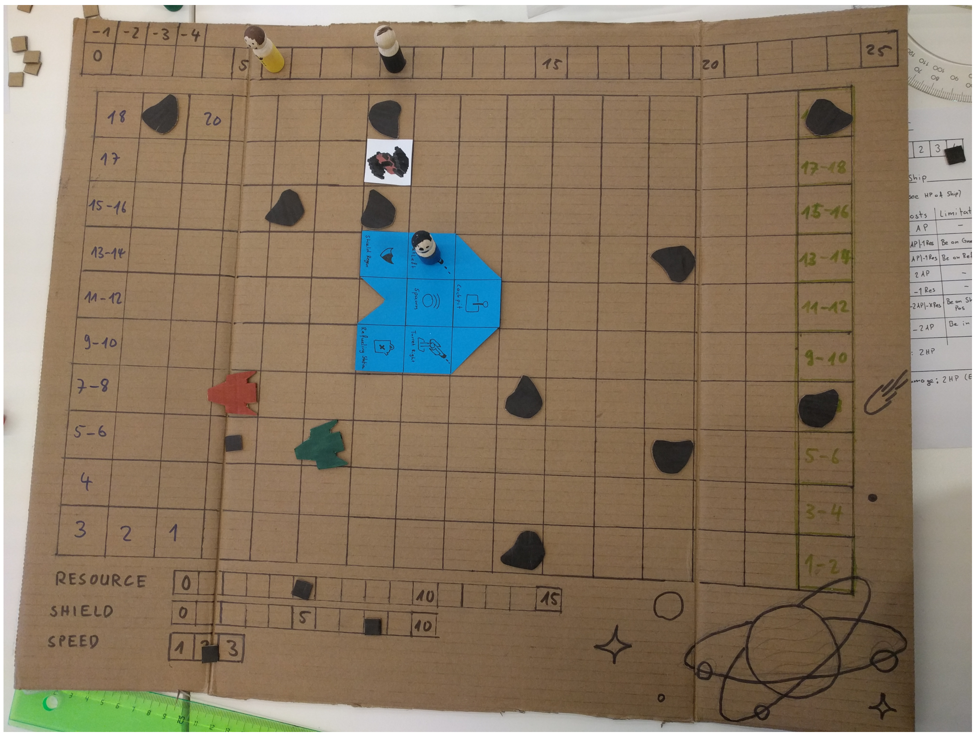
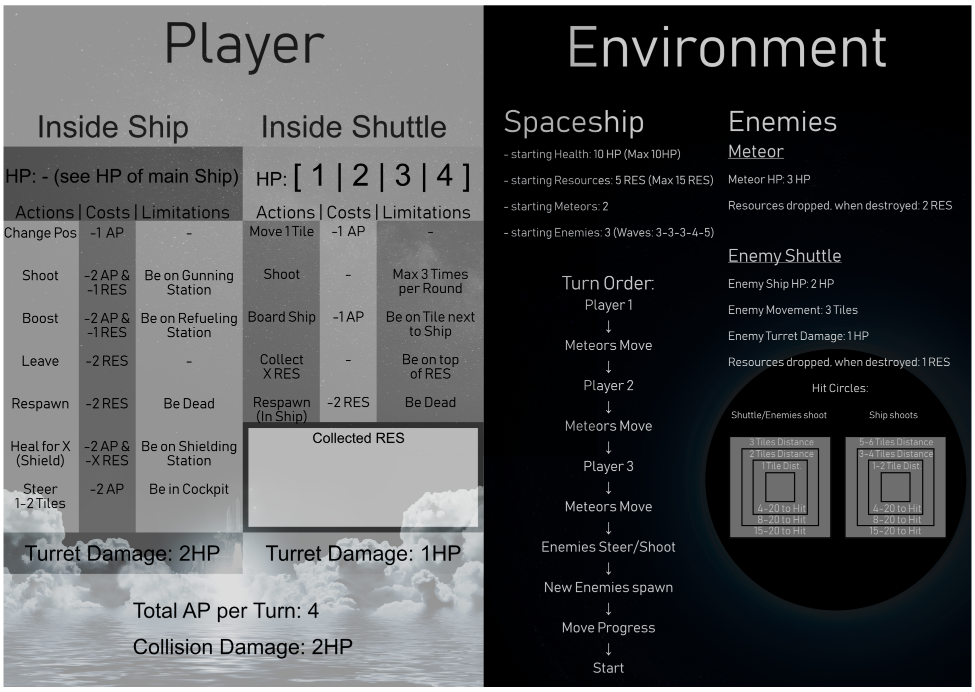


Figure 2.1: *Overview of the board. The counter bar is on top, while the indicators are displayed at the bottom. Meteoroids (black pieces) spawn on the right, the exact tile is determined with a W20 die. Similarly, the enemies (printed out spaceships) spawn on the left side of the board. The players can perform actions inside the main spaceship (blue) or can board smaller spacecrafts and can go collect resources, fight off enemies or destroy meteoroids.*

Figure 2.2: *Playing the Prototype usually involved a lot of discussion between the players on the best strategy. This is exactly the collaborative playstyle we hope to capture in our video game.*



Player

Inside Ship

HP: - (see HP of main Ship)

Actions	Costs	Limitations
Change Pos	-1 AP	-
Shoot	-2 AP & -1 RES	Be on Gunning Station
Boost	-2 AP & -1 RES	Be on Refueling Station
Leave	-2 RES	-
Respawn	-2 RES	Be Dead
Heal for X (Shield)	-2 AP & -X RES	Be on Shielding Station
Steer 1-2 Tiles	-2 AP	Be in Cockpit

Turret Damage: 2HP

Inside Shuttle

HP: [1 | 2 | 3 | 4]

Actions	Costs	Limitations
Move 1 Tile	-1 AP	-
Shoot	-	Max 3 Times per Round
Board Ship	-1 AP	Be on Tile next to Ship
Collect X RES	-	Be on top of RES
Respawn (In Ship)	-2 RES	Be Dead

Collected RES

Turret Damage: 1HP

Total AP per Turn: 4

Collision Damage: 2HP

Environment

Spaceship

- starting Health: 10 HP (Max 10HP)
- starting Resources: 5 RES (Max 15 RES)
- starting Meteors: 2
- starting Enemies: 3 (Waves: 3-3-3-4-5)

Turn Order:
Player 1
↓
Meteors Move
↓
Player 2
↓
Meteors Move
↓
Player 3
↓
Meteors Move
↓
Enemies Steer/Shoot
↓
New Enemies spawn
↓
Move Progress
↓
Start

Enemies

Meteor

Meteor HP: 3 HP

Resources dropped, when destroyed: 2 RES

Enemy Shuttle

Enemy Ship HP: 2 HP

Enemy Movement: 3 Tiles

Enemy Turret Damage: 1 HP

Resources dropped, when destroyed: 1 RES

Hit Circles:

Figure 2.3: *Player card with all the important informations.*

the previous round, the computer also spawns new enemies on the left side of the board. Three enemies are spawned for square 5, another three for square 10, four for square 15 and finally five for square 20. After the computer has finished with the actions of the enemies, the counters are increased and two additional meteoroids are spawned.

Every player can use 4 action points (AP) each turn. He can use AP to move inside the ship, shoot from the main ship, repair the shield, steer the space ship, board a smaller spacecraft and/or move a smaller spacecraft. Detailed information on the costs and additional information can be found on the player card (see figure 2.3) . Enemies behave the same way as a player that is inside his smaller spacecraft except that they only have two HP and three AP and can only shoot two shots per round.

The spaceships take damage if they are hit by a meteoroid or by a bullet of an enemy. When a small spaceship is destroyed the player can respawn inside the main ship, however this costs resources. Destroyed meteoroids and enemies drop resources (two and one respectively) which can be collected by the main spacecraft and the smaller ships. In the latter case, the player carries the resources with him and has to get back to the main spaceship in order to use the resources.

2.2 Playing Experience

We were able to find a nice balance in our prototype and therefore it was really fun to play. What we especially liked about our game is that it encouraged intensive discussions between the players about the strategy. The players really need to work together to make it to the end of the game. This is exactly the gameplay experience that we want to achieve in our video game.

Another thing we realised was that the overall perceived challenge of the game could quickly change. One moment we thought that we were easily cruising to victory and just one round later we felt like we were in over our head. This is not necessarily a bad thing because it keeps the players engaged and the game entertaining.

One game typically took between 45 minutes and one hour. It depends on how far the players get and how much discussion is needed to agree on a common strategy. Although this is quite some time and we hope that our video game will not take as long to play, we feel like this time is needed to experience the full strategic depth of a round-based board game.

2.3 Findings and Conclusion

The first thing we learned was that even though the overall game can be quite chaotic, there still needs to be some predictability and order in the individual parts of the game. For example, we first considered a random spawning of meteoroids and enemies from every direction. This made the game very unpredictable and chaotic and it was very hard for players to plan ahead. We will keep this learning in mind when we design our video game.

Furthermore, we were reminded that our game does not need a lot of fancy features to work but that it is fun with just a small set of performable actions. This showed us that we do not need to implement too many features early on to get a playable game and that we should carefully consider the benefit of each new feature for the overall game experience. An additional feature should not add too much complexity to the game (at least for the player and the controls) but it should add a lot of new fun possibilities.

Overall, we did not really change our original game idea much. However, we were able to roughly learn what could work and what will not and also where our priorities during the development should lie.

3

Interim Report

3.1 Progress

During development we worked on targets of different layers concurrently. At the time of writing we finished all of the targets of the functional minimum layer and almost all of the low and desired targets.

3.1.1 Engine

Before starting with any game logic we focused on constructing a robust engine. The game is separated into Scenes managed by the SceneManager, i.e. there will be one Scene for the actual game and at least one Scene for the menu. We have also created different testing scenes which allow us to draw up certain scenarios and test different mechanics pretty quickly.

An important separation we implemented early on is the one between game logic and game representation. All game entities have a seperate class which manages all its logic but all things concerning the representation of the entity in the game are outsourced into so-called Presenters. Entities can have multiple Presenters which encapsulate all the logic to display various game elements (e.g. health bars, debug shapes). Game entities can also be nested into "position trees". This means that an entity which is a subordinate to another entity only moves relative to that entity not to the world. An example is the movement of the crew members inside the spaceship as they only move relative to the spaceship while the spaceship itself moves relative to the world. All entities are managed centrally by the GameEntityManager.

We also implemented simple collision detection and resolution for all entities. Collisions are detected by the GameEntityManager and for each collision a collision vector (the direction in which to bounce away) is calculated and passed to the respective game entities. Based on the collision vector, the mass of the entity and some other properties each entity then resolves the collision.

3.1.2 Gameplay

Players start as crew members in the main ship. They can move around and interact with different stations by pressing the A button. The control station allows players to move the main ship up and down to evade obstacles. The cannons on either side of the ship can be used to shoot enemies and meteoroids. The acceleration station increases the overall speed of the main ship and the shield station repairs the shield. At the moment the two latter stations use up resources (dark matter) of the main ship. Players can exit the main ship via the exit stations and will automatically board a shuttle which can move in all directions to shoot enemies and meteoroids. Destroyed targets leave resources which can be collected by the shuttles. When players reenter the main ship via one of the exit stations the collected resources are added to the main ships resources. When a shuttle is destroyed the player respawns at a specific position in the main ship (as a crew member).

At the moment enemies follow the main ship and attack it from a fixed distance. Meteoroids are randomly spawned and fly from right to left.

3.1.3 User Interface

In order to render menu elements, we decided to create a view system similar to something used in android app development. This means we can align text, images inside a container and relative to other views in our game. This functionality greatly reduces the overhead in UI creation and enables us to make the UI somewhat resistant to changes in aspect ratio and screen size on different monitors.

3.1.4 Assets

For our game we chose voxel-style 3D graphics and already created most of the graphical assets. The 3D models were created with Magicavoxel and Blender. The game also features a basic sound tune for the main game scene and sound effects for shots and damage. A simple animation system is in place.

3.2 Challenges

One of the main challenges was the collision detection and resolution between game entities. All collisions are handled centrally but game entities have to decide how to interact with other entities. Also, entities within the main ship should only interact with and the same for the entities outside of the spaceship. However there are some entities which should do both (spaceship, exit station).

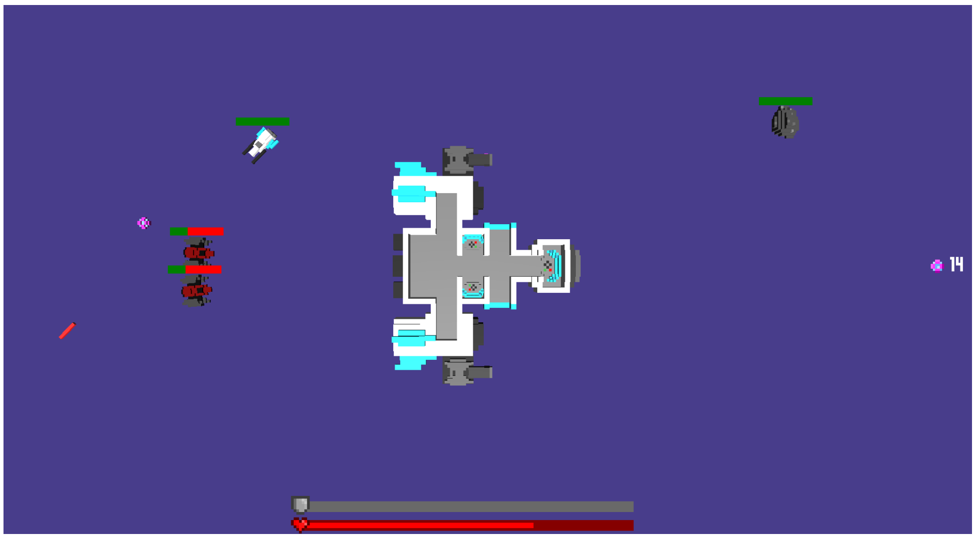


Figure 3.1: *The current state of our game. Meteoroids spawn on the right side of the screen and enemies on the left. Players can mount staitons inside of the spaceship or board smaller shuttles to fight enemies or destroy meteoroids. At the bottom, the shield and health of the spaceship are displayed.*

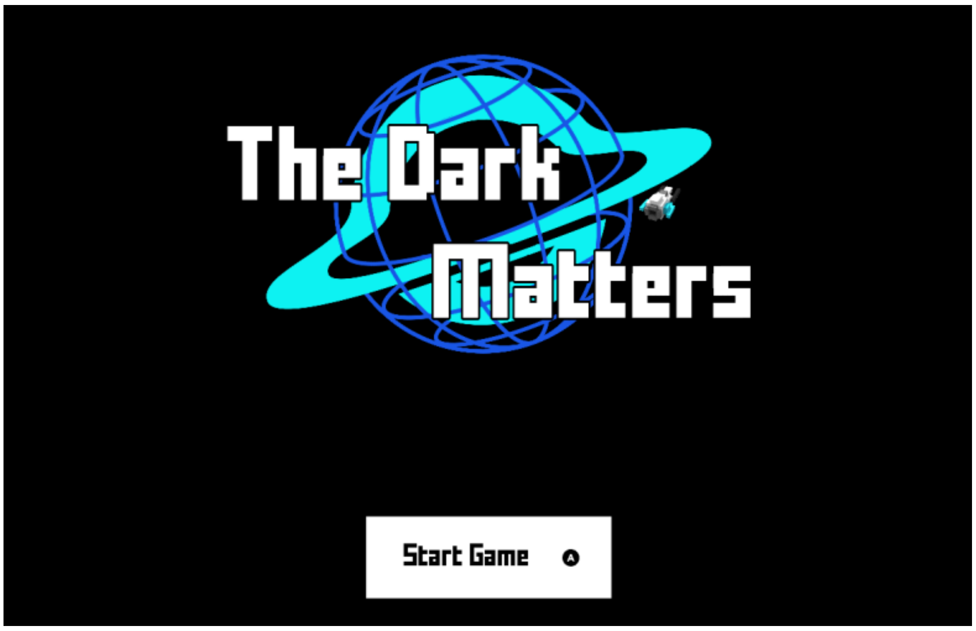


Figure 3.2: *The current start screen of our game. From here the players should be able to join a game, change game settings, view high scores etc.*

Another challenge is the balancing of the game at this point in time. The enemy behaviour is not completed yet and the difficulty of the game is hard to assess since most of the time there is only one player whereas the game is made for three or four players.

We also struggled to implement our own shadow shader, because most of the sources refer to old XNA code and link their resources to dead websites. Debugging shaders is especially difficult, when there is no print or break functionality. Although we managed to get a simple shadow shader working, we did not add it in time for the second demo. There was just not enough time and we had other priorities for the release.

3.3 Future Work

The map generation, i.e. spawning of meteoroids, will have to be improved by for example grouping meteoroids and possibly introducing other obstacles or backgrounds to make the map more interesting.

One important part that is missing at the moment are UI elements to guide the user. When a player approaches stations there should be buttons with descriptions popping up. Furthermore, entities will have to be named and/or coloured differently to differentiate between players.

We will also have to add more Scenes to the game such as the start screen, a menu screen, a highscore screen and possibly a settings screen. There should also be a way to save the current state of the game. What is also missing is a game goal which should either be to get as far as possible or to reach a certain distance. As stated in the challenges game balancing is another important aspect which we have to address.

4

Alpha Release

4.1 Progress

The last three weeks have been very productive. Overall, we have implemented all of our desired targets and also almost all of our high targets. We are currently mainly finetuning controls, UI, assets and balancing our game but we have also started working on some new exciting features such as player statistics and different game modes and difficulty settings. In the following we will present some of the new features we implemented in the last three weeks.

First of all, we added a game over screen and a pause menu to allow players to quit or restart the game. We also save the highscore of the game so players can try to improve their scores over time.

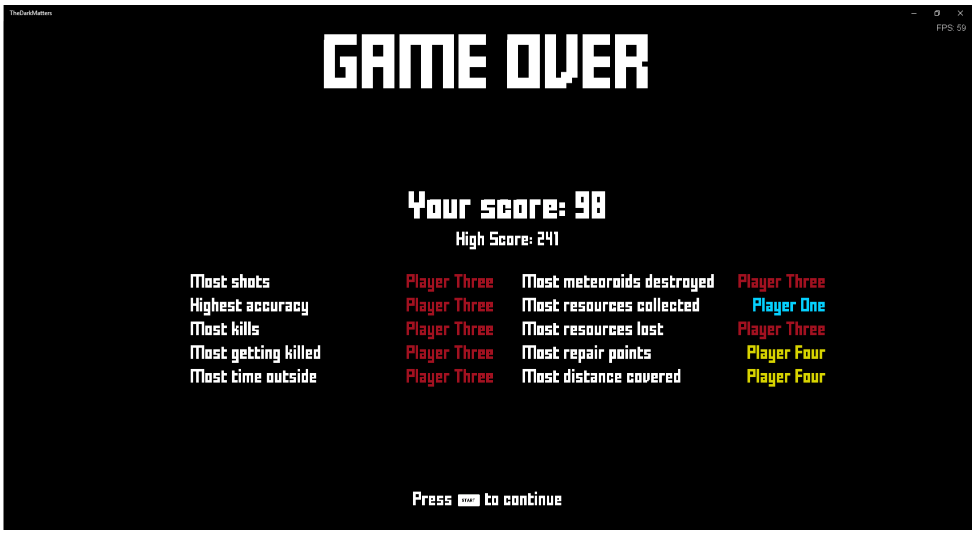
We also implemented a system for difficulty management to increase the difficulty of our game over time. Obviously, there are many decisions and factors that affect the difficulty of the game but we implemented a robust, scalable way to make changes to allow quick changes to the overall difficulty. This is necessary to adjust the difficulty based on the number of players.

Another bigger improvement we worked on was the UI. At the interim report we had some basic UI elements in place and we had already implemented a system to allow for quicker development of new UI elements. We built on top of this and now our game UI is complete with indicators for score, health, speed and resources and we also display tooltips near starship stations to tell the players which buttons to use for the corresponding actions. We also made some smaller changes to the UI of the game menu and we plan to extend the options here to allow the player to choose a difficulty setting and maybe some other options such as the the game mode as well.

To help the players understand the game we added a short interactive tutorial. Players are presented with textboxes to click through and the tutorial proceeds when certain actions are completed.

We further improved the overall looks of our game. First of all, we added shadows to our game. This sounds like a small improvement but it actually makes the game feel more realistic and adds depth to the shapes. The four different players now have characters and shuttles in four different colors to easily distinguish between them. We also reworked the background of our game which is no longer blue but rather black with the occasional star or satellite flying by. Another thing we added were particle effects. Those allow us to easily develop and display animations, e.g. for the boost of the ships or the explosion/destruction of the game objects.

We also had our first playtesting meeting during the easter break. We met to playtest our game with all four developers in one room and to discuss future work. It was a very fun afternoon and we gained many insights into what was still needed and which parts of the game had to be improved. We were also able to stress test our game regarding performance and identify room for improvement. We even fixed some major bugs during the session.



GAME OVER

Your score: 98

High Score: 241

Most shots	Player Three	Most meteoroids destroyed	Player Three
Highest accuracy	Player Three	Most resources collected	Player One
Most kills	Player Three	Most resources lost	Player Three
Most getting killed	Player Three	Most repair points	Player Four
Most time outside	Player Three	Most distance covered	Player Four

Press START to continue

Figure 4.1: *The new game over screen. We show the highscore of the whole team and the individual strengths of all players.*

4.2 Challenges

One major problem that kept us occupied throughout these last three weeks was game balancing. And we quickly realised: 'This is f*cking hard'. Our game has so many parameters that could be tweaked. This makes it very hard to decide which parameters should be changed in what fashion. For instance, if we decide that there are too few meteoroids spawned, should we change the average distance between meteoroid swarms? This could make the game unplayable if the players decide to accelerate the speed of the starship. The same goes if we just increase the overall game speed. If we increase the size of meteoroid swarm, thus spawning more meteoroids in every swarm, we make it harder to navigate around the meteoroids. Also, more meteoroids mean that more resources can be collected which meant that we suddenly had enough resources to keep the game going on forever. To tackle this problem should we just adjust the cost for

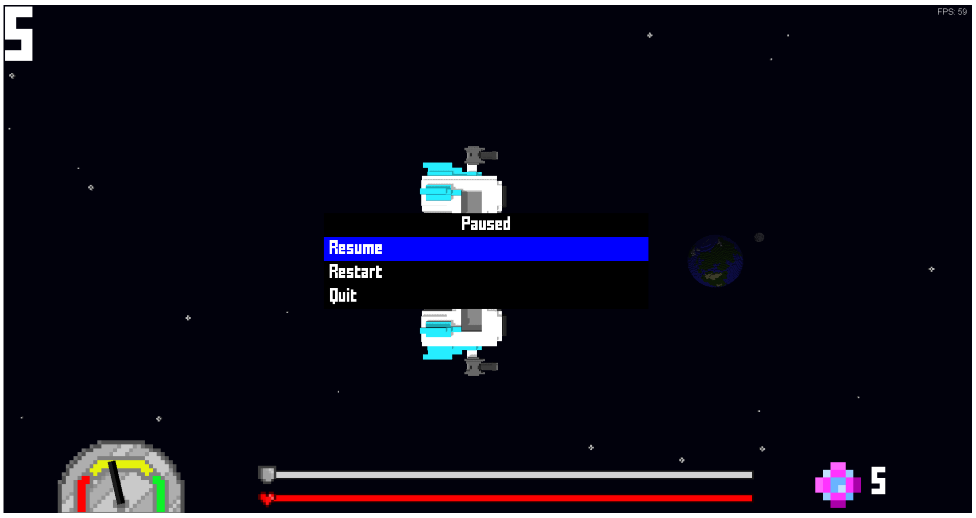


Figure 4.2: *The pause menu, added in order to be able to take a quick break and restart/ leave the game. Later on we will also add a settings sub-menu here as well as in the start screen.*

healing, upgrading and accelerating? To make the point clear, this was just one example which made it sometimes difficult to change even just small parts in our game. In the end, we managed to get a good balancing with a little trial and error though.

Another aspect we realised in our playtesting session was that our collision detection mechansim was too inefficient for the current size of the game. We had a massive drop in FPS once the screen got clustered with game entities. A fix wasn't that difficult to implement but it's something to keep in mind if we decide to add even more entities.

Finally, we also had some trouble with entities that left the screen but weren't correctly destroyed. That lead to some memory leakage and also to performance issues. Once we identified the culprits, we were able to solve this problem rather quickly, but once again this is something to keep an eye on.

4.3 Future Work

We haven't fully decided yet on what to work on in the last remaining weeks until the public presentation. One big thing will certainly be the playtesting with uninvolved people and the findings we get from there.

If time allows we will certainly start working on some new features that are in our high targets or weren't listed at all. For example we already have a simple boss fight in place that just needs some testing and refinement. Or we could invest time in a 'story mode' so the player has some variety in game play. Also, it would be interesting to add more types of enemies with different attacks.

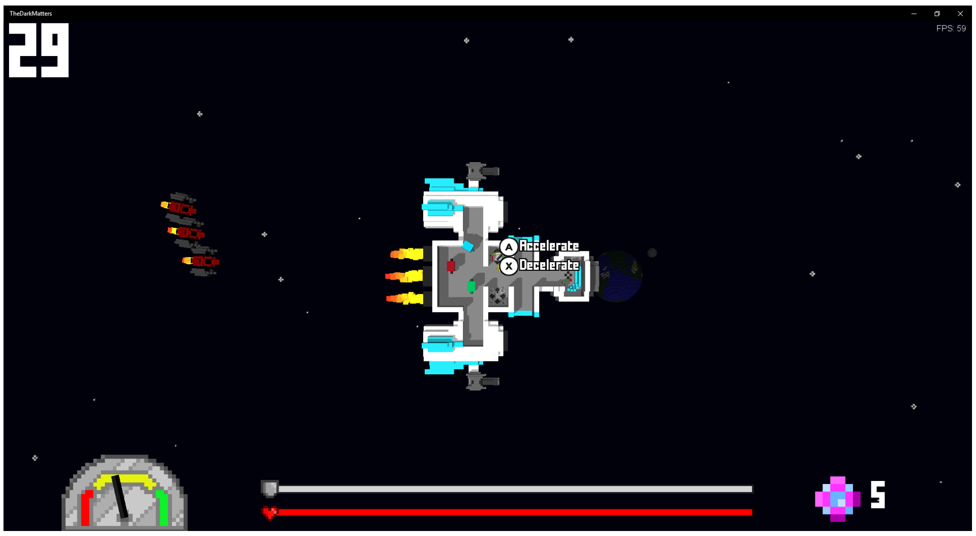


Figure 4.3: *The updated UI and background of our game. We added a speed indicator on the bottom left, pushed the resource counter to the bottom right, added tooltips to the stations and changed the look of the refueling/repair stations. Additionally we added a more pleasing background with stars, planets and satellites flying by.*

5

Playtest

5.1 Playtesting Session

For our playtesting session, we set up our game in the foyer of the CAB building in order to maximize the chance of random passerbys and to promote it to a broader audience. It was the perfect choice for us as the big screen draw a lot of attention to our project.

In advance of the session, we invited four people that were not involved with the Game Programming Laboratoy. In addition, we found six spontaneous participants in the CAB that were interested in playing our game. These people were split up into three groups, one of them consisting of four and the other two of three players. Each group was playing around 30 minutes before participating in a survey. In this time, we did not interact with the participants and just let them explore the game on their own.

While the participants were testing our game, at least one of our team members was watching closely how the session was advancing. This was done to catch details like for example group dynamics and thoughts of the players during gaming. Overall, most of the discussions between the players were about job assignments and different game strategies. Sometimes they asked how specific things could be achieved or why certain things happened. That was because they did not pay much attention to the tutorial, but more about that later. One interesting example was when we saw that a group of three people simply ignored the control station, which is used to steer the big starship. After asking if they did not know about that feature, they responded that they were well aware of it but chose to ignore it in favour of a different strategy. One of them protected the starship from the outside in a shuttle and the other two occupied the cannon stations in order to maximize their fire power.

At the end of each test run, we distributed a link to a survey that the participants could fill in. To motivate the participants to evaluate our game, we provided them with snacks and drinks.

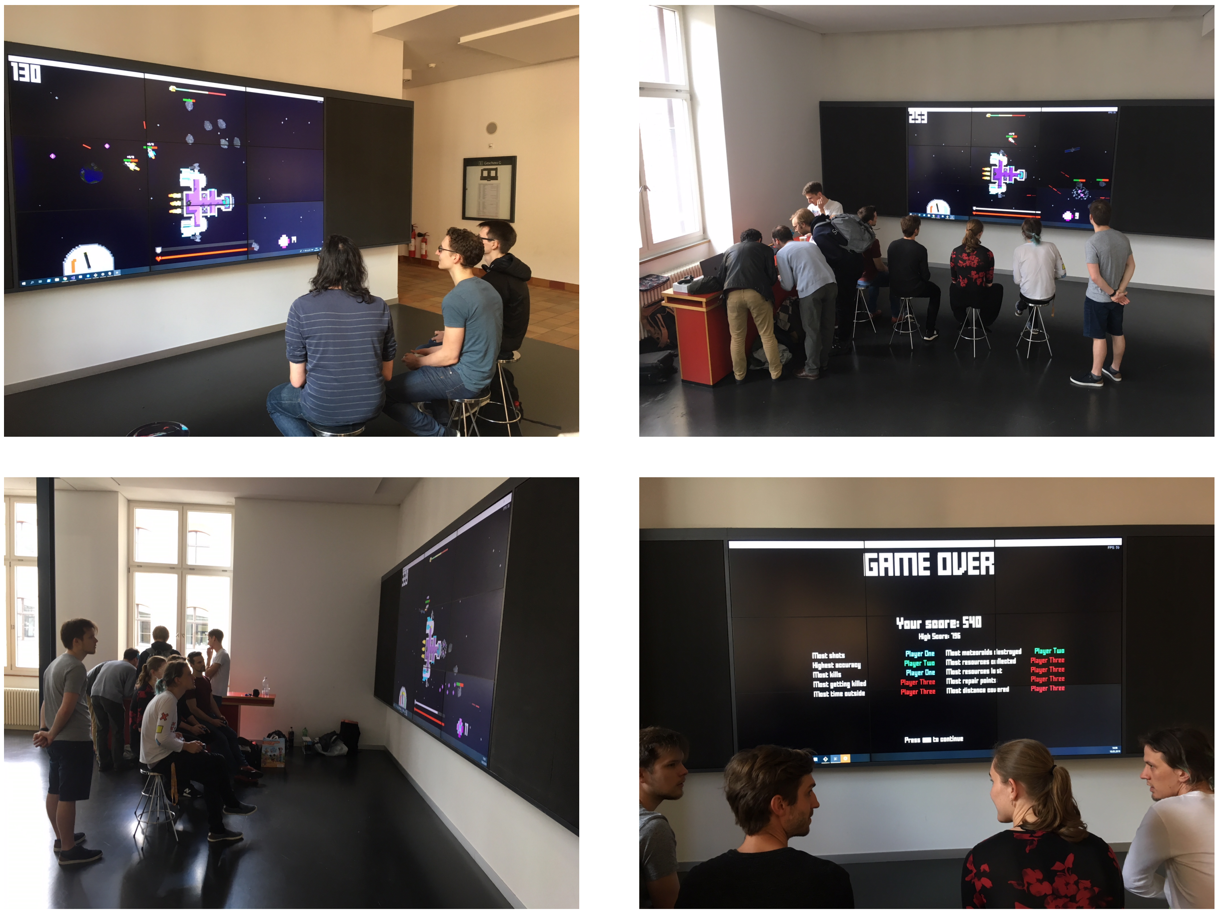


Figure 5.1: *Some impressions from the playtesting session.*

5.2 Questions and Comments

In total, we asked the participants to answer 18 questions and provide any further comment if they have any. The questionnaire consisted of four parts:

- Overall experience
- The different game aspects
- UI
- Enemy fights

We now discuss each part in more detail.

5.2.1 Overall Experience

This section of the survey consisted of seven questions. The first one was what the participants thought of the game. In Figure 5.2, the results are shown. As one can see, the feedback is very positive to this question.

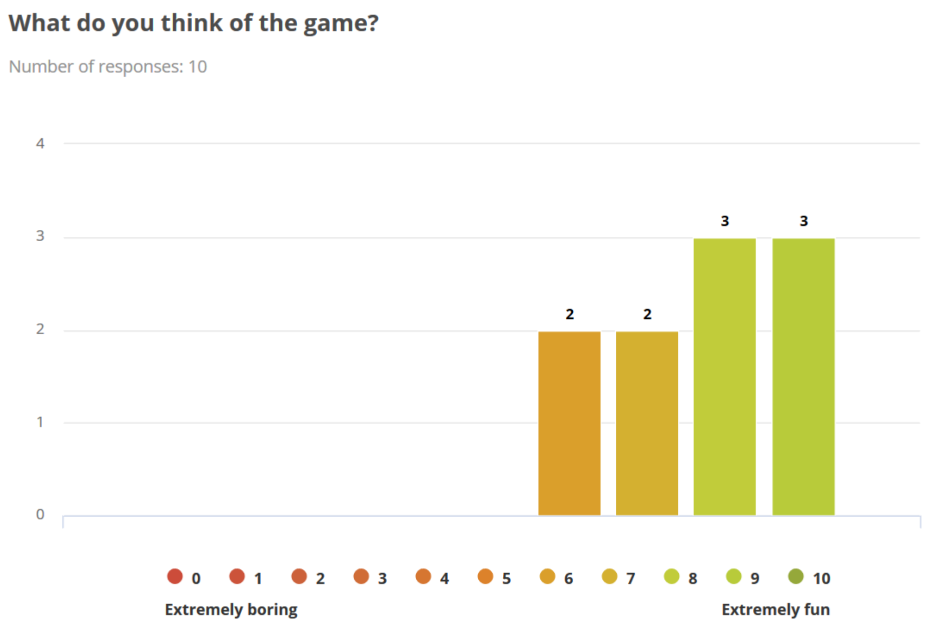


Figure 5.2: *The results of what the participants thought of our game.*

The next thing we wanted to know was how the players evaluated the controls. The result was rather positive with a score of 6.7 out of 10. We already anticipated such a result as the movement of the shuttle is indeed not very intuitive at first but can be picked up quite rapidly.

Thereafter, we wanted to know more about how the players learned the gameplay. We captured this aspect by asking how much the tutorial was helping and how hard it was to understand the different possibilites the game offers. The results are displayed in Figure 5.3. They clearly show

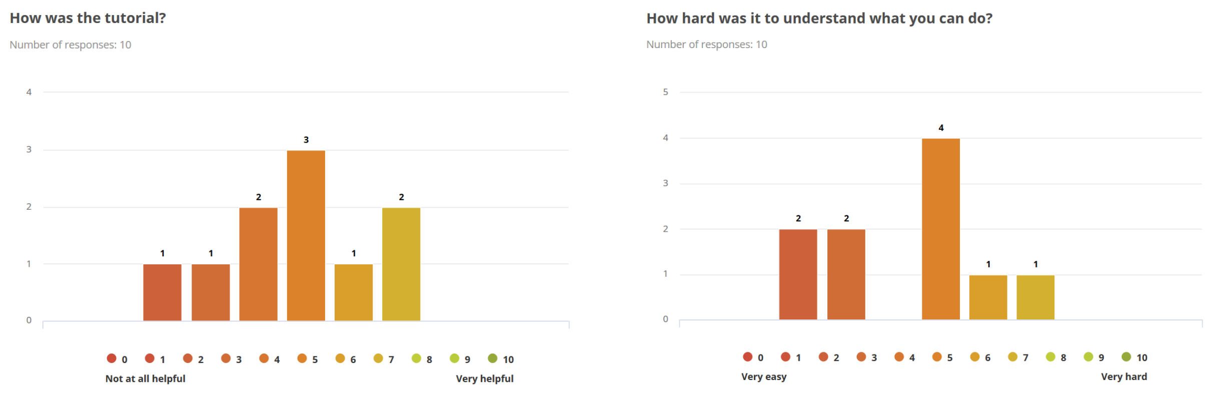


Figure 5.3: *Survey results of the learning parts.*

that we still need to improve the tutorial. We gathered this feedback also when talking to the participants after the session. Most of them found that there was too much to read and too less to actually try out.

We then asked three questions regarding graphics and sounds. The general feedback was that the graphics were already very appealing and that the music and sound effects are nice.

5.2.2 The Different Game Aspects

In this part of the survey, we wanted to evaluate what the players think of the different aspects of the game. We began by asking what game mechanic they found most useful. Then we inquired about their favorite game aspect and which one they would leave out.

The top three game mechanics are collecting resources followed by the shooting mechanisms (shuttle and cannon station). When we compare that to the answers for the favorite aspect, we see that more than half responded that either shooting or collecting resources were their favorite. Another two aspects that participants liked were the cooperation needed between the players and also the Orange Guy reference.

The results of what the participants would take out are very diverse. Some did not find anything while others complained about having too less resources and high costs for bombs. Another complaint was that the tutorial had too many explanations which we already discussed in Section 5.2.1.

5.2.3 UI

The purpose of this section in the questionnaire was to evaluate how informative the UI was and whether it was distracting or not. Figure 5.4 shows the first three questions and their results. Most of the participants liked the UI and found it informative. Furthermore, the UI was not distracting the players from the gameplay which is very important.

We also collected some individual responses that there is no indication on how much the costs

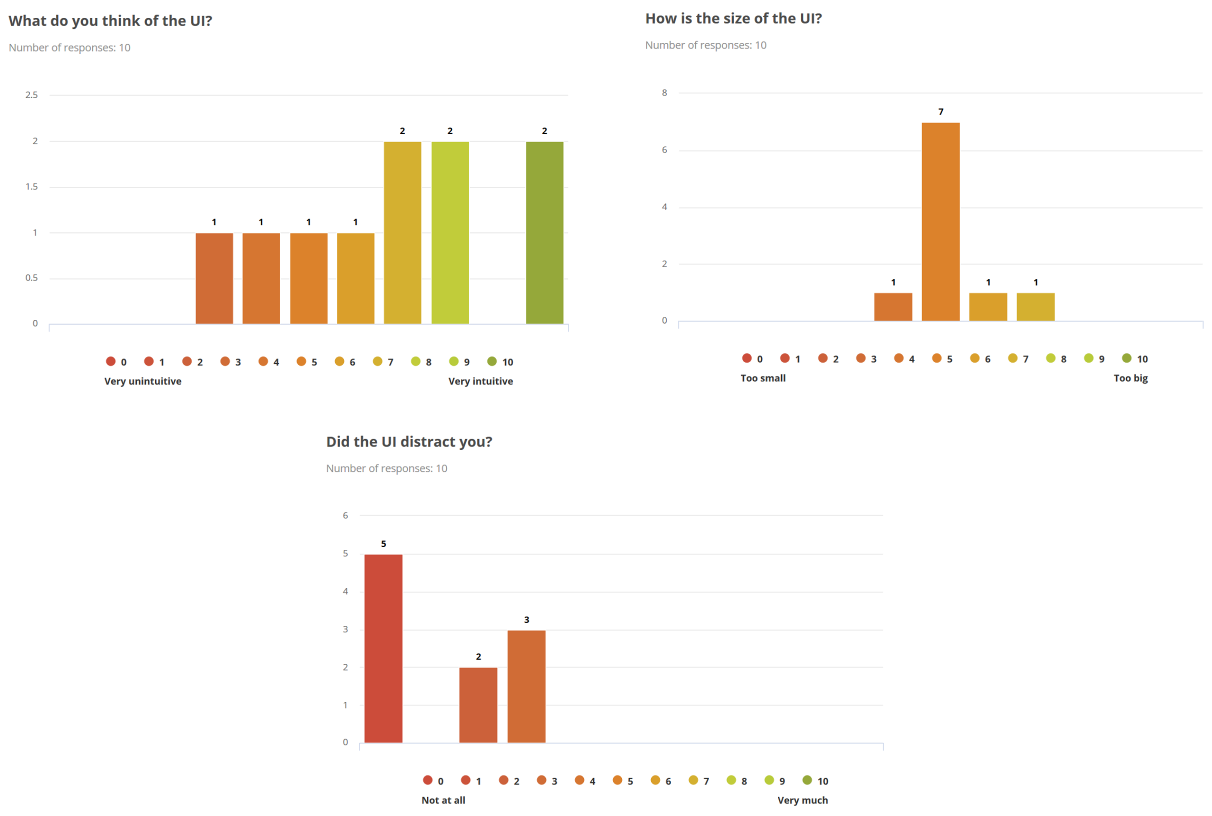


Figure 5.4: *Survey results of the UI.*

are. Especially confusing was this for deploying a bomb but also for accelerating the starship and upgrading a cannon station.

5.2.4 Enemy Fights

The last section helped us to understand what people think about the enemies and fighting them. First, we asked whether the enemies were too easy or difficult to fight followed by questions about the boss fighting mechanism. The results are that the enemies were a bit too hard to fight against. This is actually what we wanted to achieve as we do not adapt the enemy behavior when setting a different difficulty setting but by controlling the number of enemies arriving per wave. Thus, enemies should be intelligent enough to make the player's lives harder independently of how many there are.

The evaluation of the boss fight was that it is not annoying but also not extremely fun. Furthermore, people found the battle not too difficult but also not too easy. This is not what we actually hoped for except that players do not get frustrated because of an over-powered enemy. It seems that the boss fight needs to have more meaning in the game play in order to increase the fun in defeating it. Currently, players were even confused about why the boss started fighting. One of the feedback was to make a level system and to have a boss fight at each end of a level. In a future release, this might be a great option to increase the game feeling.

5.3 Design Revisions

5.3.1 Design Revisions Before Playtesting

Between the alpha release and the playtesting session we already implemented some changes such as the boss enemy encounter. We added a bar which shows the distance of the boss enemy to the starship. The boss will get closer to the starship if the speed is low. When the boss enemy reaches the starship the players have to defeat the boss to proceed.

We implemented a new bomb station which allows the players to drop a bomb which defeats multiple enemies at once when they are gathering behind the starship.

To give the players an incentive to leave the starship more often, we added spots on the hull of the starship that start burning once the health is reduced to some point. The player will then have to repair the spots from the outside.

We also added a settings screen with difficulty settings to make the game also playable for one or two players.

5.3.2 Design Revisions After Playtesting

One of the design revisions concerns the tutorial since many people at the playtesting session criticized that the tutorial was not enough interactive. In the revised tutorial, the player is more guided in the way that stations are locked until certain actions are performed. Also, the explained UI elements and the stations to be accessed are highlighted with animations. Additionally, we added a function to replay the last explanation. The tutorial has more explanations than before but after most explanations there is an indicator or an action has to be performed.

Another feedback we got is that people did not really pay attention to the boss enemy distance bar and thus the boss enemy appeared suddenly. Some people did not even know why the boss enemy appeared. To solve this problem we added more explanations to the tutorial and made the distance indicator bigger. Furthermore, when a boss fight is over the starship will be accelerated to the initial speed to prevent spawning of the next boss too soon.

Since the playtesters did not or could not use the bomb very often because of the high costs, we made using the bomb for free. Instead, we added a cooldown timer which prevents the strong weapon from destroying the balance of the game difficulty.

Last but not least, players often did not know the cost of using a specific station. Therefore, we added indicators for the resource costs (if there are any) to the stations.

6 Conclusion

6.1 Final Results

The final game does not differ much to our alpha release. Most of the improvements after the alpha release were done for the build that was send off to studio gobo. This included a cool down timer for our bomb stations (instead of having to spend resources to drop bombs) and improvements to the tutorial. Because our tutorial is quite text heavy we decided to also add a loading screen which gives an overview of the controls and some usefuls tips. This way players don't necessarily need the tutorial they can theoretically also jump right into the game while gathering most necessary information from the loading screen.

6.2 Experience

It is interesting to see that our core gameplay is still the same as in the designed physical paper prototype. All features we added then were also implemented in the digital version. This shows how important the prototype has been in our whole development process. However, there are some parts of the game that came up further down the road and were added like for example the boss fight or the bomb station. These features were found during development and increased the fun of the game as well as the uniqueness.

Our playtesting session showed some minor issues that we fixed afterwards but other than that we were already at a point where the game was well received by the testers. Furthermore, we were proud to see that the participants were communicating (sometimes it was more yelling though) in order to achieve their goals which was one of our personal targets during the whole process.

When we look at the initial development schedule, we can proudly say that we stuck to it almost

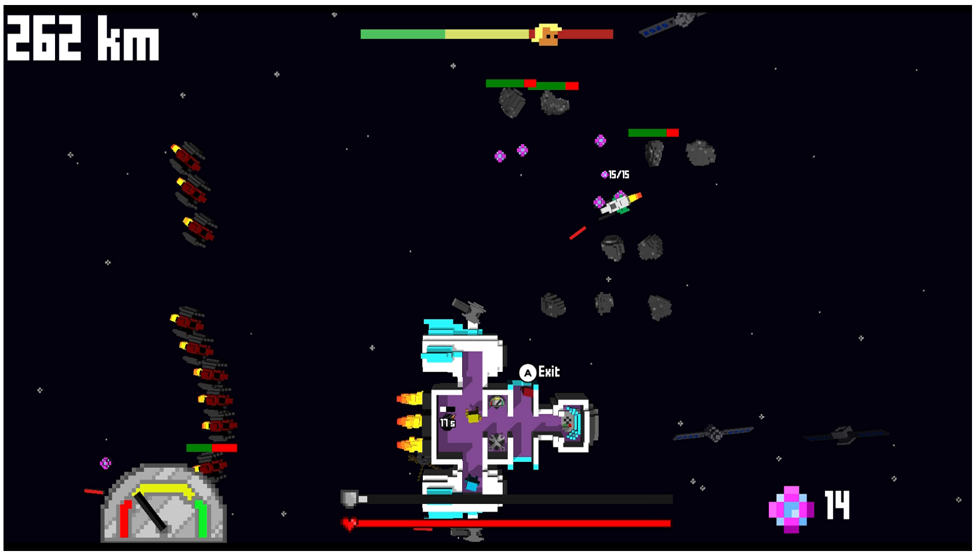
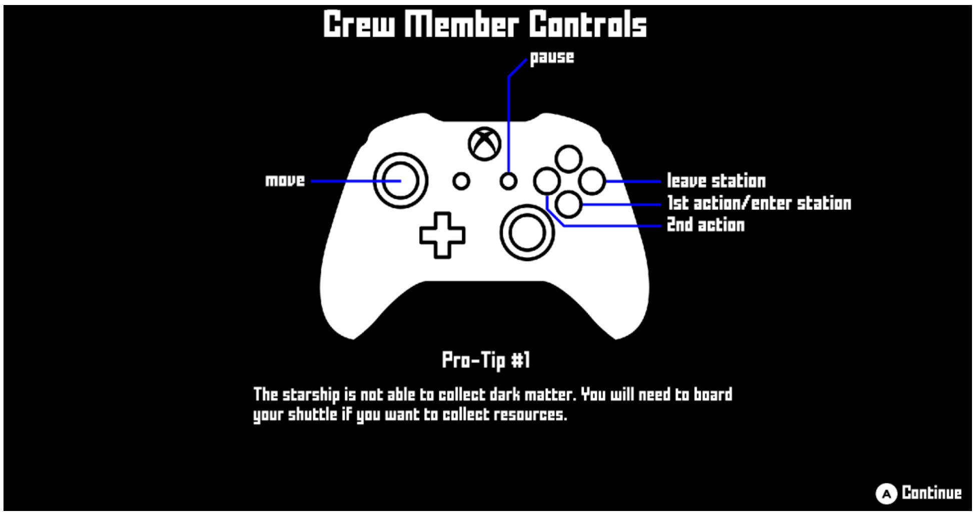


Figure 6.1: This is a screenshot from our final verison. Compared to the alpha release, we added a bomb station as a comeback mechanic and after the playtesting session we switched from bomb which costs resources to drop, to a timed version as can be seen in the picture above.


Crew Member Controls

pause
move
leave station
1st action/enter station
2nd action

Pro-Tip #1

The starship is not able to collect dark matter. You will need to board your shuttle if you want to collect resources.

Continue

Figure 6.2: In the last week we also added a loading screen. Our game doesn't really need it because the loading time is very fast but we figured it would be helpful to explain the controls and give the players some usefull tips.

all the time. Sometimes we had to change the order in which the features had been developed because of unexpected problems but at the end of each milestone we had everything as planned. In the end, the only high target that is missing is the making of different types of enemies for which we did not have enough time. However, we added a lot of additional content like the boss fights that weren't even included in our original timetable.

We believe that our successful progress is due to our strong communication between the team members and the extensive use of GitLab with its task management tools. This reduced development overlaps and merge conflicts that could cost a lot of time in such a project.

6.3 Personal Impressions

Our expectations of the game were met in every aspect. We implemented almost all features that we wanted and the visuals are very polished. In our opinion, the time that we had to develop this game was enough.

For us it is very satisfying to see that friends and family can play our game and have fun for some time. Some of them even played our game for two full hours which is quite a respectable time for a game of this scale.

Technical difficulty One of the most time-consuming tasks was to implement shadows. It was very confusing how to deal with the shadow maps and to make them actually look good.

Theme For us the theme was actually not so relevant in the idea finding process. Our idea could have been placed in different settings and therefore we adapted our game to fit with the theme.

Learnings for next game project For a next game project it would be nice to have someone that has experience in designing sound-effects and music as well as a visual artist. We did what we could in these aspects but were limited by our practical skills (for example we used Voxel art instead of a more elaborated style). In addition, this would have provided a chance to have an interdisciplinary exchange.

Success of project Overall, we see our project as a success. We met all our expectations and had a lot of fun during the whole process. The greatest success was that a lot of participants in the playtesting session said that our game was already very well polished and looks appealing. Especially the second made us proud as we did not have a ZhdK student in our team.

Monogame Monogame was a nice experience to work with as you mainly work with your own code and don't have to google for every feature you want to use. You just implement it yourself. As it has a very limited functionality, we had to code a lot on our own which helped us to understand some interesting concepts. The downside is that you're stuck writing engine code for a long time which can be a little bit stressing when working with such a tight time schedule. However, there are also some parts of Monogame that were annoying such as the lack of documentation and proper up to date examples of more advanced topics that do not only scratch the tip of the iceberg. Most of the helpful posts were written for XNA and were either nolonger compatible or incomplete because the forum shut down. Monogame also suffers from various bugs for example in the sound engine and object loading.