Game Programming Laboratory

Rafael Sterzinger - Producer
Christoph Mueller - Visual Artist
Felix Tockner - Technical Designer
Lukas Gygi - Technical Designer
Nina Willis - Visual Artist

All were Programmers and Game Designers

Contents

1 Formal Project Proposal **1**
1.1 Game Description . . . 1
1.1.1 Overview . . . 1
1.1.2 Background Story . . . 2
1.1.3 Design Decisions . . . 2
1.2 "Big Idea" Bullseye . . . 4
1.3 Technical Achievement . . . 4
1.4 Development Schedule . . . 5
1.4.1 Layered Task Breakdown . . . 5
1.4.2 Task List . . . 7
1.4.3 Timeline . . . 7
1.5 Assessment . . . 7

2 Prototype **9**
2.1 Prototype Setup . . . 9
2.1.1 First Iteration . . . 10
2.1.2 Second Iteration . . . 10
2.2 Playing Experience . . . 12
2.3 Findings and Conclusion . . . 13

3 Interim Report **15**
3.1 Progress . . . 15
3.2 Challenges . . . 17
3.2.1 Controls . . . 17
3.2.2 Textures . . . 18
3.2.3 Game Mode . . . 19
3.3 Future Work . . . 19

4 Alpha Release **21**
4.1 Progress 21
4.2 Challenges 23
4.2.1 Separation of Concerns 23
4.2.2 Character and Controls 24
4.2.3 Gameplay and Creativity 24
4.3 Future Work 24

5 Playtest **25**
5.1 Playtesting Session 25
5.2 Questions and Comments 26
5.2.1 Closed-Ended Questions 26
5.2.2 Open-Ended Questions 27
5.3 Design Revisions 28

6 Conclusion **29**
6.1 Final Results 29
6.2 Experience 30
6.3 Personal Impressions 31

1

Formal Project Proposal

1.1 Game Description

1.1.1 Overview

Our game Octodash is based on the highly versatile sea creature, the octopus. With its ability to change its appearance color- and form-wise, shoot ink, and the fact that it possesses eight tentacles, it offers itself as an ideal candidate to design a game around its capabilities while ensuring sea-thematic setting.

Octodash is a challenging 2D, two player, co-op game with its main gameplay revolving around the cooperative steering of physically simulated tentacles. By means of cooperation, i.e. jointly and strategically performing certain actions with the tentacles, two players must maneuver themselves through the depths of the sea as quickly as possible in order to reach their feast before starving to death.

With this silly-looking, quirky and maybe even sometimes frustrating movement, Octodash offers a challenging but enjoyable gameplay to both, casual and skilled, players.

In order to increase variety within the gameplay, players can collect and utilize power-ups, interact with the environment to overcome obstacles, and race through different sea-themed settings such as the deep sea, a fishing ship, or a sushi restaurant.

A far distant inspiration for the movement is the gameplay of "Octodad".

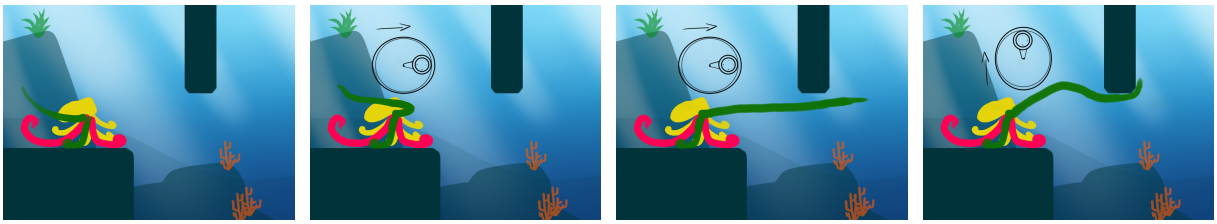

Figure 1.1: *Illustration of the basic controls*

1.1.2 Background Story

On the pursuit of satisfying basic octopus needs, our eight-armed friends are on a hunt for some delicious crabs, clams, and snails deep under the wild sea. However, for some reason on this day they are having an especially hard time finding seafood.

On the verge of giving up, one of the octopi spots a weirdly looking object swarmed by fish. Driven by hunger, it does not hesitate and immediately notifies the others of this potential source of food. Skillfully utilizing their tentacles, they start racing each other while overcoming a variety of obstacles posed by the deep sea in order to reach the feeding place first.

1.1.3 Design Decisions

A rag doll octopus is placed in a 2D level seen from the side even though the octopus might be rendered in 3D, creating a 2.5D environment. For now, however, Octodash is kept in 2D to greatly reduce the complexity of animations, physics, rendering, modeling, visual effects, etc.

Two players form a team where each of them controls two out of four tentacles of an octopus, one with each joystick. Furthermore, pressing shoulder buttons activates the suction cups of individual tentacles to stick to the surface or to objects and tentacles can stretch depending on how fast the joystick is moved to overcome a larger distance. Additional buttons might be utilized to allow for interactions with the environment/items and dashing-like movement.

These controls, if utilized skillfully, jointly, and strategically, should be enough to maneuver the octopus across the sea-themed level all the way to the finish line. A rough illustration of the basic movement mechanics are depicted in Figure 1.1.

In order to help the players differentiate between their own controllable tentacles and the others, tentacles will be colored for each player accordingly. Additionally, four out of the eight tentacles are either not modeled at all or are not controllable to facilitate gameplay.

As the controlling will feel the most natural when played with a controller, Octodash will be solely developed to support, e.g. an XBOX 360 controller. However, depending on whether keyboard and mouse controlling seem intuitive enough or not it might be supported in a later stage of development.

The tracks are styled after underwater or water-related themes. Some potential examples are an abandoned fisher ship, the deep sea, a sea cave, the inside of a whale, and a sushi restaurant.

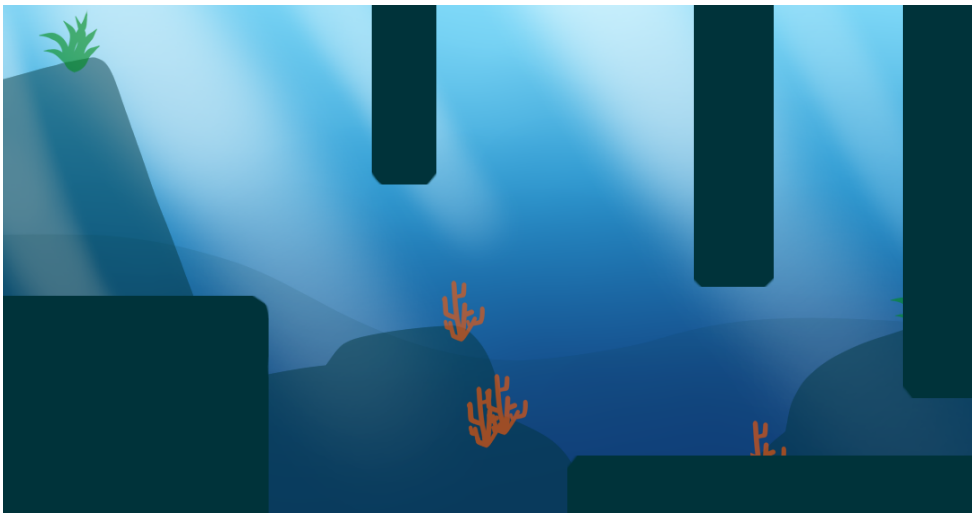

Figure 1.2: *Sketch of a section of a sea-themed racetrack*

The levels will include some challenging terrain/obstacles such as pits or walls to climb, gaps to overcome, as well as barriers that have to be removed by player interaction.

In order to make the gameplay more diverse, items such as speed boosts aid the players to reach the finish line faster but can also impede the controlling/coordination of the tentacles. On the other hand, obstacles such as e.g. jelly fish can sting the octopus which will slow the players down if not dodged in time.

As a measurement of performance, some form of stopwatch will be incorporated. For replay value of the different racing tracks, bronze/silver/gold medals are rewarded depending on the time it took the players to complete the track or for instance which way they choose to take. The later can be incorporated via collectibles that reward the players for choosing a more challenging detour.

Further down the line, depending on development progress, the interface might be extended to allow for a horizontal split-screen/zoomed-out view such that two teams can race against each other.

In order to balance out the difference in skill of two teams and to keep the gameplay more interesting, an additional blue-shell mechanic/some other form of item/power-up, special removable barriers, or opposing team-interaction might be incorporated.

As an extra goal, an online multiplayer allows players to compete/cooperate against/with each other even if they are not physically at the same place.

For the art style a low-poly look in 2.5D is planned, with a 3D model for the octopus and 2D assets for the environment. A first glance on the envisioned sea-themed art style can be viewed in Figure 1.2

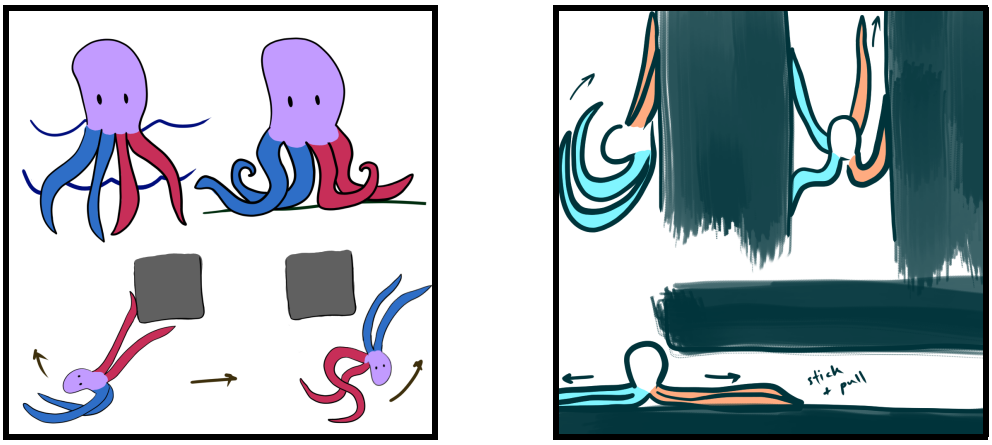

Figure 1.3: *Illustration of envisioned possible movements performed by players*

1.2 "Big Idea" Bullseye

The main idea of Octodash is that two people are stuck in one octopus, each controlling two out of four tentacles. This idea stems from the fact that octopi possess an astonishing amount of nine brains.

In order to realize this, we aim to create a simulated rag doll body resembling an octopus that has as a sole mean of moving around through actions performed jointly with the four tentacles. These controls, if utilized skillfully, jointly, and strategically, and when modeled realistically should be enough to maneuver the octopus across the sea-themed levels all the way to the finish lines.

Our "Big Idea" Bullseye is summarized in Figure 1.4.

Some movements/environment interaction we envision that players might perform via this unique controls are for instance swinging, climbing, as well as springing, i.e. building up tension to launch itself though tight tunnels. See Figure 1.3 for a more illustrative description.

1.3 Technical Achievement

Due to the highly complex anatomy of an octopus, animating all possible movements is infeasible. Instead, we concluded that it would be better and more technically challenging to make a 2D or 3D model of the octopus and then use physically-based modeling to determine the motion of the body. In order to achieve this, we plan to use existing physics libraries and then customize them corresponding to needs.

Furthermore, an optional extra goal mentioned in Section 1.1.3, is a 2v2 online multiplayer mode which comes along with the implementation of a well-performing netcode that enables enjoyable and responsive online gameplay.

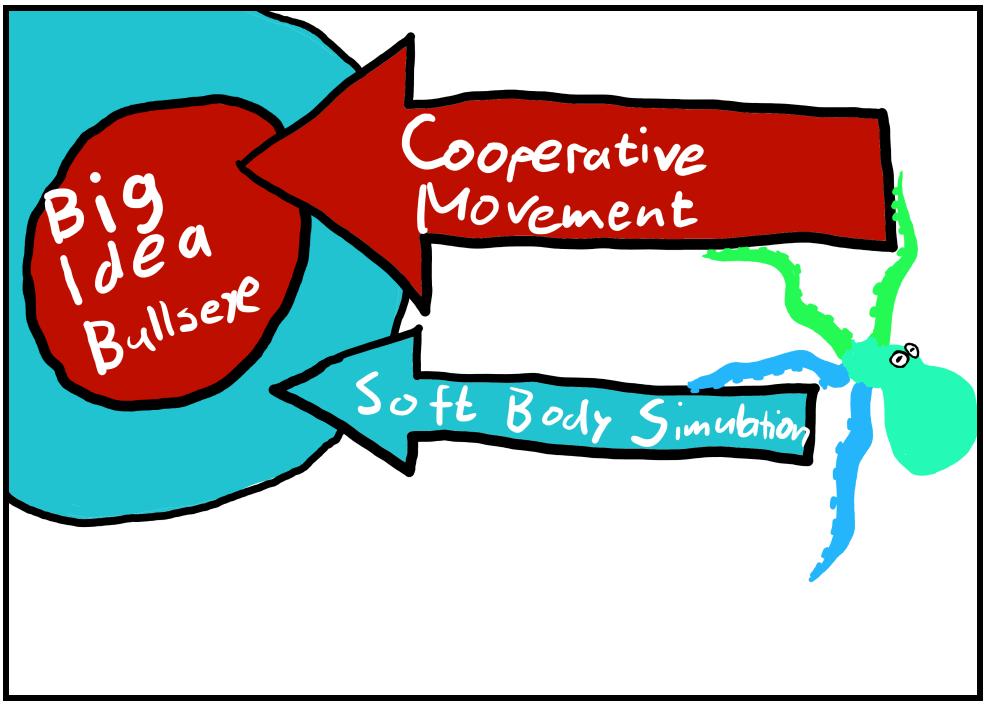

Figure 1.4: *"Big Idea" Bullseye of Octodash*

1.4 Development Schedule

1.4.1 Layered Task Breakdown

The following list depicts multiple targets which are completed over the course of the development phase in order to always guarantee a deliverable version of Octodash. The consecutive subsections also represent a list of priorities from top to bottom.

Functional Minimum

- Core gameplay
  - two controllable, stretchy tentacles
  - suction cups for interactions
  - dash-like movement
  - basic racetrack
  - controller input
- Physics system
- Rendering system
- Game engine

Low Target

- Attractive 2D model of octopus
  - Cooperative gameplay, four controllable, stretchy tentacles
- Individually colored tentacles
- Stopwatch
- Racetrack (deep sea)
  - Attractive gameplay-relevant assets

Desired Target

- Menu and UI
- Interactive objects that require cooperative gameplay
  - Movable objects (barrels, stones, ...)
  - Objects (jelly-fish, currents, spikes, ...)
  - Destructible objects (walls, barriers, ...)
- Additional racetracks (fisher ship, sea cave)
  - Attractive non-gameplay-relevant assets
- Music and sound effects
- Scoreboard
- Medals

High Target

- Tutorial/Instructions
- 2v2 mode with split screen/zoomed-out view
- Power-ups/Items
  - Speed boost
  - Extended tentacles
  - Super strength
  - Power jump
- Start and finish animations
- Collectibles, e.g. three coins

Extras

- Attractive 3D model of octopus
- Two player and 2v2 online multiplayer
- Visual effects for destructible/movable objects
- Advanced coloring system
- Mouse and keyboard support
- Random events that dynamically change the map
  - High- and low tide
  - Occurring barriers
  - Occurring currents
- Opposing team interaction
- Additional racetracks (inside a whale, sushi restaurant)
- Credits

1.4.2 Task List

(See Figure 1.5 for the list of tasks)

1.4.3 Timeline

The workload was distributed such that each student is expected to work 176 hours over the course of the semester on Octodash. This number stems from the initial course workload which amounts to 250 hours minus the time spend on lectures and preparing slides/speeches.

1.5 Assessment

The main strength of Octodash lies in the smoothness and versatility of the quirky movements of tentacle-driven octopi. These qualities will be exercised in the form of two-player cooperative control over a single octopus in an obstacle-filled race. The sticky, stretchy tentacles of the octopi and their interactions with the environments should make the game both fun and challenging as friends collaborate and compete in a race against one another. With this silly-looking, quirky and maybe even sometimes frustrating movement, Octodash offers a challenging but enjoyable gameplay to both, casual and skilled, players, regardless of age.

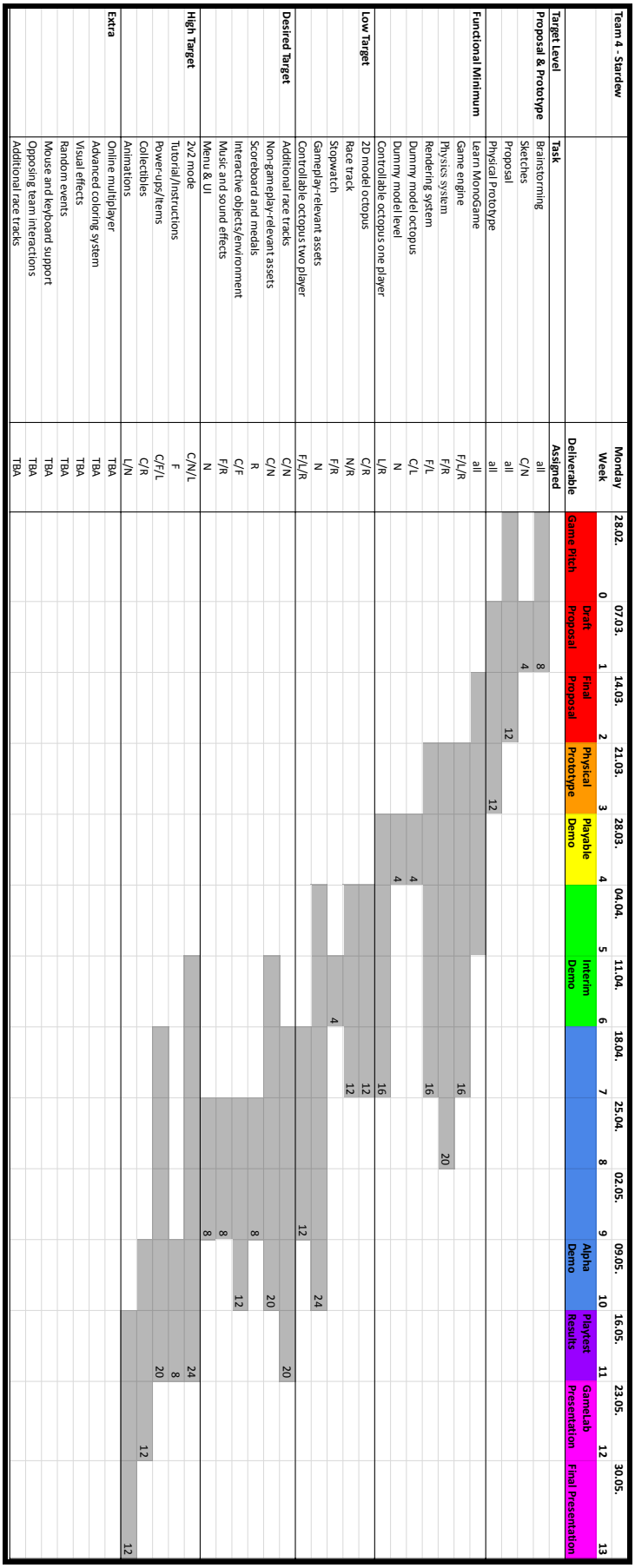

Figure 1.5: *A Gantt chart illustrating the distribution of the targets, their corresponding anticipated duration as well as their deadline. Initials are as follows: R: Rafael, C: Christoph, F: Felix, L: Lukas, N: Nina.*

2

Prototype

2.1 Prototype Setup

As an initial step before diving into exploring gameplay details, we first brainstormed on possible obstacles we could imagine appearing in a racetrack. These obstacles include: flat tiles and tiles with ascent/descent, spikes, breakable barriers, a stone to break the barrier, as well as some foes such as a fisher man and a jelly-fish. Finally, we added a crab which depicted the goal/finish line. With these obstacles, we created our first prototype racetrack by freely distributing them around the level (see Figure 2.1) in a similar fashion to a level editor.

Due to the fact that Octodash's core feature revolves around unique player inputs to maneuver the octopus around the tracks, i.e. controlling independent tentacles, building a physical prototype that incorporates this aspect well was challenging. This challenge mainly stems from the aspect that Octodash does not only involve fast-paced decision making as seen in other platforming games but on top has a complex movement mechanic which made the prototyping process ever more so difficult.

For this reason, both, prototyping the controls and feature/gameplay testing, were explored iteratively within two distinct set-ups.

1. controller for input, turn-based decision making, actions fixed by human computer (see Figures 2.1 and 2.2).
2. chopsticks for input, metronome-based decision making, actions fixed by basic rules (see Figures 2.3, 2.4 and 2.5)

Figure 2.1: *Level design of the first iteration*

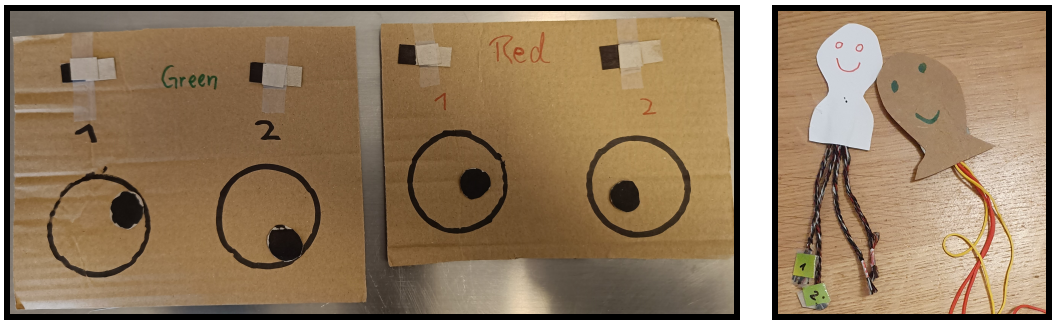

Figure 2.2: *Left: Hand-crafted controllers for movement and activating the suction cups*
Right: First iteration playable character

2.1.1 First Iteration

In a first attempt to solve the big question on how to incorporate our envisioned octopus movement we decided on a system where the cooperating players individually reveal their next action at the same time. For this, we designed two cardboard controllers (see Figure 2.2, left). At each time step, a team member acting as a computer observes these actions and moves the octopus accordingly. Finally, regarding the playable character we tried out two versions of tentacles: one with cord and one with rubber bands. Even though we imagine that the tentacles in our final game to be elastic we settled on cord-based tentacles as it worked better for prototyping.

2.1.2 Second Iteration

In the second iteration, players were able to perform the actions themselves which was realized by attaching chopsticks to the ends of the tentacles (see Figure 2.3). Additionally, as gravity plays of course a big role in Octodash, we also placed the map vertically, allowing for a more

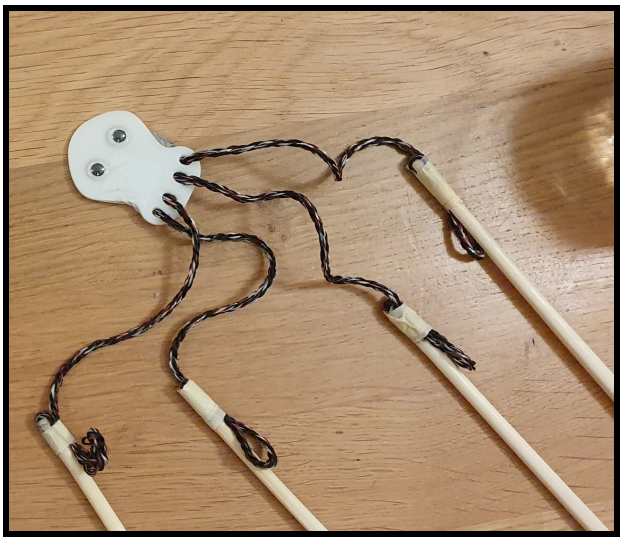

Figure 2.3: *Playable character of second iteration*

Figure 2.4: *Iterative adaptation of level design of the second iteration*

realistic gameplay experience. Due to this set-up change, the octopus as well as its tentacles clipped through the platforms and obstacles. In a final step we alleviated this problem by adding depth to our level design which enforced a realistic gameplay experience (see Figure 2.4). Since the players could move the octopus now freely, we set some basic rules on the allowed move set:

- At each beat of the metronome, both players can move one tentacle.
- Tentacles must be stuck to a platform.
- Tentacles can only pull and not push the body of the octopus.
- Tentacles can only be moved within a reasonable range.
- Only moves that do not violate these rules are allowed.

2.2 Playing Experience

The **first iteration** of the prototype was not as enjoyable as initially imagined due to the fact that between each input of the players, a sizable pause took place as they had to wait until their actions were interpreted and the octopus and its tentacles were adjusted accordingly. This made the gameplay very slow and not captivating.

Additionally, it was difficult to interpret the stickiness of the tentacles correctly. In order to facilitate the interpretation we made the movement even more granular, i.e. such that per arm only a movement or a state change in regard to stickiness can be performed, which further throttled down the gameplay.

We also observed, that it could happen that one player does continuously nothing in order to let the other player take over the control. However, this could just be a side effect of the physical prototype and might not occur if the gameplay is in real time.

Furthermore, we also noticed that maneuvering the octopus across a flat surface by simply walking is quite monotone. Despite this and all the above shortcomings, experiencing the octopus swinging from one place to another firsthand was very satisfying.

The changes made in the **second iteration**, i.e. chopsticks for inputs and metronome-based decision making made the gameplay a lot faster which lead to additional chaos and an overall more fun prototyping experience at the cost of imprecise rules. However, this also shortened the feedback-loop substantially which allowed us to try out more features and gamemodes in a much shorter period of time.

For instance, as we were not able to prototype a whole racetrack but only a small subsection of a racetrack, we tried Octodash as a non-classic race game, i.e. instead of racing linearly to the finish line, the octopi race to some source of food that is positioned at a random location at each start. Furthermore, besides the two player race mode, we also tested out a 2v2 arena mode which we think could be another potential direction for Octodash with more interesting opposing team interaction.

An epic gameplay moment of the prototyping session can be observed in Figure 2.5.

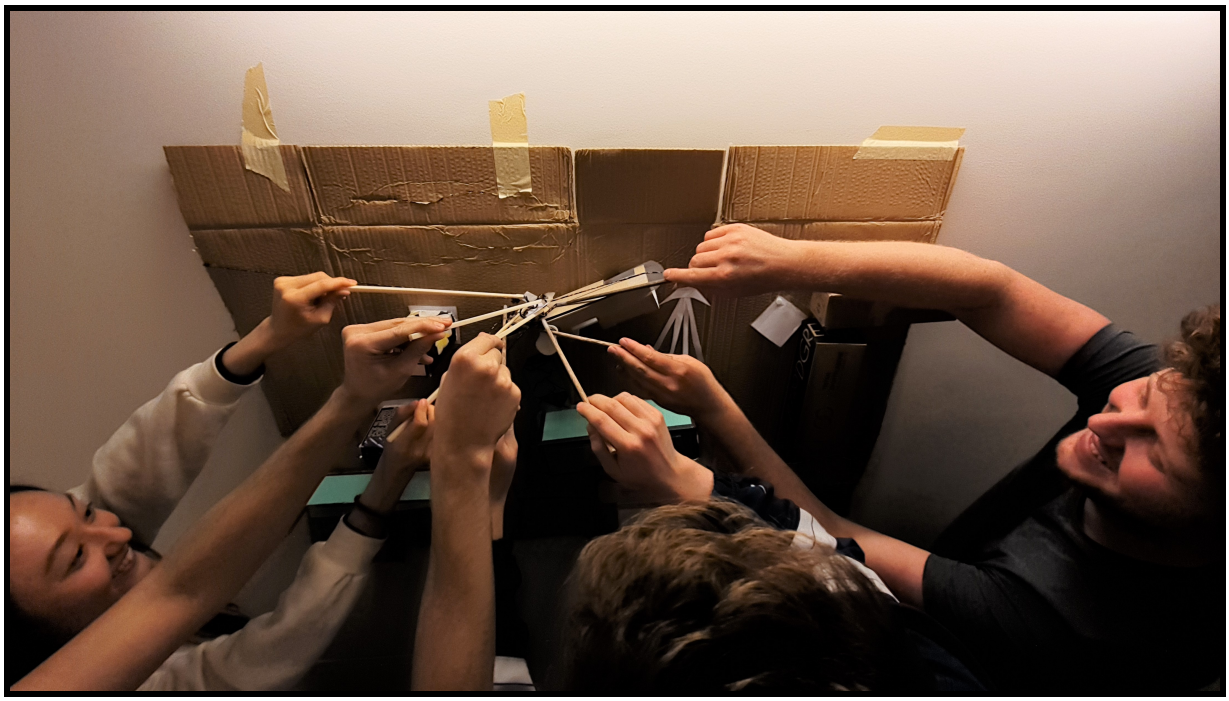

Figure 2.5: *Ingame gameplay highlight when testing the physical prototype of Octodash*

2.3 Findings and Conclusion

Octodash heavily depends on physics simulations and split second decision making of players which are two concepts we struggled to incorporate well into the physical prototype. Hence, we came to the conclusion that a small software prototype might be more helpful for testing the movement in detail and for further decision making. Nevertheless, it was an overall fun experience and we managed to get a couple new findings/ideas:

- During the first iteration we noticed that the horizontal movement felt very clumsy and unpleasant. We aim to alleviate this by implementing low gravity within the game, which will hopefully make the movement on flat surfaces more dynamic and fun. Moreover, we plan to design levels such that the players are given a more varied option of locomotion such as swinging and climbing and additional mechanics such as bouncy platforms and currents which interfere/aid with the movement.
- Once a team has played through a racetrack a couple of times, coordination will be optimized and communication reduced. To alleviate this problem, we consider an arena type gamemode where players are trying to collect as many crabs as possible within a given time frame. Crabs would randomly spawn on the map and the players would have to navigate to them as quickly as possible. This incorporation of randomness might not only increase replay value but also ensure that each round offers a unique experience and as a result communication would remain a vital part of the gameplay.

3

Interim Report

3.1 Progress

In the first iteration of the development process, we heavily focused on implementing all the underlying systems of our game (Physics and Rendering system, Game engine) in an easily extendable and robust manner such that development will be facilitated further down the line.

After this initial phase, we attempted to select tasks more granular such that all team members can work individually to maximize productivity. As a result, some team members advanced quicker to higher target layers while others continued working on lower target ones.

As of now, the current state of the game at the Interim Demo includes all aspects of the Functional Minimum and almost everything from the Low Target. In the latter case, we decided to only postpone the refinement of our first racetrack as the final game mode of our game will heavily influence the level design and is still up for debate. However, to compensate for this, we have already started with the development of features of the Desired Target.

An exact breakdown of the current state can be seen below. Furthermore, Figure 3.3 gives a visual illustration of the progress made so far and also showcases the advancements made between the First Playable Demo and the Interim Demo.

Functional Minimum

- Core gameplay — Done
  - two controllable, stretchy tentacles
  - suction cups for interactions
  - dash-like movement

  - basic racetrack
  - controller input
- Physics system — Done
- Rendering system — Done
- Game engine — Done

Low Target

- Attractive 2D model of octopus — Done
  - Cooperative gameplay, four controllable, stretchy tentacles
- Individually colored tentacles — Done
- Stopwatch — Done
- Racetrack (deep sea) — In development
  - Attractive gameplay-relevant assets

Desired Target

- Menu and UI — In development
- Interactive objects that require cooperative gameplay
  - Movable objects (barrels, stones, ...) — In development
  - Objects (jelly-fish, currents, spikes, ...) — Not yet implemented
  - Destructible objects (walls, barriers, ...) — Not yet implemented
- Additional racetracks (fisher ship, sea cave) — In development
  - Attractive non-gameplay-relevant assets
- Music and sound effects — Not yet implemented
- Scoreboard — In development
- Medals — Not yet implemented

High Target

- Tutorial/Instructions — Not yet implemented
- 2v2 mode with split screen/zoomed-out view — Not yet implemented
- Power-ups/Items — Not yet implemented

- Start and finish animations — Not yet implemented
- Collectibles, e.g. three coins — Done

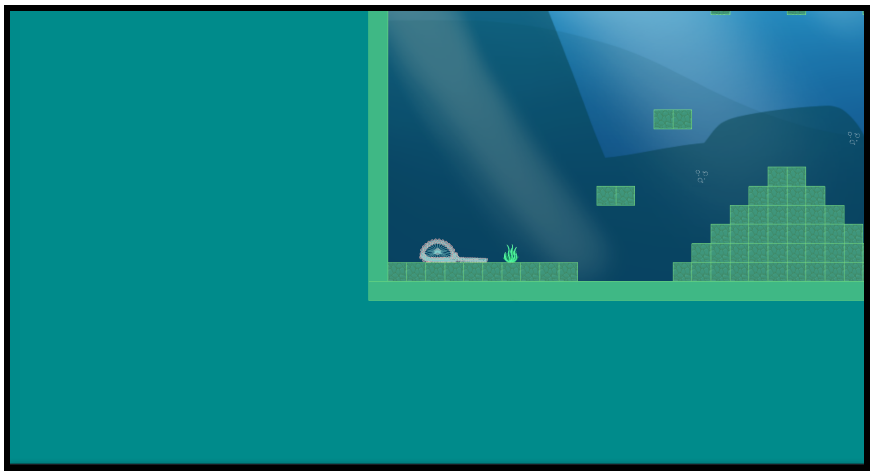

Figure 3.1: *First Playable Demo*

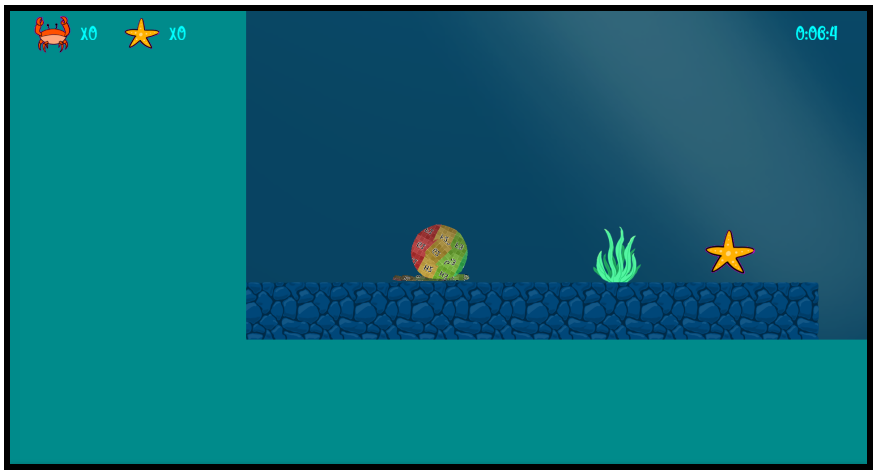

Figure 3.2: *Interim Demo*

Figure 3.3: *Visual overview of the progress made between the First Playable- and the Interim-Demo*

3.2 Challenges

3.2.1 Controls

Being able to control a character that reacts and deforms according to player-inputs turned out to be incredibly satisfying. However, mass-spring systems which we use for the softbody simulation are notoriously unstable if not fine-tuned exhaustively. Moving the body too fast or stopping it abruptly can lead to a collapse, e.g. the center body gets pushed outside the enclosing ring of bodies. To solve this issue, we had to come up with, test, and fine-tune a lot

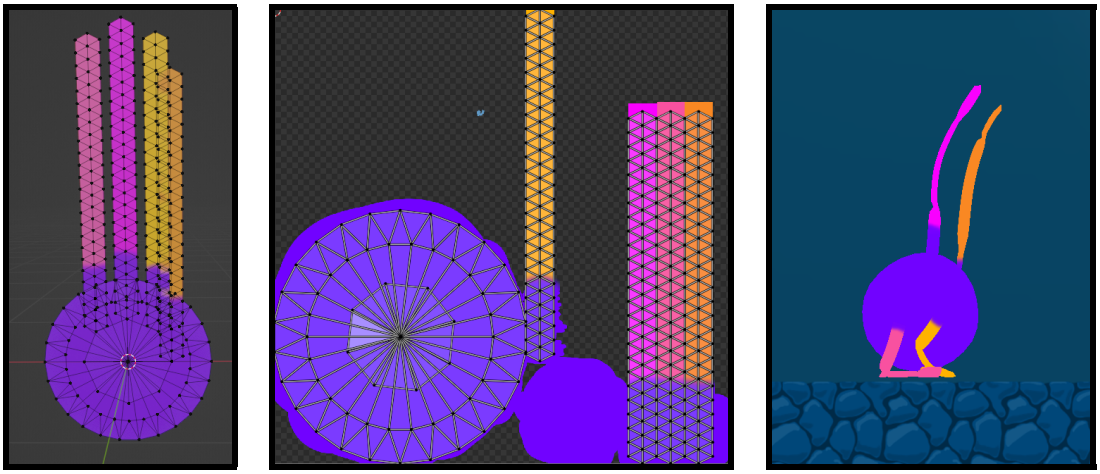

Figure 3.4: *Left: Model exported from the physics engine ingame*
Middle: UV unwrapping of the model
Right: Final result in game

of different architectures to make the mass-spring system stable. Unfortunately, this resulted in a less squishy but finally stable octopus which, however, can sometimes feel sluggish. Ideally, in the next iteration, we are able to accomplish all: a stable, squishy, and vigorous octopus.

3.2.2 Textures

Another challenge we faced and want to point out is the texturing of the softbody. Since the shape of the character model is dictated by the physics engine ingame, we had to come up with a method that allows for dynamic deformation of the texture. For this, we first exported the model, i.e. the vertices and edges of the character model as a file which we could then import into Blender.

In Blender, we performed UV unwrapping to create a texture for the character model that we then exported together with added UV coordinate information into two new separated files. In code, we then load the previously created files and use the `BasicEffect` shader with a `VertexPositionTexture` which provides the option to draw the mesh as a list of vertices, UV coordinates, and a texture.

During each `Draw()` call we update the positions of the vertices in the `VertexPositionTexture` according to the current state of the softbody provided by the physics engine. With this approach the texture is drawn dynamically and as a result allows for deformation which makes the texture of the octopus appear squishy.

One problem, however, we noticed with this method is that arms can become quite thin which makes them visually unsatisfactory. This is based on how arms are currently modeled by the physics engine and to counter act against this we need to make some more refinements to this approach in next iterations.

3.2.3 Game Mode

A final challenging aspect we want to mention is the decision/agreement on the final game mode of Octodash. This topic was heavily discussed throughout meetings and is still undecided. The main question we are faced with is whether movement alone will be interesting enough to keep two players captivated. As of right now, we decided to iteratively refine the gameplay and add features to increase cooperation and engagement where necessary.

3.3 Future Work

In the next iteration, we aim to focus more on the general gameplay experience, i.e. visuals, the user interface, and music/sound effects or more generally speaking: refactoring and polishing the current state.

This includes refactoring the arms and the body to allow for a better separation of concerns and increased code readability. Additionally, the current texturing mechanic of the softbody does not display the arms properly and thus also needs refinement for better visuals.

Further adjustments to the controls/the mass-spring system will be postponed for now until the game mode/objective is clear and agreed upon. For this, we want to start as soon as possible with playtesting cooperative gameplay with four arms to see if our current vision is sufficient action-, fun-, and interaction-wise or if further additions to the gameplay are necessary to keep the players captivated.

4

Alpha Release

4.1 Progress

With the Alpha Release of Octodash most of the features of the Desired Target are completed. Solely an additional racetrack and medals are features that are still missing but are in development. A breakdown of the current state of the game can be seen down below. Furthermore, Figure 4.3 depicts the progress made between the Interim Demo and the Alpha Release.

Functional Minimum

- Core gameplay — Done
  - two controllable, stretchy tentacles
  - suction cups for interactions
  - dash-like movement
  - basic racetrack
  - controller input
- Physics system — Done
- Rendering system — Done
- Game engine — Done

Low Target

- Attractive 2D model of octopus — Done
  - Cooperative gameplay, four controllable, stretchy tentacles
- Individually colored tentacles — Done
- Stopwatch — Done
- Racetrack (deep sea) — Done
  - Attractive gameplay-relevant assets

Desired Target

- Menu and UI — Done
- Interactive objects that require cooperative gameplay
  - Movable objects (barrels, stones, ...) — Done
  - Objects (jelly-fish, currents, spikes, ...) — Done
  - Destructible objects (walls, barriers, ...) — Done
- Additional racetracks (fisher ship, sea cave) — In development
  - Attractive non-gameplay-relevant assets
- Music and sound effects — Done
- Scoreboard — In development
- Medals — Not yet implemented

High Target

- Tutorial/Instructions — In Development
- 2v2 mode with split screen/zoomed-out view — Not yet implemented
- Power-ups/Items — Not yet implemented
- Start and finish animations — Not yet implemented
- Collectibles, e.g. three coins — Done

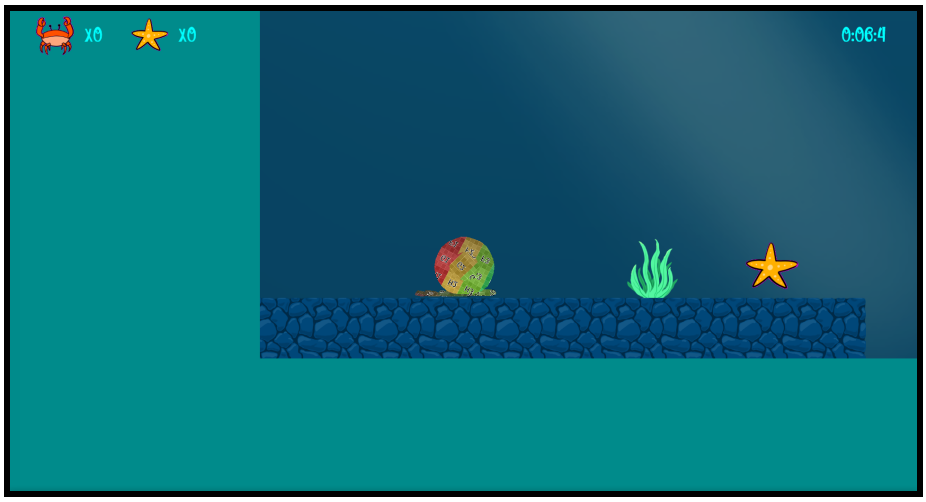

Figure 4.1: *Interim Demo*

Figure 4.2: *Alpha Release*

Figure 4.3: *Visual overview of the progress made between the Interim Demo and the Alpha Release*

4.2 Challenges

4.2.1 Separation of Concerns

One of the biggest challenge we faced during this development phase was concerned with the code structure. Creating code that is modular, readable, and robust proved to be harder than we anticipated. Throughout the project we aimed to separate tasks in a way such that everyone can progress individually. Even though this approach has a lot of upsides, it can also be the cause of difficult resolvable merge conflicts. In the beginning of Octodash, this was not a problem as the code base was still small and fairly dependency free. However, over time the project grew both in size and complexity. This made collaboration ever more so complicated and slowed down

our progress as file hierarchies and class imports changed more frequently.

4.2.2 Character and Controls

Since our softbody character in Octodash is simulated by a physics engine, we also decided to implement all of its movement mechanics using the same physics engine. However, this poses significant challenges for creating more advanced movement as, for instance, the impact of too strong forces can break the softbody. As a result, we had to keep the movement options fairly simple. However, movement based on physics is still diverse and offers a fun and enjoyable experience despite the limited controls.

4.2.3 Gameplay and Creativity

Although developing a game with a game engine such as MonoGame can offer more creative freedom, we also witnessed its downsides. Mainly, producing a game with such a game engine means that more time is spent on the technical side and less on actual gameplay. During the development phase of the Alpha Release, we noticed this shortcoming by being very limited time-wise in regard to quickly prototyping additional gameplay mechanics as it would consume too much time.

4.3 Future Work

In parallel to play-testing the Alpha Version of Octodash, we aim to polish the current state by adding more assets and to resolve discovered and not yet discovered bugs. Other planned next steps include trying to further increase the robustness of the softbody, adding a tutorial and an additional racetrack, as well as incorporating the feedback obtained during play-testing.

5

Playtest

5.1 Playtesting Session

During playtesting week, each team member was occupied during different times and as such we were unable to commonly organize a public playtesting session. Instead, we all individually invited a couple of friends to play Octodash while providing food and beverages. Additionally, we sent out our game to friends and acquaintances where they were able to playtest Octodash by themselves.

Based on this, the received feedback must be taken with a grain of salt as it will be in favor of us. However, these playtesting sessions still allowed us to gain valuable insights on how to improve our game.

In general, all playtesting sessions took around 30 minutes and were mostly played by two players, as intended for our game. After each session, the players were given a short questionnaire to fill out, the results of which are presented and analyzed in the following section. About the questionnaire, it was designed to not only figure out general trends but also to finalize gameplay decisions we had difficulties agreeing on.

Regarding the results, the overall sample size was unfortunately rather small which is mainly due to the fact that in many cases we were unable to provide the required controllers as Octodash cannot be played with mouse and keyboard. This meant that our pool of possible playtesters was already greatly reduced from the beginning.

Lastly, some snapshots, taken during a playtesting session, are displayed in Figure 5.1.

5.2 Questions and Comments

The following section summarizes the results obtained during our individual playtesting sessions. Overall, Octodash received a very positive feedback and with slight adjustments we hope that it will offer a great playing experience to everyone.

5.2.1 Closed-Ended Questions

- How would you rate the game? (1 to 10)

  Our game Octodash received an average rating of 7.6 with 10 being the best achievable score. Although we were hoping for a better score, we are still happy that our game was overall positively perceived.

- Did you play with another player?

  In order to facilitate playtesting for people who did not have a playtesting partner, we quickly implemented a single-player version of the game. However, in the end all except one participant played the game together with another player. Furthermore, this person indicated that playing with another player would have greatly improved the gameplay experience which aligns with our perception.

- How important was cooperation to the gameplay? (1 to 6)

  Here, Octodash achieved an average score of around 5.3 indicating that cooperation is a vital aspect of our game. As in the beginning it was unclear what cooperative gameplay would mean for our game, we are glad that with our design choices we were able to achieve this goal.

- Did you feel encouraged to communicate during the game?

  Building upon the previous question, all participants indicated that they felt encouraged to communicate during the game which we feel is an important aspect of a party/racing game such as Octodash.

- How did the controls feel? (1 to 6)

  This question obtained very mixed answers, achieving an average rating of 3.5 where 6 would be the best rating. This is inline with our vision of sightly frustrating but interesting controls which might become more and more satisfying over time as players perfection them. Additionally, this illustrates that Octodash possesses a learning curve which increases replaying value.

- Was the objective of the game clear? (1 to 6)

  For this question, Octodash received an average rating of 4.6 which we view as very critical feedback that we must incorporate as a game with no clear objective is certainly not fun to play.

- Did you have any trouble navigating the level and finding the objectives? (1 to 6, linear scale)

Similarly to the previous question, Octodash obtained an average rating of 4.8, emphasizing on the fact that the clearness of the game objective needs to be improved.

- How would you rank the difficulty of the game? (1 to 6)

  Regarding the difficulty of the game, it seems to be well enough balanced with an average rating of 3.5 where 6 would be equivalent to "frustrating to play". In a similar fashion to the question about the controls, Octodash offers a challenge for players with the possibility of mastering the movement as well as the game itself.

- How long did it take you to complete the level, excluding time penalties?

  On average, our playtesters spent 23 minutes on average in our test level with some of them not even completing it. For instance, there were two participant who were able to collect just 3 out of 10 crabs in around 40 minutes. This is also reflected in the rather large standard deviation of 11 minutes.

- How did you feel about the length of the level? (1 to 6)

  As hinted by the results of the previous question, the length of the level was the most criticized aspect of our game, achieving an average rating of 4.9 which indicates that the designed level was too long for the average player.

5.2.2 Open-Ended Questions

- Were there any particularly satisfying aspects?

  The most enjoyed aspects of our game were the music, the interaction with the environment (hitching a ride, breaking a wall), the controls after getting used to it, as well as the overall art style.

- Was there anything that you found frustrating?

  The most frustrating aspects indicated by many participants were the length of the level as well as locating their corresponding tentacles. Additional problems were posed by our design choices. For instance, it was pointed out that it was frustrating that the octopus detaches after it hits a spike or that crabs were difficult to find.

- Is there something that you feel is lacking in our game?

  The participants mentioned a couple of features they found lacking in our game on which we agree on as well. However, the following list includes lacking features we are aware of but will not include due to time constraints. For instance, it was pointed out that enemies or a 2v2 mode would increase the fun.

  Furthermore, some additions were mentioned which we wanted to add but did not find the time yet to do so. For example, it was pointed out that Easter eggs, additional levels, and more sound effects would be a great addition.

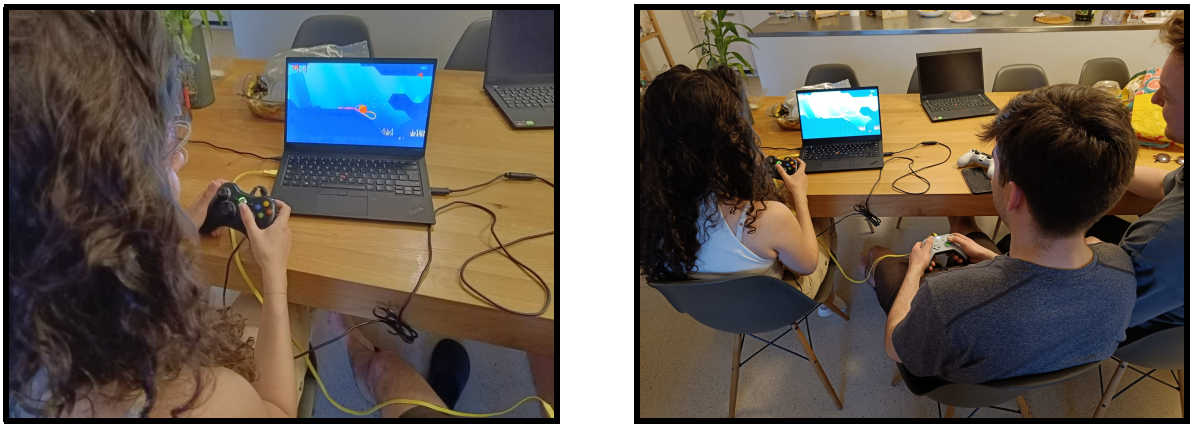

Figure 5.1: *Exemplary snapshots of one playtesting session hosted during playtesting week*

5.3 Design Revisions

Despite the fact that, overall, Octodash received very good feedback, there were some hints we found particularly useful and adding them would greatly improve the overall game. Among these hints, we are mentioning the most striking ones that we incorporated into our game.

For instance, to make the objective as well as the overall controls more clear, we added a tutorial level to lower the entry barrier for new players. Furthermore, we had to change the length of the level as many respondents complaint about this. However, we wanted to offer a variety of options such that both, new and advanced, players have a satisfying experience. Based on this, we split the original level into two separate ones as well as kept the original one to be experienced by the professionals. Overall, as of now, Octodash offers three different levels plus a tutorial level. Finally, another highly requested change was with regard to the distinctiveness of the tentacles. For this, we completely overhauled the design of our character which improved clarity by a lot.

We hope that with these changes we can increase the rating and improve the overall gameplay experience.

6

Conclusion

6.1 Final Results

Octodash is a challenging 2D, two player, co-op racing game with its main gameplay revolving around the cooperative steering of physically simulated tentacles. By means of cooperation, i.e. jointly and strategically performing certain actions with the tentacles, two players must maneuver themselves as quickly as possible through the depths of the sea in order to collect all the crabs and satisfy the food cravings of their octopus. During their journey, the players have to overcome spikes, break walls, and might hitch a ride from cute little seahorses when needed. With this silly-looking, quirky and maybe even sometimes frustrating movement, Octodash offers a challenging but enjoyable gameplay to both, casual and skilled, players.

Since the Alpha Release, we performed many adjustments to Octodash which mainly revolve around the gameplay experience but not additional features. For instance, we completely overhauled the character model (see Figure 6.1) and adjusted the controller layout. Additionally, we added restrictions on the tentacle span as well as the falling speed and included a stun effect

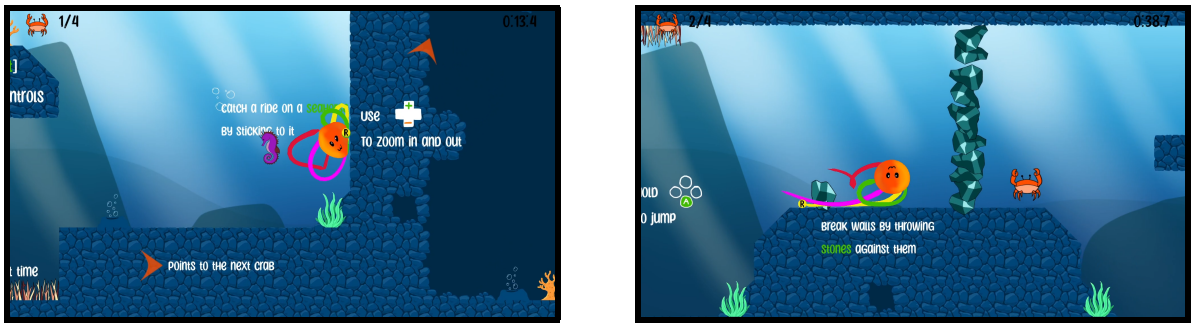

Figure 6.1: *Exemplary snapshots of the overhauled model and the tutorial level*

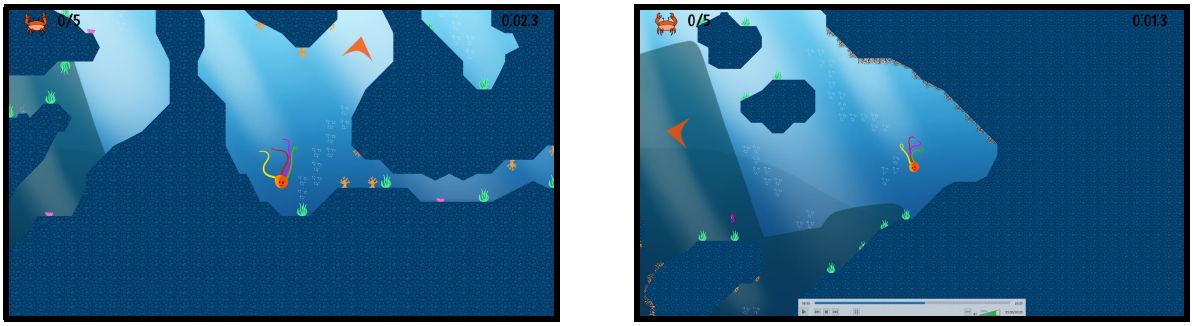

Figure 6.2: *Exemplary snapshots of the spawning points of the newly created levels*

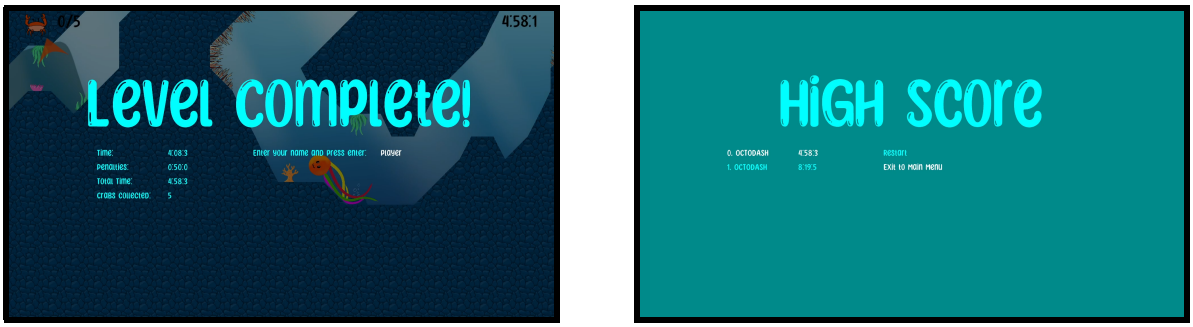

Figure 6.3: *Exemplary snapshots of the scoreboard*

when the players hit spikes.

Furthermore, we also made adjustments to the level as our Alpha Release version was generally perceived as too long. In order to resolve this issue, we split the original level and made two additional ones (see Figure 6.2). On top of that, a tutorial level (see Figure 6.1) was included to get new players quicker acquainted with our unique controls as well as illustrating the objective of our game better. Other general changes about the level include determinism such that every route is the same which in order enables the possibility of speedrunning our game, as well as additional sound effects and level assets.

We also added a per-level scoreboard (see Figure 6.3) to Octodash to give players insights about their performance as well as the persistence of scores and settings.

6.2 Experience

Looking back, our initial ideas of having a cooperative game with unique and fun movement is still very much at the core of our game. Moreover, we really wanted our playable character to be squishy which we managed to achieve by implementing a softbody simulation in the form of a mass-spring system. However, implementing a stable and controllable softbody simulation proved to be a lot harder than we anticipated. As a result, we had to adjust the development schedule as we assigned one of our team members to work solely on the softbody. Nevertheless, the project structure helped us to keep an eye on the most important aspects of our game.

Unfortunately, it was difficult to implement our initial game idea with physical prototyping and as such was not of great help to us. However, the playtesting phase helped us realize that the current version of the level was too long and, based on this, we made adjustments accordingly.

6.3 Personal Impressions

- What was the biggest technical difficulty during the project?

  The most hours definitely went into making the softbody of the octopus work smoothly. Finding a mesh structure that would not collapse, dent, implode or explode, tweaking the spring stiffness, the dampening factor, the size and the weights of the rigid bodies as well as creating a balanced and believable control system with forces and counter forces, consumed a large portion of our time. However, a lot of other tasks also revolved around the core topic of making the octopus feel fun to control. To name a few:

  - Translating units and camera projections between the physics engine, MonoGame, and Tiled such that collision boundaries are set correctly and align with what is shown on screen and using this to iterate the softbody.
  - Creating our own system for extracting the mesh topology for UV mapping and using this map to correctly map back onto the mesh (mass spring system) of the body of the octopus to render it with effect shaders.
  - Implementing our own force-based camera system (solving the ODE for velocity and position via symplectic Euler integration) that follows the octopus but remains smooth and controlled even through glitches and fast flings.
  - Iterating on the texture of the octopus and providing audible and visual hints to let players know when an arm is connected and which joystick controls it.

- What was your impression of working with the theme?

  The theme definitely helped as a starting point since none of us had a game idea at the beginning of the course. It provided something to talk about during the brainstorming sessions which really helped the process along. We also did not really feel restricted by the theme since it still left a lot of room for creativity.

- Do you think the theme enhanced your game, or would you have been happier with total freedom?

  Having a theme assigned to our house helped with finding a vision since it gave us a starting point for the brainstorming sessions. Moreover, due to the thematic constraints, we did not discuss the setting as much and could focus more on providing a unique experience within the given theme.

- What would you do differently in your next game project?

  We would definitely start earlier with building a software prototype to iterate on the core gameplay. We would potentially also focus earlier on closing the gameplay loop in hopes of getting more time to iterate on that one as well.

- What was your greatest success during the project?

  Of course it is difficult to pinpoint one single greatest success in an iteratively progressing project that consisted of many milestones. One moment we celebrated was when the octopus had a face for the first time. We were avoiding shaders from the start but when we finally completed the last stage of the pipeline to render the octopus with effect shaders, we felt a genuine sense of accomplishment. It was a large step towards making our game take shape.

- Are you happy with the final result of your project?

  We were able to meet most of our desired target and some of our high target goals given the limited time frame. There is certainly room for improvement such as visuals, UI, and level design, and there are many possibilities for fun game mechanics and environmental interactions that can still be explored. But despite our limitations, we managed to create a complete game that is unique, cooperative, and fun to play, and we received lots of positive feedback from playtesting-session attendees. Overall, we are quite happy with the final result of our project.

- Do you consider the project a success?

  During the public play session our booth was busy and the feedback of the attendees was very positive. Most importantly, the players and we were having a lot of fun. Hence, we consider the project a big success.

- To what extend did you meet your project plan and milestones (not at all, partly, mostly, or always)?

  Overall, we believe that we met our goals mostly on time. For instance, we anticipated to complete the rendering system as well as the physics system between the Interim Demo and the Alpha Release but were able to accomplish those sooner. On the other hand, the character model was planned to be done by the Alpha Release but was fine-tuned until the Final Release.

  Regarding the different layers, we completed almost everything until the High Target and even finished some tasks of this layer. Towards the end, our priorities shifted and we placed more weight on providing a complete game experience. Based on this, we reused existing assets to build more levels and used the gained time to build a tutorial level.

- What improvements would you suggest for the course organization?

  Overall, we felt that the ideation stages were a bit long in relation to game development time, especially, considering how people generally get busier towards the end of the semester. Additionally, since there were so many students this semester, it could have been helpful to separate the houses in some sessions such as for draft proposal presentations and first playable demos, to have more time for feedback and discussion, and have all houses meet for lectures and more showcasing-type presentations, like interim demos. It also would have been nice if the prototyping format was more flexible as the physical prototype was not a great fit for efficiently capturing the game feel and mechanics of our more simulation-oriented game.

- Did you like using MonoGame?

Although, most of us have never worked with MonoGame before, it offered a decent learning experience. We agree that MonoGame can to some extend enable creative freedom but in others also hinder it. For instance, not being allowed to rely on tools provided by more prominent game engines enabled us to truly reflect which tools we will need to build our vision and not just check what can be accomplished with existing ones. However, building everything from ground up slows down development in other areas. For instance, we felt that based on this additional overhead we were unable to properly explore more environment interactions.