A FRACTION OF TIME (GDD)

How to split reality (again and again)

A project by Fabian Kowatsch, Jonathan Rißler, Josue Buck, Lukas Willmann, Nick Häcker, Thyra Hultsch and Umberto Falkenhagen

HFU Furtwangen
Game Design Workshop

Contents

A FRACTION OF TIME

- scientific elaboration -

CONFIDENTIAL
CONFIDENTIAL
CONFIDENTIAL

STORY

The protagonist is a scientist who invented a device to unite himself with versions of himself from other realities. However, when trying it out, the device has a malfunction and instead of pulling the alternative version into his reality, he gets teleported into the reality of other versions. Since the malfunction remains, the protagonist is now caught in the alternative realities. Due to the current function of his device, his multi-reality journey will lead him from one alternative version of himself to another, getting to know them and hopefully being able to convince them to follow on his journey and help him find a way home.

INTERNAL GOALS

- Challenge the player with complex puzzles based on an innovative core mechanic (splitting reality)
- Immerse the player with interesting alternative realities and make them think about the many directions into which our reality could have gone

EXTERNAL GOALS

- Advance knowledge about game design elements
- Better understanding of prototyping games and how to work with Unity
- Creating an immersive game with well-designed puzzle mechanics

GAMEPLAY GOALS

- Give the player opportunities to solve the puzzles in creative and possibly different ways
- The Player has to develop a good understanding of the core mechanic to be able to solve the puzzles and achieve the highest possible score (based on factors like speed, completeness, side goals)

TARGET AUDIENCE

Especially puzzle game enthusiasts, brain-teaser lover, people who are interested in the concept of alternative realities and fans of games like Portal will enjoy "A fraction of time".

Platform: PC

Engine: Unity

Genre: 3rd-Person-Puzzle-Stealth Game

Game Style: Low Poly

1. EXECUTIVE SUMMARY

Alternative realities were always a fascinating thing. The idea of an endless number of realities that are based on the same origin but were split up during time by sometimes smaller or larger events in history offers similarly endless ways of painting unique, yet familiar universes as has happened numerous times in all areas of modern pop culture.

"A fraction of time" is based on this idea. The protagonist is a scientist who invented a device to unite himself with versions of himself from other realities. However, when trying it out, the device has a malfunction and instead of pulling the alternative version into his reality, he gets teleported into the reality of that other version. Since the device's malfunction remains the protagonist is now caught in the alternative realities as there is no version of himself in his original reality. Now the scientist is forced to go forward and travel the multiverse until he can find a way to fix the device to travel back home. Due to the current function of his device, his multi-reality journey will lead him from one alternative version of himself to another, getting to know them and hopefully being able to convince them to follow on his journey and help him find a way home.

While playing, the user will experience…

- Solving puzzles based on an innovative mechanic to split reality becoming more complex with every other version they collect
- Various alternative realities that show "what if" scenarios and give a feeling of the endless possibilities that lie within the theory of alternative realities

1.1. Gameplay Summary

- Puzzle game with stealth game elements, using alternative versions ("AVs") through reality splitting to solve puzzles
- 3rd person game
- Solving puzzles consists of 2 major parts:

    - Examining the puzzle elements and planning the order of tasks
    - Executing the planned steps in the correct order
- Every AV has a certain basic skill that has 2-3 different usage variations within the puzzles (e.g., the strength basic skill can be used to push/carry/throw heavy objects)
- Gameplay loops between the reality map (kind of a world map where the player can select the levels they want to play) and the respective levels with the puzzles
- Gameplay follows a linear story structure (player actions have little to no effect on the development of the story) in a semi-linear level structure (the overall level goals are determined by the story while the player can go different ways within a level to reach them)

1.2. Goals

1.2.1. External Goals

- Advance knowledge about game design elements
- Better understanding of prototyping games and how to work with Unity
- Creating an immersive game with well-designed puzzle mechanics

1.2.2. Internal Goals

- Challenge the player with complex puzzles based on an innovative core mechanic (splitting reality)
- Immerse the player with interesting alternative realities and make them think about the many directions into which our reality could have gone

1.2.3. Gameplay Goals

- Give the player opportunities to solve the puzzles in creative and possibly different ways
- The Player must develop a good understanding of the core mechanic to be able to solve the puzzles and achieve the highest possible score (based on factors like speed, completeness, side goals)

1.3. Summarized Product Details

Game title: "A fraction of time"

Game System: PC

Input System: Keyboard and mouse

Engine: Unity

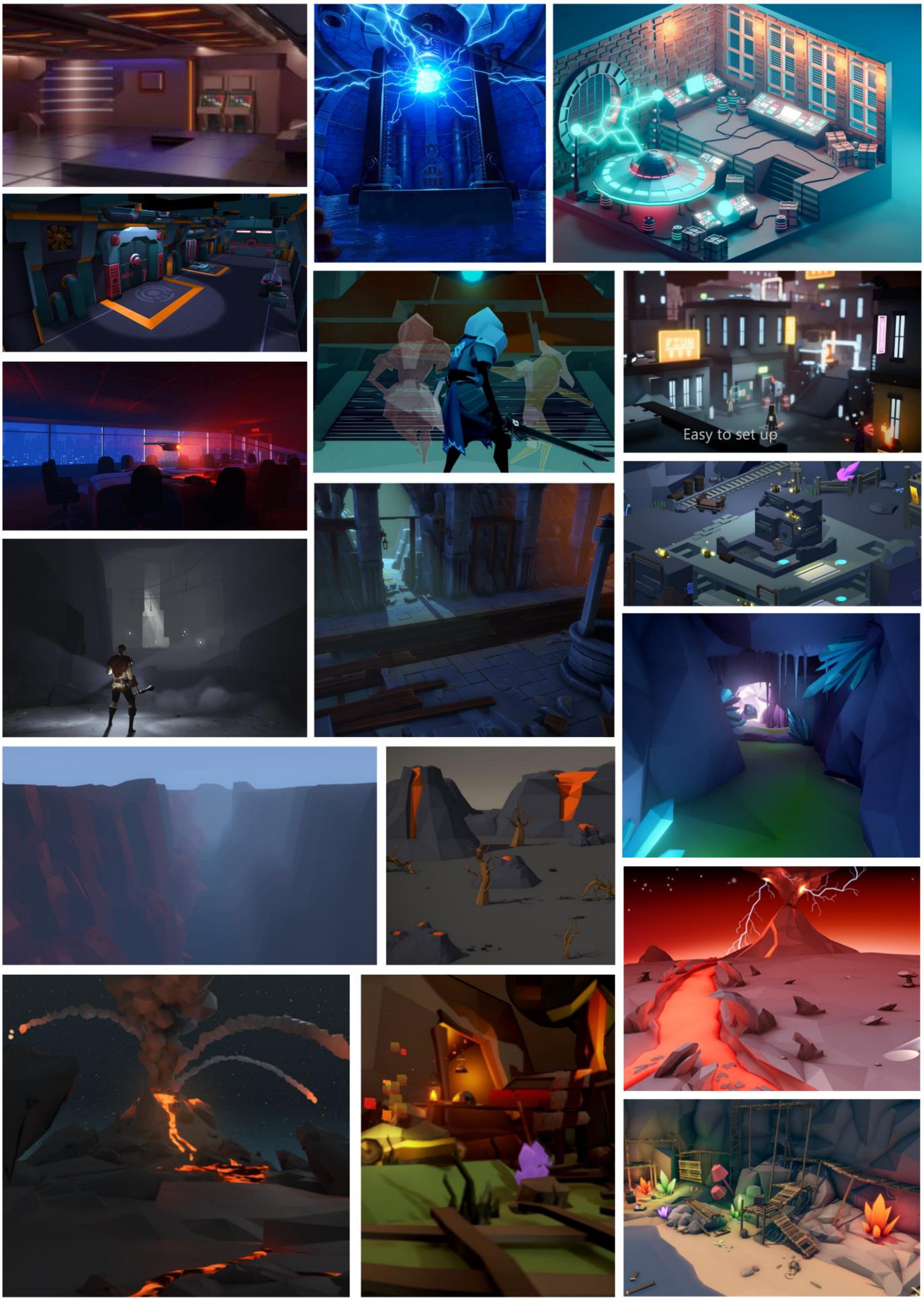
Target audience: puzzle game enthusiasts, people who are interested in the concept of alternative realities, fans of games like Portal, ...

USP's:

- ➔ The unique and innovative puzzle mechanic
- ➔ Experiencing a diverse collection of alternative realities

Art-style: Low-poly look with a cautious/darkish atmosphere

2. Moodboard



3. Dramatic Tools

3.1. Theme

- Futuristic origin, but due to visiting different realities, the settings & themes are based on the respective realities (e.g., Middle Ages, Sci-Fi scenario, etc.)
- Careful and detailed designed scenes and sound make it easier to get started with the story and game mechanics and to empathize with the situation the character is in
- Exploration of "What if"-scenarios, that differ heavily from our world (e.g., a world destroyed by meteorites like in the mood board above)
- Venture from one reality to the next and meet alternate versions of the main character
- Catching a glimpse of the endless possibilities that lie within alternative realities
- Solve puzzles and explore the world with caution
- Discover the secrets and stories of different realities throughout the journey of the character

3.2. Mood

- Lightweight and linear story that focuses on interesting puzzles with an innovative gameplay mechanic
- The main character is caught in realities that differ from his own in many ways, but with careful planning and keeping a cool head the player can adapt and overcome these challenges
- Rather small, atmospheric, and calm (e.g., only a few NPCs) levels, that tell the story through the environment
- Minimal use of light leads to an adventurous atmosphere that invites discovery and creative thinking for solving puzzles
- Subtle and mysterious sound puts the player in a ruminative mood and gives appropriate auditory feedback on solving the puzzle

3.3. Form & Style

Form	Style
Interactions between the main character and alternative reality versions of himself	Explain the AR's point of view on the world and how their reality came to be in its current state
Artifacts (e.g., letters, objects of interest, etc.)	Tell the story of the respective worlds
Hostile NPC's talking to each other/displaying their views	Show different factions within a world and what they are trying to achieve
Inner monologues of the character (possibly also including the voices of the ARs within the main characters head	Discussions about how to solve puzzles and how to interact with the world, possibly also giving the player hints for solving the puzzles

3.4. Plot

The player follows the story of a scientist, who invents a device to unite himself with alternative versions of himself from other realities. The device however has a malfunction, and instead of pulling the AVs into his reality the protagonist is pulled through a portal and finds himself in an alternative reality.

Since the device does not work as intended, he now must find a way to fix the device. To do that he must travel from one reality to another and find clues on how to achieve that.

However, his motives will change as he begins to realize that all the realities have already been visited by another version of himself who took vital parts from them and turned them into dystopias (for example one of the realities was destroyed because the atmosphere dissolved, which happened because somehow parts of it "were stolen"), and soon, he finds himself chasing an evil version of himself and trying to help his AV's fixing their realities while he tries to find his way back home.

The protagonist will visit different realities on his journey and learn of their story to achieve his goals. During his travels, he will grow a closer bond with his AVs as he learns how they view their surroundings and what they have gone through. As mentioned earlier, the history of the world will mainly be told through the environment and by small objects that can be collected as well as hearing the thoughts of the AVs in his head who will from time to time talk with each other. To keep the focus on these relationships as well as the puzzling, the worlds will be mostly abandoned by other friendly NPCs, aside from enemies.

3.5. Pacing

The pacing of the overall game is meant to be quite slow. This should give the player enough time to experience the different realities. Therefore, the story follows the same structure as the gameplay, meaning that it proceeds in small steps with each level. Due to the visiting of the different realities, each reality can be viewed as an "act" within the storytelling, consisting of a small introduction to each reality, finding an issue to solve and a final in which it becomes clearer why the reality turned into its current state.

At the beginning of the game, the levels will be small, focusing on teaching the game's core mechanics. Though they will eventually become larger throughout the game, they will remain restrictive and not come anywhere close to an open world. Instead, the puzzles will become much more complicated, as the player can access more characters and through that use more abilities.

When only focusing on playing through the main story the game will continue linear, however, there are also some backtracking elements within the game, as even the levels which happen in the early stages of the game can be revisited later to collect additional rewards by using abilities that the player unlocked after first playing them.

3.6. Visual Experience

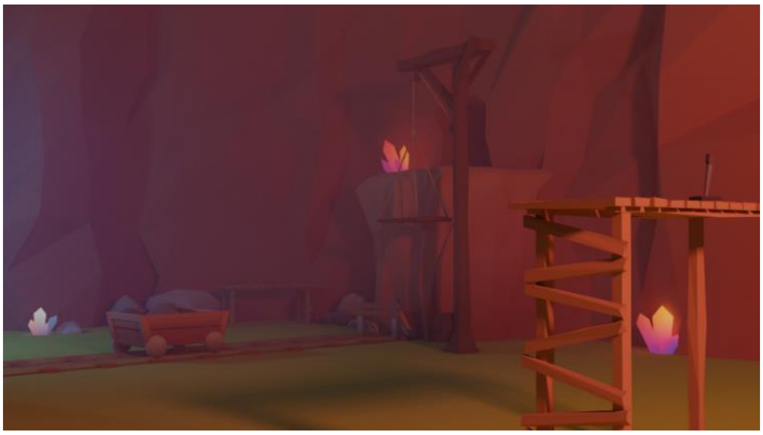


Figure 1: Screenshot for mood of a level

Relating to the mood boards at the very beginning of the document, the game will be created in a low-poly art style. This simplistic look makes it possible to create an immersive mood that still delivers a playful game character. With our game, we are not trying to achieve a physically completely correct depiction of how multiverses work and how to travel between them but instead, give the player a distinct motivation to think about these theories while still allowing them to use their imagination. The usage of this art style as well as an amount of detail that focuses on fewer objects will give us the chance to do that.

While the different realities might also differ heavily from each other, the overall mood will stay calm, focusing on a rather small perspective instead of showing the whole world. With that, again, we want to use the player's imagination, meaning that we give them a small piece of the world which should be kind of a representation of the world in general and then allow them to make up the rest of it in their minds.

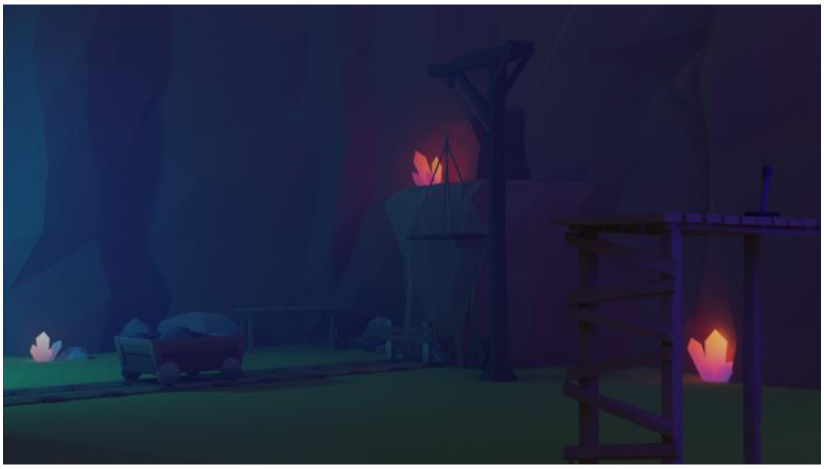


Figure 2: Example of darker lighting with highlights through shining crystals

Another important factor is the usage of light within our levels. By using a dark lighting, we want to make the player feel a little bit uneasy which leads them to act cautiously while also possibly using it to offer more cover when it comes to avoiding patrolling enemies. This also helps us to set harsh contrasts within our levels and highlight important places and objects. An example of how different the levels could look if we put more emphasis on the dark lighting can be seen in the second screenshot. After all, the mood heavily depends on the individual level and what the player is supposed to achieve and experience there, so like making the levels darker, we can also make some levels explicitly brighter to give the player an easier feeling. This could be especially useful in levels at the beginning of the game or when unlocking new abilities to allow for a safer learning environment.

3.7. Audio Experience

Besides the visual components in the game, the different audio elements, which the player will experience during his journey, are playing a huge part when it comes to the mood and the overall experience of the game.

Because the game is based on solving puzzles, discovering secrets, and exploring different realities, it is important to create an interesting audio environment, which supports the "exploring" mood, but also gives the other game elements enough space, so the player can enjoy long gaming sessions without losing focus. Therefore, the music is atmospheric and calm but also contains interesting and exciting parts, based on different events and parts of the story. While there are story-based music changes, to accentuate the story of the game, different audio effects and music pieces are used to improve the gaming experience regarding the core mechanic of the game, the ability to split the reality. Because multiple splits in reality will increase the instability of time, which can cause all realities to collapse, the instability will also affect the music of the game. Therefore, the more instability is caused by the player, the more distortion and noise will be added to the music of the game. Furthermore, different audio effects and sounds are used to increase the effects of different actions especially regarding solving different puzzles. This will also help to emphasize the actions of the player regarding the gameplay.

4. Gameplay

4.1. General Gameplay Flow

The player will loop between the world map, which functions as a hub, and the respective levels throughout the game.

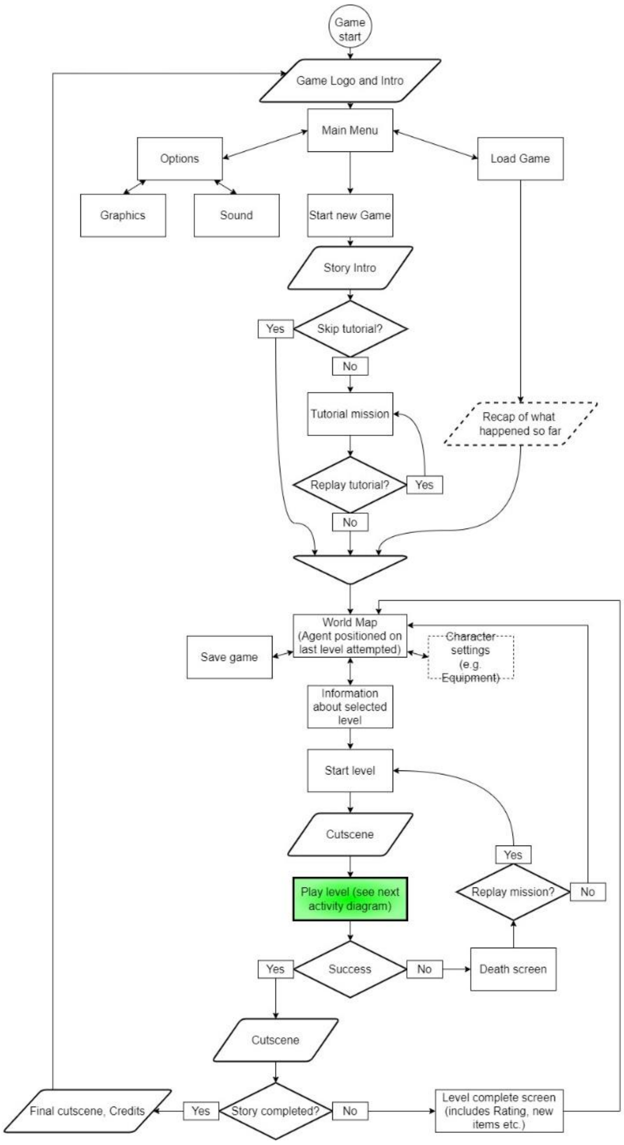


Figure 3: Game Activity Diagram

The world map aims to give an overview to the player and provide information on their current progress. It also allows you to go back to already completed levels and play them again. This is possible because the players can reach new areas of these levels once they unlock new characters and their abilities, allowing them to improve their scores in these levels, which depends on the collection of extra collectibles.

Due to the story taking place in different realities, the world map should be split into different areas but still part of one large map (like in New Super Mario Bros). Alternatively, the world map could also be split into smaller reality maps through which the player can navigate for example by arrow buttons on the side of the screen.

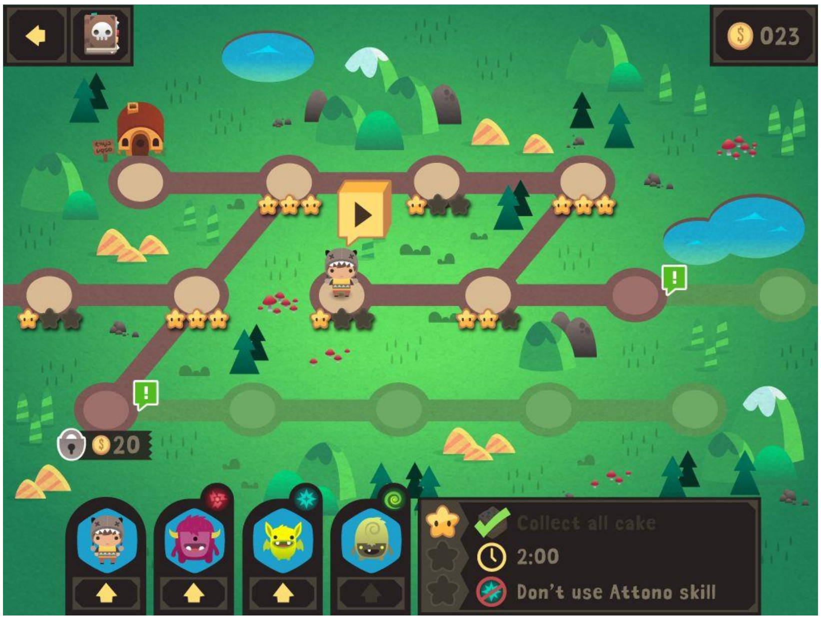


Figure 4: Possible World Map with information on levels (Source: https://i.pinimg.com/originals/4f/19/7c/4f197c291c165a3430ee7b54a64f37f9.jpg)

Selecting a level in the world map will provide necessary information about the level like the current highest score, how many collectibles have already been collected, the fastest completion time, etc.

As shown in the Game Activity Diagram, before entering the gameplay loop the players always start in the main menu which allows adjusting different settings as well as choosing between starting a new game or loading existing ones.

4.2. Level Gameplay Flow

Once entering a level and reaching the interactive part of it (assuming they might be shown a cutscene first) the overall level gameplay loop will consist of navigating through the environment, identifying the puzzles, solving them, and then proceeding further through the level, eventually repeating the process until the end of the level is reached.

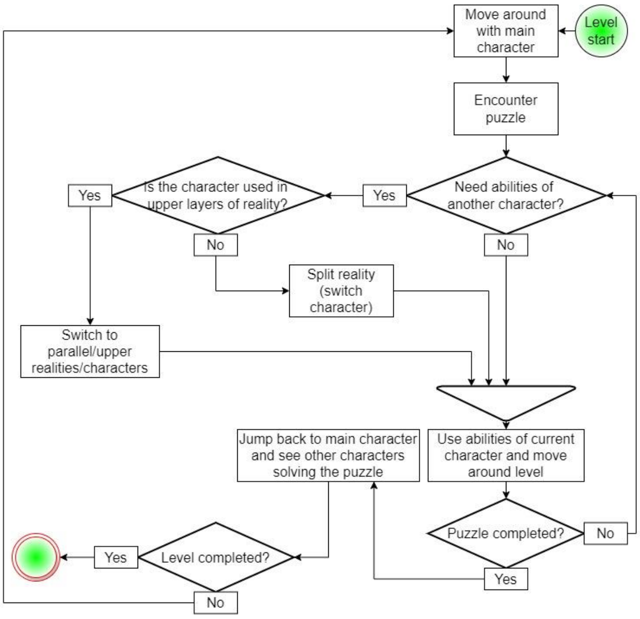

Level start
Move around with main character
Encounter puzzle
Need abilities of another character?
Yes
No
Is the character used in upper layers of reality?
Yes
No
Split reality (switch character)
Switch to parallel/upper realities/characters
Use abilities of current character and move around level
Jump back to main character and see other characters solving the puzzle
Puzzle completed?
No
Yes
Level completed?
Yes
No

Figure 5: Level Activity Diagram

This basic gameplay becomes more complex as the players unlock more characters with new abilities and combine them to solve the levels while also increasing their skills with the different characters to act efficiently and achieve better scores.

4.2.1. Core Gameplay Components

The heart of our game is the puzzles. To solve them the players must use our core mechanic of splitting reality and combine it with their skills of planning. In addition, the difficulty is increased through stealth-game elements, meaning that they must move with caution to avoid being seen by enemies.

The list below should provide a core understanding of the core gameplay components while most of them will be discussed in more detail in the "Mechanics" chapter:

- Different characters

Throughout the game, the player will unlock more and more different characters. These characters are versions of the same person but from different realities on the story layer. As the main character travels through their respective realities, they will join him in hope of restoring their realities to their former state (before the evil version came and destroyed them). However, they are bound to the main character which means that through his device the main character basically unites himself with his AVs and then "spawns" them when needed within a puzzle.

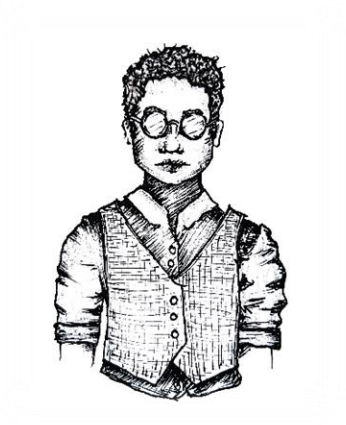


Figure 6: First sketch of the main character

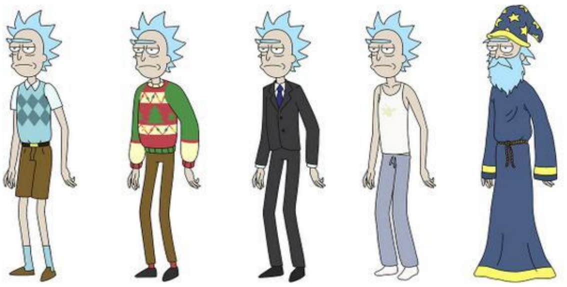


Figure 7: Alternative versions of one character (Source: https://pbs.twimg.com/media/Bk4buQ1CQAAdWRU.jpg)

When looking at the gameplay layer, the characters represent different abilities as well as variants of these abilities, which then can be used within the puzzles. For example, one of the characters might have the ability to jump extraordinarily high, which is then used within the game in 3 different variants, just for jumping on high platforms, jumping across larger distances as well as doing a wall jump to get even further. In contrast to that, another character might be especially strong which is then be used to either push heavy objects, carry, or throw them.

- Splitting/Merging realities

This is the most important gameplay component and probably the most unique aspect of our game.

As shown in "Figure 5: Level Activity Diagram" the player must create different layers of realities to solve the puzzles. After analyzing the puzzle and finding out what abilities are needed, they must decide in which order the different characters have to be used to solve it. Splitting realities allows for 2 different ways of doing so. The first one is to split to another character first, perform actions, merge back to the source of the split, and then split into another character and repeat it. This will lead to the parallel execution of the recorded actions (a video with a visualization of this can be found here: https://youtu.be/G_y6u8OjylO).

The second way is to split from the original character, performing actions and then splitting again to another character, before merging back to other layers.
Both options can and even must be combined throughout the game to solve the puzzles.

- Planning

This one is also a result of the combination of the 2 previous gameplay components. Like his alternative versions, the main character is also able to access a certain special ability that is used in different variants. In contrast to the other abilities, his is a little more abstract, namely “Planning”.

That decision intends to enable a gameplay flow where the player first moves through the level (or at least to all parts of it that are currently accessible without splitting and using other characters' abilities) and spends time analyzing the puzzle. Within that planning phase, the player should gather important information like where certain other abilities could be used, the position of important quest objects or collectibles as well as enemies within the level and their walking paths. To help the player do that the main character has variations of that extraordinary planning skill.

So far, there are two variations which are:

- Switching to a “Drone view”, which allows a top-down view on the close surroundings of the character (with a small radius of about 5 to 10 meters)
- Switching to a “Ghost” view allows to practice splitting and plan which realities to use at which moment and in which order while it does not allow interactions with any of the interactable objects and neither the use of character abilities

There could also be a third one that would align this character a little bit more with the others, but it should be well thought through to avoid giving too much help for solving the puzzles.

The table below was created in the early stages of developing our concept, but it shows well how the individual characters differ from one another.

Character	Basic Skill	Variation 1	Variation 2	Variation 3	Scenario/Homeworld ("What if"-Scenario)
Maino	Planning	Drone view (small eagle view)	Ghost view (plan which realities to use and when to split, no interactions possible)	n.a.	Like earth
Strongo	Increased strength	Carry heavy stuff	Push heavy stuff	Throw heavy stuff	Stone Age (Strong from fighting mammoths)
Jumpo	Increased jump height	Jump higher	Jump further	Jump onto things with more pressure	Meteors have destroyed earth (mainly canyons)
Divo	Diving	Dive in water	Go through glibber without suffocating	Can go through spaces without air	Post-climate change (the planet is mostly flooded)
Climbo	Climbing	Climb (certain) walls	Hold onto things (e.g. a platform that moves, a box that gets thrown later)	Grappling element	Jungle (Tarzan like)

Table 1: Characters and Scenarios

- Executing actions and keeping reality stable

Once the players have analyzed the puzzles, they must solve them. While they can use the different characters in any order they like and deem right, they must pay attention to how long they use each character (besides the main character). Using an alternative version decreases the time stability level for the time it is used or exists as a ghost, which results in worsening conditions (e.g., faster-moving platforms) and ultimately in failing the game and having to restart.

There will be collectibles that help improve the stability throughout the levels but after all, the players must find a balance between giving themselves enough time to execute their actions while keeping the reality possibly stable.

4.3. Control of the game flow with the help of the UI-Elements

The font of the game is Exo-Bold. It is a sans-serif font that is well suited for reading on screens. Textual information should mostly be presented in white, as many scenes of the game take place in darker environments.

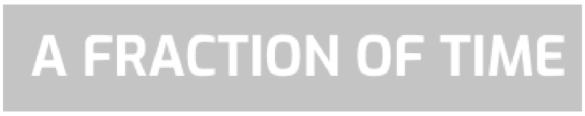


Figure 8: Font Exo-Bold

The UI elements have mostly rounded corners and a light color tone. The game itself is kept in a dark mood, but the UI design of the low-poly style, as well as the design of the main character, should underline that this is not a horror game. Thus, the design aims to create a more relaxed mood. The gloomy lighting sets a certain mood in the game, but the tension should not override the main motive of the game, which is solving puzzles with the help of the time-splitting mechanics. The linear decrease of the alpha value towards the edges of most UI elements is meant to emulate the game's logo (a clock that dissolves).

In the main menu as well as in the pause menu, a simple interface shall prevail, which includes the font of the logos and reflects the low-poly design of the game.

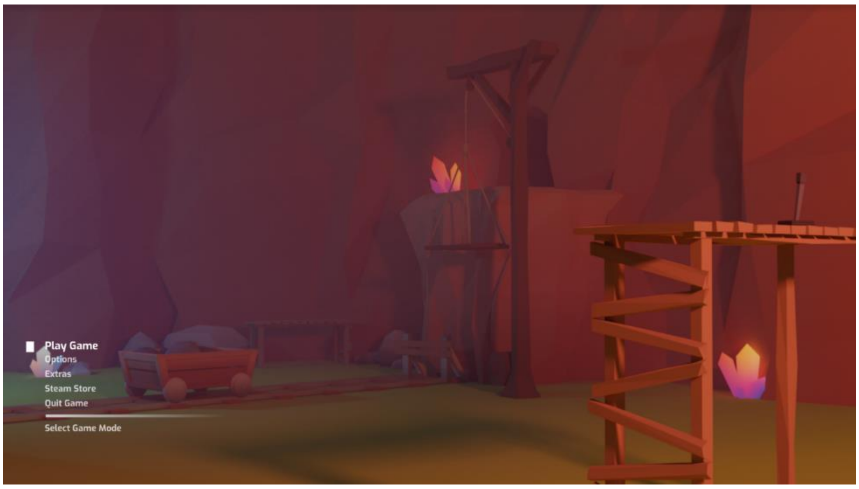


Figure 9: Main Menu of the game

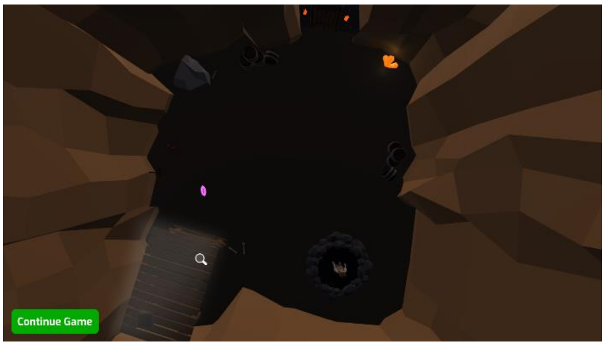
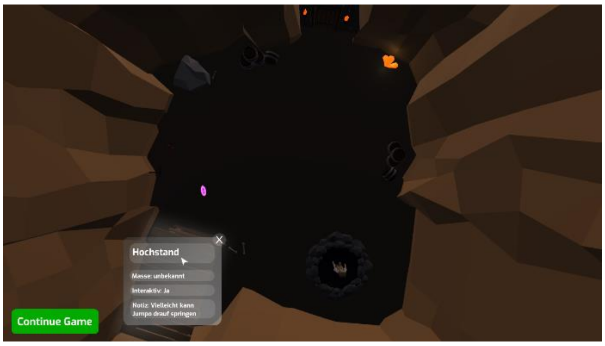


Figure 11: Hovering over an interactive Object



Masse: unbekannt

Interaktiv: Ja

Notiz: Vielleicht kann Jumpo drauf springen

Figure 10: Information of an interactive Object

In the planning tool of the main character, it is possible to hover over various objects using a magnifying glass as a cursor. If an object contains information that could be important for the player, the object lights up and an information box appears. Information about the name of the object, the mass, whether there are interactions with the object, and a note containing a hint is displayed. This tool is therefore important for the player, especially at the beginning of the level, to plan his approach and to find starting points on how to approach the level. ([Figma Prototype]())

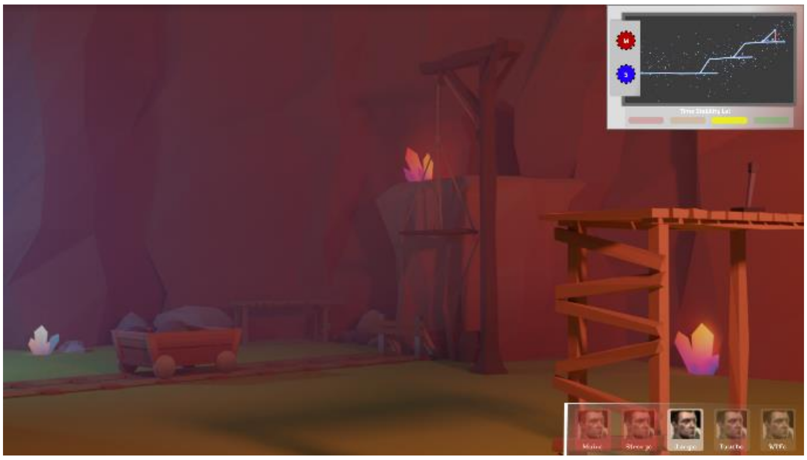


Figure 12: Runtime HUD

The game HUD should be as unobtrusive as possible but still informative for the player. The Timekeeper module in the top right corner should display information on the different time strands and show the time stability level with the help of 4 levels. In addition, the time in minutes and seconds is shown how long the game has been in progress from the main character's point of view. The design is supposed to be reminiscent of a small "time-control device" that the main character, as a scientist, may have constructed himself to find his way in time. At the bottom right there is a small character overview. It should show which character is currently being used. In addition, it could show in a later development status whether another character is currently blocked, has died, or is otherwise unusable or available. In the runtime HUD, information can also be displayed when the player reaches a checkpoint with the current character.

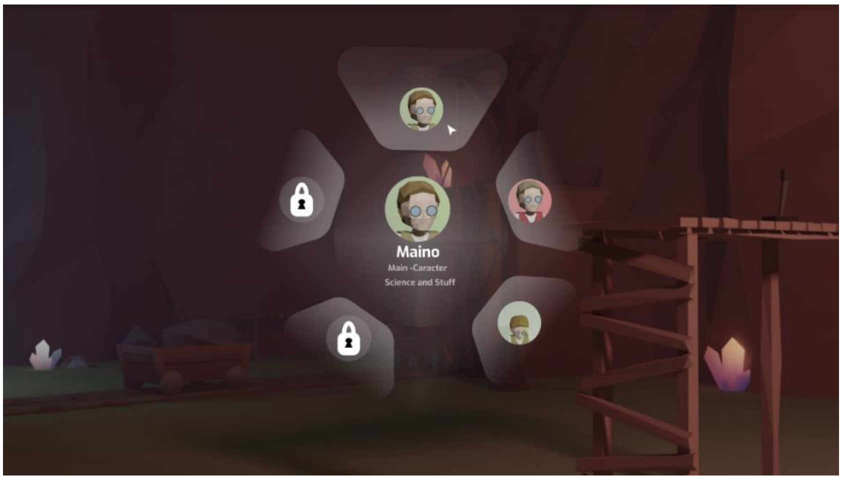


Figure 13: Circle Menu to select the next Character

The menu used for splitting and merging is made from a circle, with each character having his own section. It can currently be accessed from the running game by pressing the "G" key. This is still the state of development. For an easier flow of the game, the "Shift" key could also be imagined. When the menu is activated, the mouse cursor is displayed (which is otherwise locked and hidden during the game) and you can simply hover over the characters to be selected, for which information is then displayed in the center of the circular menu. Unlocked characters are visible with their picture. Characters that have not yet been unlocked have a lock as their symbol. The character from which the menu was called up is greyed out and cannot be clicked on, all others can be activated by clicking on them.

4.4. Puzzles and Challenges

As our game is mainly considered to be a puzzle game, they understandably play a vital role within our concept. While we want to create exciting puzzles that allow for different solutions and ways to experiment, we also have to keep in mind how we challenge the players, how difficult said challenges are, and what kind of players we want to attract with our game. So, when it comes to designing the puzzles in our game, we must focus on the kind of problem-solving that users are supposed to perform while playing. As mentioned before, players must analyze their environment, identify the parts of the puzzles and their relation to one another, and decide which abilities of the different characters are suited for them.

These gameplay parts resemble different skills of problem-solving and can be viewed as increasingly difficult steps of our gameplay. Analyzing the environment requires pattern recognition, which will be even more amplified as the surroundings change throughout the different chapters of the story (mainly when visiting other alternative realities). Following that, combining the gathered information, and understanding in which order the puzzle components can be completed means abstracting the problem. These two steps are the core of solving the puzzle, however, we also want our users to be creative with the options that they are given. Puzzles, especially in the late game should be designed in a way that allows different solutions and through that enable the players to use their skill of lateral thinking, which describes the creative interpretation of given rules.

As a result, it can be said that puzzles though they should be easier at the beginning and become more difficult should try to engage the users on these three layers. While the first two layers will be more important in the early stages, there should be a shift towards the lateral thinking component later. If used correctly this will help us to attract a wider range of players, because we give players who are new to the genre an easier start and the possibility to learn step by step while also catering to more advanced players with the necessary freedom to experiment.

4.5. Player progress and rewards

After we have discussed the challenges that our users have to overcome, we now want to have a look at how the users progress throughout the game and how they are rewarded.

As our game follows a linear story structure, the progress does so as well. Based on that it can be said that the main progress happens through unlocking new characters as well as abilities. Through that, the user will feel more empowered as they solve more complex puzzles. They can also go back to already completed levels to improve their scores and reach other parts of these levels. This feeling of progress and becoming stronger could also be diversified by adding items that can be found and used like abilities (e.g., a grappling hook). So far these kinds of collectibles are not part of the concept which is why they will not be discussed more in-depth for now.

The main reward for users will be the high score that they receive at the end of each level. The score will be based on different parameters like the level completion time, the stability levels, optional collectibles (like the star coins in Super Mario Bros.), or other additional targets. However, this score is just a collective resemblance of smaller rewards that happen throughout the level. Likewise mentioned collectibles serve as smaller rewards as they might provide more information on the world or trigger additional events (like a side quest). After all, our game targets users who find joy in problem-solving and through that reward them with positive emotions when doing the puzzles.

5. Worldbuilding

5.1. World Rules

According to the linear main story and the semi-linear level structure, it is possible to define the most important rules for this world:

- The player CAN NOT change the outcome of the story (in general)

BUT

- The player CAN experience the game to individual extends and solve puzzles in different ways

When defining the world rules, it is especially important to us how the users interact with the world. This relates to the circumstance that we record the player's actions and must reconstruct them afterward. To do that we have to find ways to save different kinds of actions (e.g., passive actions like walking vs. more active interactions like using a lever). This means that we must characterize the player actions and decide which information is needed to reconstruct them. Table 2 should illustrate some examples of different kinds of actions, how the player character, as well as the environment, reacts to them, and the key information that we would need to store.

Categorizing the different actions is a core task that we already dealt with while creating the prototype (it will be discussed again in the mechanic section) and will also deal with further as we implement new features, abilities, and everything that is not static within our levels. As we jump back and forth between timestamps, we will have to store all the information not only in a way that allows us to reconstruct everything but also efficiently to avoid huge amounts of data as everything must be reconstructed during runtime without drops in the framerate.

Action	Player Reaction	Environment Reaction	Key Information
Walk	Moving at a certain speed, towards chosen direction	Sounds and speed based on the ground they are walking on	Timestamp, Position, and Rotation
Jump	Jumping upward/forward/ backward/sideways, height and distance based on character, possibly taking fall damage	Sounds based on starting and landing ground, volume based on jumping height and weight of character	Timestamp, Position, and Rotation
Split	Splitting to any other character, If the chosen character does not yet exist in another timeline it is spawned where they currently stand, otherwise, jump back to those characters last saved position	Freeze while in the character selection menu. If an already existing character is chosen set the world to the level time of the last recorded action	Timestamp, Split origin (splitting character), split target (chosen character), and player position
Use lever	none	Activate/Deactivate objects linked to lever, clicking sound and other sounds based on linked objects	Timestamp, lever activation status (on/off)
Use pressure plate	none	Activate/Deactivate objects linked to the pressure plate, clicking sound and other sounds based on linked objects	Timestamp, Player collision (enter or exit)
Push heavy object	Character and object are linked (player movement restricted to the movement of the object (e.g., a minecart can only be pushed along the tracks)	Sounds based on the object, object position can be manipulated by player input	Timestamps begin and end push, object position at start and end

Table 2: Player actions

5.2. World Content

5.2.1. Item Placement

The number of collectible items within our world is limited to a small amount for now. We can narrow it down to two types of items:

- Collectibles with additional information on the game world
- Trophies (previously compared to the Star Coins in Super Mario Bros.)

Both item types are more as extra motivation for players to stray a little more within the levels and give them a reason to look out for other puzzles aside from the main one leading to their level target.

In addition to these types, we also considered Pickups or Powerups (so items that give a temporary or situational boost) as part of the concept. These items could help players to easier achieve their goals and at the same time serve to lead players through the level. For example, a well-placed pickup that lowers the time stability level could help the player to get through a certain puzzle element as they can use their destabilizing characters for a long time without reaching the time stability cap and die. At the same time, the placement of this item could help players to connect the dots, as they might automatically try to include these items within their logical construct. However, placing such items is also a question of balancing the difficulty of our levels. Because of that, we must cautiously try and test to what extent they are helpful without taking away the challenge.

5.2.2. Scripted Sequences

Scripted sequences are a great way to tell stories within a game. This is especially true for our game. As we follow a linear story whose outcome cannot be changed by the players, we must find ways of immersing them into the story without them influencing it. By using scripted sequences, we can detach the gameplay from the story. This means that we can tell the core story moments through these sequences like a movie while still giving the players relative freedom within the levels.

With that approach in mind, it seems likely that we will often follow a structure where the levels start with a scripted sequence that tells a part of the story and includes mission targets and other vital information. Following that the level will consist of gameplay parts which might occasionally be interrupted by more scripted sequences to tell further parts of the story. After concluding the final gameplay part of a mission, it will then end with another final sequence. This loop will most likely be repeated in the different levels, leading to the story being told parallel to the player progressing through the game.

In addition to these sequences, it also seemed interesting to us to have the alternative versions of the main character converse with one another. This will happen while walking through the level. As the player reaches certain points of interest, they will hear the voices of the different characters commenting on their surroundings. However, all the characters can talk at any given point regardless of which character is currently active and controlled by the player, displaying the conversation as an inner monologue instead of the characters physically talking to each other.

5.2.3. AI/Actor Placement

In general, it can be said that our levels are scarcely populated with actors other than the ones controlled and instantiated by the player. This relates to the general mood of our game where the focus is on telling the story through surroundings and because we mostly visit apocalyptic scenarios in which most of the population is probably already wiped out.

However, we will have to place some actors at some point. For example, before the player unlocks a new alternative version, these AVs are independent of them. When entering a new reality, the player might first have to find their AV and do a task for them in which they have to free the last family member from a group of bandits. In this example we can already find 3 different types of actors:

- The AV, who is an independent actor until the player accepts the quest at which point, they both unite
- The family member, to whom the player can talk and maybe gather some additional information and
- The bandits are hostile NPCs that the player has to avoid or kill.

This example illustrates different ways of using actors within the game. However, the kinds of actors in our game as well as their numbers were not discussed in-depth yet. The decision of which actors to use and when can therefore only be decided after larger portions of the story are written while also considering how well they can be integrated into our current concept.

5.2.4. Icons and Symbols

These components and their placement in the world are vital to the difficulty level as well as the overview of our game and therefore must be used carefully. Using many icons and symbols might improve the overview but also make the game too easy as players only must look out for them and just do what they instruct. To avoid that we have two different kinds of icons/symbols.

Firstly, we have symbols that are important to understand the game, for example, tooltips that explain the fundamental controls or mechanics. Due to their importance, most of them are visible regardless of which character is currently selected (excluding the ones that are character-specific, e.g., explaining a character ability for the first time. We already used symbols like that in our prototype and gathered positive feedback on them and most testers also expressed that they would like even more of them.

Secondly, we have symbols that help structure levels and hints on how to solve puzzles. This additional information can only be accessed through the special ability of the main character. As mentioned in previous chapters, the main character has access to 2 different perspectives: The first one is a different view, which highlights the objects that are specific to certain character abilities in different colors (e.g., objects that can be pushed by the strong character are highlighted in red, while the ones for the climbing specialist are highlighted in green). The second view is a top-down view that shows the close surroundings of the main character from above and by that enabling a "strategic perspective". In this view, the player might be able to mark points of interest so that a symbol at the chosen point is shown to all characters.

By dividing the important information into these different layers, we want to split the sheer amount of information into smaller bites while also giving the main character more gameplay relevance than just walking around and instantiating other characters, mainly giving the task of planning to that character.

6. Mechanics

6.1. Core mechanics

The player controls one of 5 characters at a time with a classic third-person controller.

- The player's default abilities are walking, jumping, crouching, sprinting, and interacting with certain items.
- The abilities may get enhanced or there may be new abilities added depending on the active character.
- The player can split his current reality to create a new one by selecting one of 4 characters in the split menu
- The current time doesn't correspond with the actual time passed since starting a mission, and is dependent on the time passed while the "root" character was active.
- Every character can split the reality or merge back into an already existing one.

6.2. Splitting and merging realities

Realities can be split up by selecting one of 4 characters in the split menu. The splitting mechanic follows a certain ruleset:

- Realities work like a node tree: one reality can have multiple child realities.
- When splitting a reality, the new reality originates at the exact time, space, and game state that was currently active.
- The new reality has to be created using a different character from the one currently in use.
- The player can continue exploring the level/solving a puzzle with his new character until the reality gets too unstable or he decides to "merge" back manually.
- After a merge, the game state from the exact moment when the split was performed gets restored, as well as the time and

position. The player now has the set of abilities of his old character.

- The game time runs again and all the actions performed in the merged child reality will be performed, in synchronization with the time the player took to perform them, by an AI, using the character that got selected when split up.
- At no point in time, 2 characters with the same set of abilities are allowed to coexist.
- Realities can be split up multiple times
- The root reality can't be merged and won't collapse ever
- A level can only be finished by the main character in the main reality

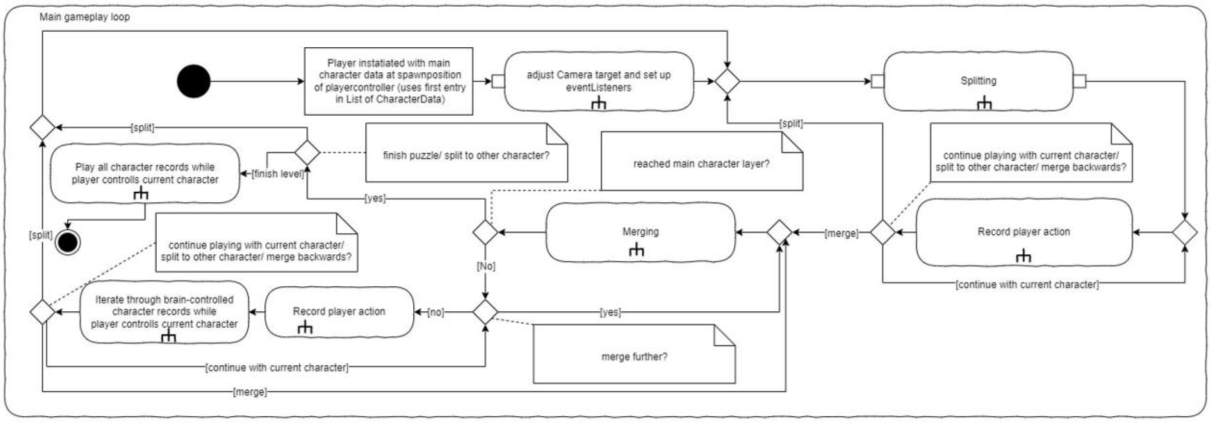

Main gameplay loop
Player instatiated with main character data at spawnposition of playercontroller (uses first entry in List of CharacterData)
adjust Camera target and set up eventListeners
Splitting
[split]
finish puzzle/ split to other character?
reached main character layer?
continue playing with current character/ split to other character/ merge backwards?
Play all character records while player controlls current character
[finish level]
[yes]
[split]
continue playing with current character/ split to other character/ merge backwards?
Merging
[merge]
Record player action
[No]
[continue with current character]
Iterate through brain-controlled character records while player controlls current character
Record player action
[no]
[yes]
merge further?
[continue with current character]
[merge]

Figure 14: Activity Diagram main game loop

6.3. Time

The time used by the splitting logic doesn't correspond with the actual time since the level is started. The time usually matches the time spent in the main reality. When the main reality is split up, this time runs in an alternate reality and when merged back, gets reset to the timestamp when the split was performed. The ghosts of the alternate characters only stay visible until the time spent in this reality is passed.

6.4. Characters

There are a total of 5 playable characters, with one being the main character and 4 of them having enhanced or special abilities. These abilities have up to 3 variations depending on the current level. All these 4 characters have all of the main characters' core abilities like jumping, crouching, etc. But similar to the main character they all have an individual special ability.

- Main character: can walk, jump, crouch, and interact with certain objects
- Jump character: has a higher, longer, or more impactful jump
- Power character: can carry, push or throw heavy objects
- Dive character: can dive, walk through slime, and doesn't need air to breathe.
- Climb character: can climb certain walls, stick to some objects or use a grappling device to pull himself to objects/pull objects towards him.

6.5. Implementation

- Timelines and Graph Controller

One of the most important aspects of the alternate timeline mechanic is the node-tree-like timeline graph. Every timeline stores information about the exact timestamp it got created at, a reference to the character using this timeline, and references to every child's timeline. Child timelines are created by splitting the current reality.

The graph controller class stores a reference to the root timeline (the original timeline used by the main character). After every merge, the graph controller loops over the root time timeline and its children, the children's children, and so on. A reference for every timeline that doesn't match the current timeline is stored in a list. Every frame the graph controller loops over this list of timelines and checks for interactions stored in the referenced character of each timeline.

- Player and Shadow

To further explain how splitting and merging are implemented, we have to declare the differences between a playable character and a shadow.

There are 4 different versions of characters, but there are two existing versions of each character, the player and shadow variant. Playable characters can be controlled by the player and have different abilities. Each form of interaction by the player with the character is a different kind of ability and causes a different type of interaction. Walking, jumping, picking up items, and more are considered abilities. Shadows don't have any of these abilities, but they can replay these interactions. After a split, the selected character must be instantiated by using his playable characters prefab, containing all his abilities. The old character prefab that was used priorly must be deleted and is replaced with the shadow variant. After a merge, the character that just got controlled must be deleted and replaced with his shadow variant, but this time the graph controller is alerted and is adding timelines as described above. The selected character that is being

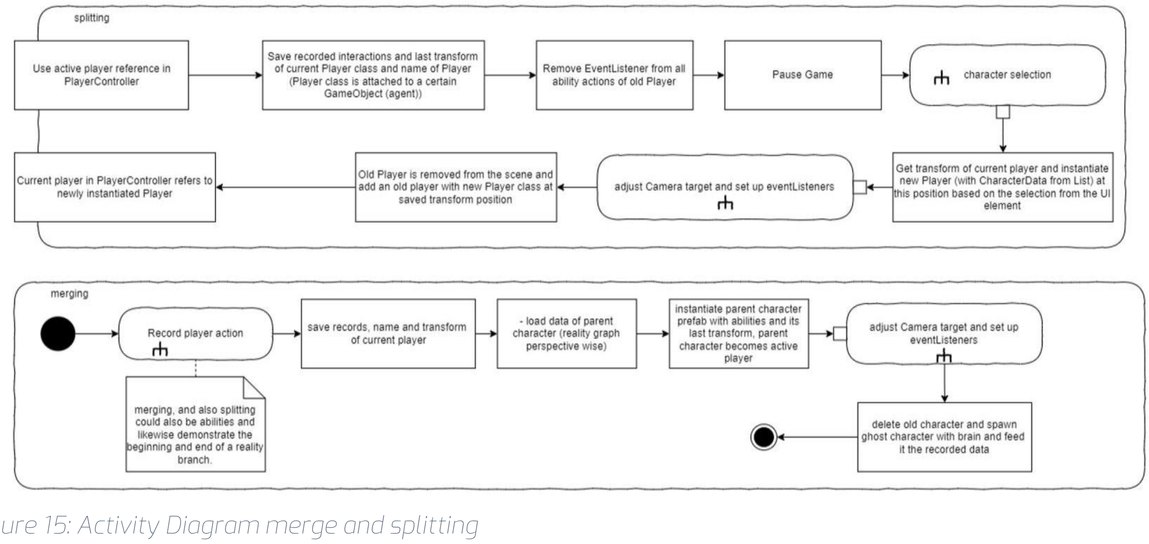

splitting
Use active player reference in PlayerController
Save recorded interactions and last transform of current Player class and name of Player (Player class is attached to a certain GameObject (agent))
Remove EventListener from all ability actions of old Player
Pause Game
character selection
Get transform of current player and instantiate new Player (with CharacterData from List) at this position based on the selection from the UI element
adjust Camera target and set up eventListeners
Old Player is removed from the scene and add an old player with new Player class at saved transform position
Current player in PlayerController refers to newly instantiated Player
merging
Record player action
save records, name and transform of current player
- load data of parent character (reality graph perspective wise)
instantiate parent character prefab with abilities and its last transform, parent character becomes active player
adjust Camera target and set up eventListeners
delete old character and spawn ghost character with brain and feed it the recorded data
merging, and also splitting could also be abilities and likewise demonstrate the beginning and end of a reality branch.

Figure 15: Activity Diagram merge and splitting

merged to has to be instantiated by using his playable prefab alongside all his abilities.

- Recording and Reconstructing

In the beginning, there were two considerations, how the interactions of the user with the player in the world could be recognized and recorded. These considerations have been tied to the system of the engine. It was important to know whether the engine was deterministic or not. In a reproduction process, the engine must be deterministic, so that what has been recorded can be represented in the same way; Unity is not deterministic, so that user inputs could not be stored, but rather individually abstracted events or interactions. These each gets their own timestamp, which comes from a time management system that had to be additionally created. Thus, points in time and their respective properties could be defined for interactions. When replaying these interactions, the graph controller class goes through the list of interactions and executes them.

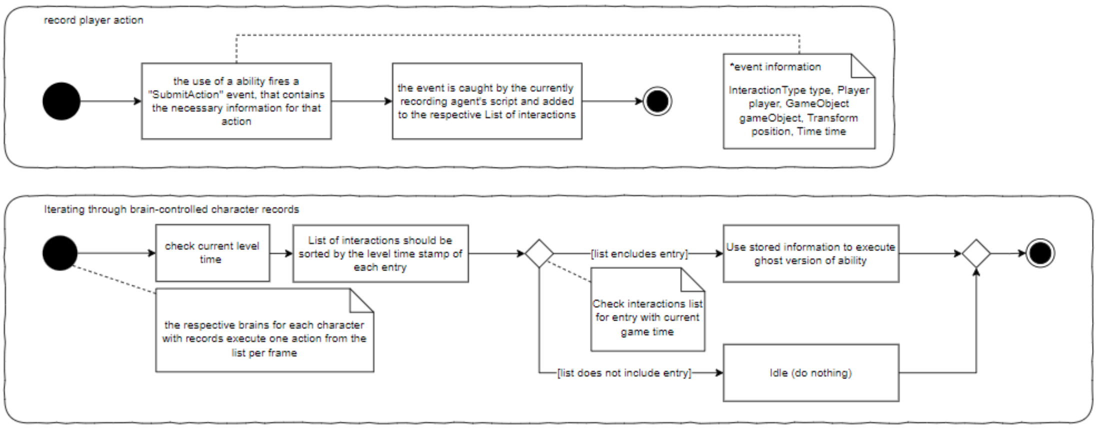

record player action
the use of a ability fires a "SubmitAction" event, that contains the necessary information for that action
the event is caught by the currently recording agent's script and added to the respective List of interactions
*event information
InteractionType type, Player player, GameObject gameObject, Transform position, Time time
Iterating through brain-controlled character records
check current level time
List of interactions should be sorted by the level time stamp of each entry
the respective brains for each character with records execute one action from the list per frame
[list encludes entry]
Check interactions list for entry with current game time
Use stored information to execute ghost version of ability
[list does not include entry]
Idle (do nothing)

Figure 16: Activity Diagram recording and reconstructing

- Conclusions from our prototype's mechanic implementation

After implementing the mechanics as explained above, we came to a few conclusions. A timeline graph and the time controller seemed to be a good way to check for alternate timelines and the corresponding interactions. However, using Unity's indeterministic Physics Engine led to us using a transform-based replay system, which came with some disadvantages: For example, the animations had to be replayed by estimating the input between two interactions, rather than directly using the input. An input-based replay system would also cause a major performance increase, since way fewer data has to be stored and reloaded. There wouldn't be a need to store interactions too, as a deterministic engine would simply make the shadow use an ability by using the stored input and replay the interaction the intended way.

6.6. Artificial Intelligence

For the integration of enemy NPC, it is planned that the navmesh agents are used to control the movements to different target points. In addition, the current rotation can be accessed, and colliders can be integrated as a field of view.

Such a pathfinding system must be tested for determinism. It follows that, if necessary, the movements of the NPC must also be recorded, since the NPC must always move in the same way as the recorded player.

As such, this integration has not yet been discussed and is a consideration for future content in the level design of the game.

6.7. Physics

Physical calculations take place mainly only in the calculation of collisions between the player character and the environment. As well as the calculations of collisions of the ghost characters that are instantiated at the same time as the player character in the scene.

6.8. Object interactions

In our prototype, we have limited ourselves to interactions of objects that perform certain actions on the proximity of a character, e.g., standing on a pressure plate opens a door or lowers a suspension bridge, to contribute to the puzzling character of the game and thereby complete the level. These interactions are therefore still independent of our game time. There are more interactions planned where we also must save the status via gametime e.g., moving a cart or flipping a lever

7. User Tests and Future Development Plans

As part of the "Tag der Medien" at our university, we performed several user tests. To make testing as easy as possible, we created a WebGL-Build of our prototype which could then be accessed via a link and played in the browser. (Link to the WebGL-Prototype: https://nickhaecker.github.io/A-fraction-of-time)

To better structure the received feedback we will discuss it in two different chapters. The first one will tackle the results that we got regarding prototype-specific topics like graphics or controls, whereas the second chapter will deal with the feedback about our general game concept.

7.1. Prototype Feedback

A lot of the feedback that we received was about issues that relate to the quality of our prototype. Since we focused heavily on making the split/merge-mechanic work, some other things like the controls or graphics came a little short.

Many people complained about the controls, which were plagued by a lot of issues and minor bugs. For example, the mouse sensitivity was much higher in the WebGL-version than we anticipated. Furthermore, jumping was quite difficult, and sometimes people had to hit the space-key multiple times to jump. On top of that, the colliders made a lot of trouble and people sometimes got stuck in walls or stairs. This led to the parkour in the last chamber being nearly impossible to finish.

Another often mentioned issue was the lighting in our scene. Many people said that the level was too dark, which made it difficult to see important objects like the pressure plates on which they had to stand. In addition to that, the camera tended to start shaking when getting too close to objects. Still, people liked the overall look of our game and enjoyed the atmosphere with the glowing crystals and the simplistic low-poly style.

Lastly, some of the users also said that the UI, as well as the texts, could have been a little bit larger.

7.2. Concept Feedback

The overall feedback on our concept was positive. All the users enjoyed the mechanic and said that it intrigued them to use it for more complex puzzles and that they would be interested to see it in more diverse surroundings and scenarios.

Furthermore, we also learned that we need more in-depth tutorials for the mechanic. The one we currently had in our game (and which only consisted of tooltips being prompted from time to time) was only explaining the controls and not how to use the mechanic to solve the puzzles. Because of that, we had to explain it to the players ourselves.

Another issue was the current structure of the level as well as the contained puzzles. We got the advice to think more about how to lead the players through our levels, using colors and rearranging the level. Some players also asked for more tools that would make it easier to keep an overview amid the complicated gameplay. Often ideas were uttered that we already had in our concept but not yet in the prototype, like the ghost-view for our main character and the time-graph visualization. This was very good for us as it assured us to implement these features and lessened the fear of making the game too easy.

Concluding, we got the feedback that we aimed for. While there is plenty of stuff that we could have done with more time to increase the quality of our prototype, we still achieved our goal of testing the mechanic and seeing whether it is interesting and understandable to players.

7.3. Future Development Plans

With the feedback that we have gathered, there are a lot of things that we can do in the future. First, we should improve the controls and graphics of our game, as these are short-termed improvements to further increase the quality of our prototype, which we will need to do more accurate testing in the future.

In the long term, we want to include more features based on our concept as well as the received feedback. The following list should give an overview of possible things that we can work on in the future:

- Improving our level design

We want to improve the structure of our levels, fill them with more content (e.g., collectibles & objects of interest) and work on the atmosphere to better immerse the players.

- Implement more characters and abilities

The gameplay core of splitting and merging is based on having different characters to choose from and using their abilities in different ways. Therefore, it is important to work on more characters in the future and a key component when it comes to improving our game.

- Add time stability mechanic as well as a system to measure the player's success

Currently, players can practically give themselves as much time as they want to, as there is nothing that forces them to act fast. The time stability mechanic should counterbalance that as it forces them to spend time with characters efficiently if they want to get higher scores. This goes hand in hand with implementing a high score system to measure how well players performed within the levels. To do that, we will need to define parameters for success within our game and find ways to motivate players to achieve a higher score and try levels again once they finished them the first time.

- Add more tools and improve overview as well as the UI

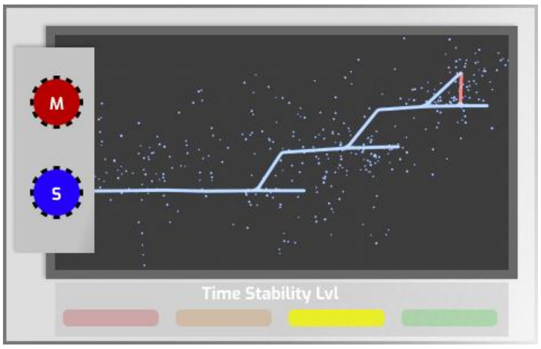


Figure 17: Time Graph visualized

As our gameplay will become more and more complex, it is important to provide players with more tools and a better overview. On the one side, this can be achieved by implementing tools like the ghost view and improving how we lead players through the levels. On the other side, we need to work on our UI and its elements. Mainly we want to visualize the player actions in a graph that shows splits and merges as well as other important player actions and the time at which they were executed.

- Changeover of the system

Currently, the user interacts with the abilities of each character. The resulting interactions are saved with an in-game time stamp so that they can be reused. The interactions are saved in the player class and used from there. In the case of a split or merge, the corresponding interactions are loaded per character and executed by the player class on the corresponding game object.

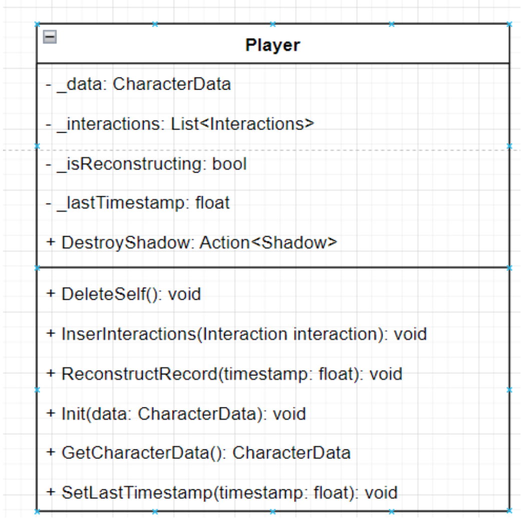

Player
- _data: CharacterData
- _interactions: List<Interactions>
- _isReconstructing: bool
- _lastTimestamp: float
+ DestroyShadow: Action<Shadow>
+ DeleteSelf(): void
+ InserInteractions(Interaction interaction): void
+ ReconstructRecord(timestamp: float): void
+ Init(data: CharacterData): void
+ GetCharacterData(): CharacterData
+ SetLastTimestamp(timestamp: float): void

Figure 18: Player class

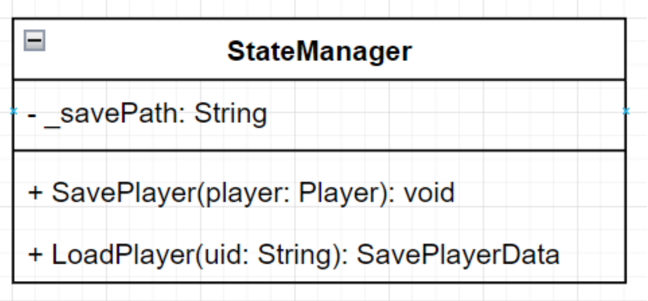

StateManager
- _savePath: String
+ SavePlayer(player: Player): void
+ LoadPlayer(uid: String): SavePlayerData

Figure 20: State Manager, saves Player

After looking at the UI in the previous section (as you can see in Figure 8), we can see that it makes more sense to save the actions that the player completes as a split character on his timeline. This means that everything that happens on the timeline should also be on it. So that the Ghost or Shadow we place in the scene becomes only a dependency of the timeline and no longer the main anchor point. This then makes it easier to resolve impatient play and misunderstandings in the time interaction.

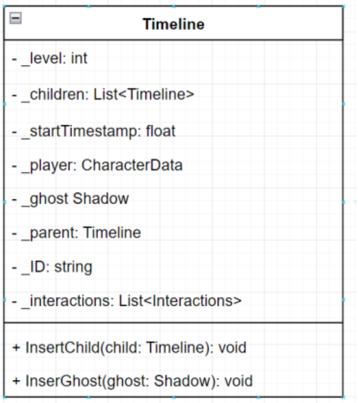

Timeline
- _level: int
- _children: List<Timeline>
- _startTimestamp: float
- _player: CharacterData
- _ghost Shadow
- _parent: Timeline
- _ID: string
- _interactions: List<Interactions>
+ InsertChild(child: Timeline): void
+ InserGhost(ghost: Shadow): void

Figure 19: New Timeline Class

It follows that the current timeline class will be extended with the player's interactions and the system with the focus on timelines now seems to make more sense.

SOURCES

Moodboard

- https://www.google.com/url?sa=i&url=https%3A%2F%2Fdevforum.roblox.com%2Ft%2Fhow-do-i-start-making-realism-in-low-poly%2F662514&psig=AOvVaw0bjWD-JpGdC7x69wptri6p&ust=1637848251301000&source=images&cd=vfe&ved=0CAsQjRxqFwoTCMDx5K-SsfQCFQAAAAAdAAAAABAD
- https://www.google.com/url?sa=i&url=https%3A%2F%2F3dmdb.com%2Fen%2F3d-model%2Ftemple-ruins-and-jungle-low-poly%2F5057680%2F&psig=AOvVaw0bjWD-JpGdC7x69wptri6p&ust=1637848251301000&source=images&cd=vfe&ved=0CAsQjRxqFwoTCMDx5K-SsfQCFQAAAAAdAAAAABAO
- https://i.ytimg.com/vi/Y_rwie8BaRs/maxresdefault.jpg
- https://nintendoeverything.com/wp-content/uploads/spirit-of-the-north.jpg
- https://cdn1.epicgames.com/ue/product/Featured/AdvancedPuzzleConstructor_featured-894x488-a1905667e4df28695d264a7a3dacde3c.png
- https://cdn.cloudflare.steamstatic.com/steam/apps/1163640/ss_312322e8fe0a012d62566e8f7013ddd7a3defcd3.1920x1080.jpg?t=1636630886
- https://dribbble.com/shots/12441464-UFO-Laboratory/attachments/4054403?mode=media
- https://www.pinterest.de/pin/9499849204532344/
- https://store-jp.nintendo.com/dw/image/v2/BFGJ_PRD/on/demandware.static/-/Sites-all-master-catalog/ja_JP/dw86ddbe67/products/D70010000011246/screenShot/599d33f75cf0308cc12fc5d9870bffefd88debd1d42077e7ffe46975ee54a3af.jpg?sw=1368&strip=false

- https://images.everyeye.it/img-screenshot/operation-tango-v1-694625.jpg
- https://i.pinimg.com/originals/ec/a8/5e/eca85ea70bcba16144080ef8fbaa56b4.jpg
- https://cdn.dribbble.com/users/1514670/screenshots/10415933/media/14400d3d467e5b2481bababe83c2d455.jpg?compress=1&resize=400x300
- https://mir-s3-cdn-cf.behance.net/project_modules/1400/4ce18840076437.5606d1c11c21d.png
- https://www.artstation.com/artwork/k40w20

Mechanics

- https://www.gamasutra.com/view/feature/2029/developing_your_own_replay_system.php

Useful links

Link to prototype: https://nickhaecker.github.io/A-fraction-of-time/Release/index.html

Link to GitHub-Repository: https://github.com/NickHaecker/A-fraction-of-time