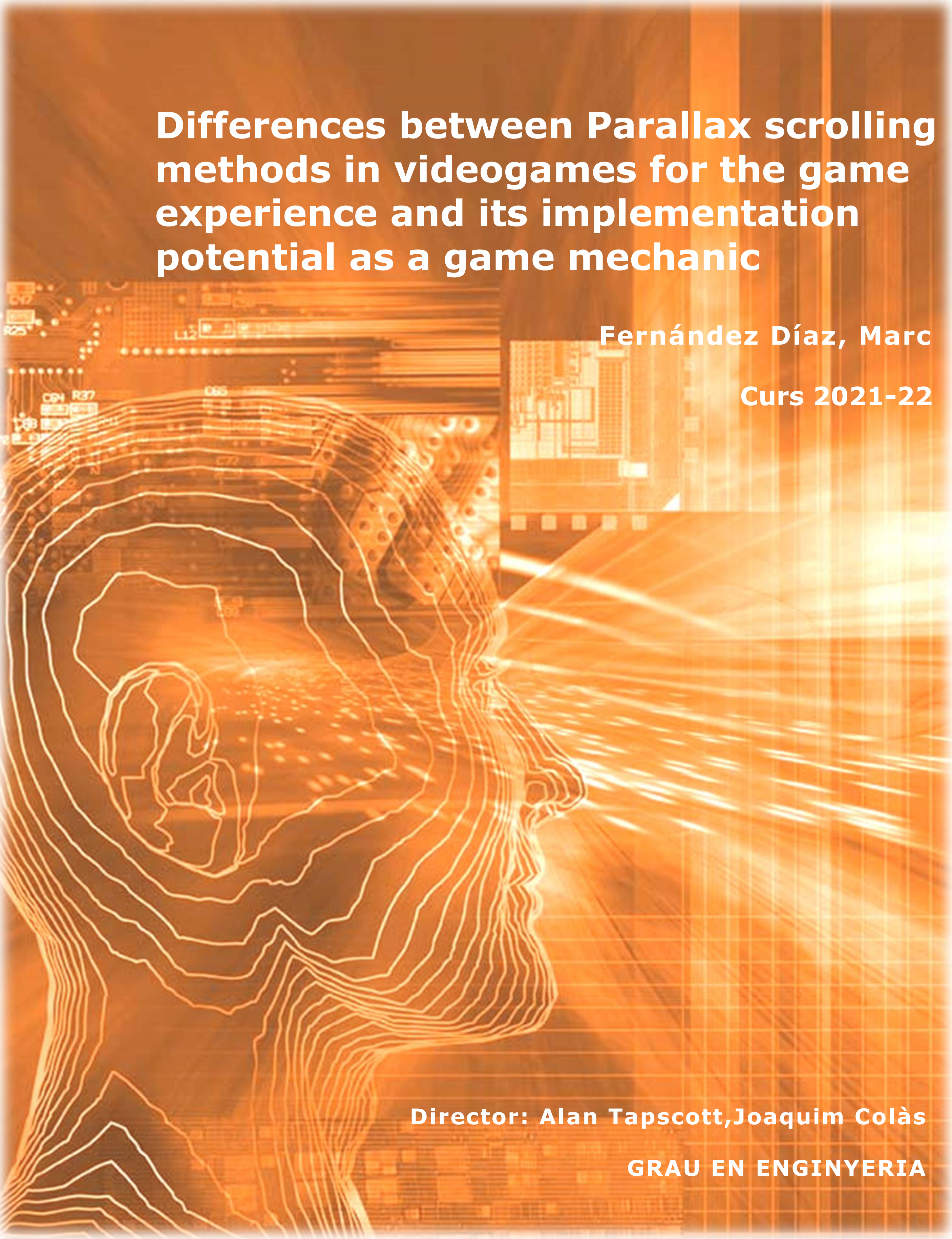


Differences between Parallax scrolling methods in videogames for the game experience and its implementation potential as a game mechanic

Fernández Díaz, Marc

Curs 2021-22

Director: Alan Tapscott,Joaquim Colàs

GRAU EN ENGINYERIA

Degree Final Project

upf. Universitat Pompeu Fabra Barcelona

Escola d'Enginyeria

Acknowledgement

Thanks to my family, friends and colleagues who helped me during the process of development of this project. To all the teachers of this degree and my supervisors Alan Tapscott and Joaquim Colàs, that gave me the knowledge and guidance to complete this work.

Abstract

In this Final Degree Thesis, I focused on the study of the effect of “parallax” on videogames. This effect is used in a computer graphics technique called parallax scrolling. It is also used in videogames to create a 3D feel in 2D side-scrolling games, which is where I focused on this work. Specifically, I focused on the differences for players in the four implementation methods, by creating different levels one with each method in Unity. Once I created those, I made some players play it and answer a form about gameplay, negative effects they felt and potential of parallax as a mechanic. With the results obtained, I made a level of a game with parallax as one of the main mechanics, based on the forms, and made the previous players play it. When they finished it, they did another form like the previous one, but with minor changes, and compared the results between them to extract conclusions.

Resumen

En este Trabajo Fin de Grado me he centrado en el estudio del efecto del “parallax” en los videojuegos. Este efecto se utiliza en una técnica de gráficos por computadora llamada desplazamiento de parallax. También se usa en videojuegos para crear una sensación 3D en juegos de desplazamiento lateral en 2D, que es donde me centré en este trabajo. Específicamente, me enfoqué en las diferencias para los jugadores en los cuatro métodos de implementación, creando diferentes niveles, uno con cada método en Unity. Una vez que los creé, hice que algunos jugadores lo jugaran y respondieran un formulario sobre el juego, los efectos negativos que sintieron y el potencial del “parallax” como mecánica. Con los resultados obtenidos realicé un nivel de un juego con el “parallax” como una de las principales mecánicas, basándome en el formulario, e hice jugarlo a los jugadores anteriores. Cuando lo terminaron, hicieron otro formulario como el anterior, pero con cambios menores, y compararé los resultados entre ellos para sacar conclusiones.

Resum

En aquest Treball Fi de Grau m'he centrat en l'estudi de l'efecte del parallax als videojocs. Aquest efecte s'utilitza en una tècnica de gràfics per ordinador anomenada desplaçament de parallax. També es fa servir en videojocs per crear una sensació 3D en jocs de desplaçament lateral en 2D, que és on em vaig centrar en aquest treball. Específicament, vaig enfocar-me en les diferències per als jugadors en els quatre mètodes d'implementació, creant diferents nivells, un amb cada mètode a Unity. Quan els vaig creure, vaig fer que alguns jugadors el juguessin i responguessin un formulari sobre el joc, els efectes negatius que van sentir i el potencial del parallax com a mecànica. Amb els resultats obtinguts vaig realitzar un nivell d'un joc amb el parallax com una de les mecàniques principals, basant-me en el formulari, i vaig fer jugar-lo als jugadors anteriors. Quan ho van acabar, van fer un altre formulari com l'anterior, però amb canvis menors, i vaig comparar els resultats entre ells per treure'n conclusions.

CONTENTS

1.INTRODUCTION……………………………………………………………...9

1.1 Problem/Opportunity……………………………………………….. 9

1.2 Idea/Goal…………………………………………………………….. 10

2. STATE OF THE ART…………………………………………………...11

3. DEVELOPMENT TOOLS…………………………………………...13

3.1 Unity…………………………………………………………... 13

a) Unreal Engine………………………………………………… 13

3.2 C++………………………………………………………………. 14

3.3 GIMP……………………………………………………………. 14

3.4 Visual Studio…………………………………………………... 14

4. DEVELOPMENT…………………………………………………...15

4.1 Requirements…………………………………………………. 15

a) Functional…………………………………………………….. 15

b) Non-Functional……………………………………………… 15

4.2 Prototypes……………………………………………………… 15

a) Test levels……………………………………………………... 15

b) Parallax mechanic game……………………………………….. 17

4.3 Development of the game……………………………………….. 18

a) Assets……………………………………………………………. 18

b) Test levels……………………………………………………….. 18

c) Parallax mechanic game……………………………………….. 21

5. TESTING RESULTS……………………………………………………25

5.1 Evaluation Method……………………………………………...25

a) Test levels results……………………………………………… 25

b) Parallax mechanic game results……………………………….27

6.CONCLUSION………………………………………………………29

FIGURE LIST

Figure1. Unity logo ……………………………………………………………………....…13
Figure2. Unreal logo ………………………………………………………………………13
Figure3. Pure platformer prototype ……………………...…………………….....…16
Figure4. Pure platformer with enemies prototype ……...………………………...…..16
Figure5. Pure action game prototype ………………...……………………...…....16
Figure6. Action game with some platforming prototype ……………………..…...16
Figure7. Puzzle mechanic ...………………………………………...………….....….17
Figure8. Side-scroller background swap mechanic....………………………………17
Figure9. Metroidvania Controller image ...…………………………………………18
Figure10. Free 2D Cartoon Parallax Background image………………………...18
Figure11. Clamp error ...…………………………………………………………...19
Figure12. Layer level parallax ...…………………………………………………..20
Figure13. Sprite order in layer ...………………………………………………… 20
Figure14. Sprite level parallax……………………………………………………20
Figure15. Raster level parallax ...…………………………………………………20
Figure16. Level layout ...………………………………………………….……..21
Figure17. Shrink animation ...……………………………………………….…..22
Figure18. Original animator ………………………………………………….….22
Figure19. Remodelled animator ....……………………………………………..…22
Figure20. Ground deactivated .....……………………………………………...…22
Figure21. Ground activated ...……………………………………………….....…22
Figure22. Enemy speed to 0 ...……………………………………………….……23
Figure23. Kill zone trigger ...………………………………………………...……23
Figure24. Pause menu canvas ...………………………………………...……...…24
Figure25. Click event pause ...…………………………………………………..…24
Figure26. Homemade background layer ...…………………………………….…24
Figure27. Finished parallax mechanic level...………………………………………24
Figure28. Test level first section first question ...……………………………...….25
Figure29. Test level first section second form question ...……………………….25
Figure30. Test level first section third question ...………………………......….26
Figure31. Teste level second section first question ...………………….…..…...26
Figure32. Test level second section second question ...………………………..…26
Figure33. Test level second section third question ...……………………….....…26
Figure34. Test level third section first question ...…………………………...……27
Figure35. Test level third section second question………………………………27
Figure36. Test level third section second question ...……………………………27
Figure37. Test level third section fourth question ...……………………….…..27
Figure38. Game first section first question ...………………………………….27
Figure39. Game first section second question ...………………………………… 27
Figure40. Game second section first question ...……………………...………… 28
Figure41. Game second section second question ...……………………...………28
Figure 42. Game second section third question ...……………………………..…28
Figure 43. Game third section first question ...……………………………...…… 28
Figure 44. Game third section second question ...……………………………..…28

1. Introduction

Video games are the main source of entertainment of most of children and young people nowadays and is increasing each year. A study [1] on how University students spend their leisure time, studied about video games as they have highly motivating properties and found that a challenge is the top-ranking motivation for the players. As the numbers mentioned before are so big, many of these players are multiplayer and competitive ones. This means that most of the popular games are not 2D, as the majority of those are not oriented to that kind of playstyle. For this reason, the potential of using and discovering new techniques or uses for already existing ones in 2D is not to the fullest, like parallax.

Parallax [2] is the observed displacement of an object caused by the change of the observer's point of view. In video games, this is used in a technique called "parallax scrolling". This technique consists of the difference of speed between the layers of images to create a sensation of depth in 2D. There are four different methods of implementing this technique: layer method, "sprite" method, animation method and raster method.

The idea of studying about parallax, specifically the possibility of being able to use this technique as an active part of the game, comes from personal experience. I've been consuming video games for years, but with the onset of the global pandemic due to COVID-19, both the amount of time people spends playing and the number of games has increased exponentially. Because of this, I have had more time at home to try out many games, including single player ones. Many "indi" games are usually in this style and especially 2D, so playing these I was fascinated by how parallax worked. It was in this way that I came to the point of reminding myself of other games that used perspective within the mechanics of these and wondering if parallax could also have that character and what ways would be best suited to do so.

1.1 Problem/Opportunity

Even if the competitive and multiplayers games take most of the player base as this study demonstrates [1], there is still many people that support and play single payer ones. Also, they are not mutually excluding, so players from one base can be at both or switch. This means that if something new pick the interest of the player in the competitive base, they can try the 2D single player games.

Parallax scrolling [3] is a technique that has not been explored at its fullest with the actual tools, as it has lost popularity during these years as the game development improved. Also, there are not many games that include this as a part of the game mechanics. Some examples are: The Witness, Paper Mario and Parallax.

If there is a lack of this, there is the possibility to improve and discover some game mechanics and techniques with much potential, but there is also the chance that as this has low popularity it will not have much expectance.

Furthermore, there might be some negative effects due to the movement of the whole scene, like headache, nausea, blurry screen, etc… also depending on the method used in the implementation.

1.2 Idea/Goal

As there is room for creating and improving with the parallax in 2D, I wanted to know how much expectance a player would have even though this is not very known or popular. Also, the effects the different methods would have on the players to improve the game experience.

To do so I built a game with Unity, that consists of 5 different levels. The first one would be without parallax and the rest with one different implementation each, to make the players notice the changes. As I wanted to make the players focus on the background, I made the level layout the same at each stage, so they focus on the parallax effect.

Once they finish the game, I provide them with a form asking about the gameplay, negative effects they felt and depth sensation, and their thoughts and expectations of parallax as a mechanic.

Next, I create the game with parallax as a mechanic based on the feedback of the form and let the players play it.

Again, once they finish, I give them another form, very similar to the previous but with some minor changes. Then I take the results of both feedbacks to see which method is better or worse, the negative effects and the expectations they had and the actual impression from the game they tested.

2. STATE OF THE ART

As this article [4] says ‚2D games were the norm back in the 1990’ as the technology was not developed enough. Then the 3D games appeared bringing something new and, with the faster growth of the industry, became the main games for many large companies.

The main difference of the 3D games with the 2D ones is the three-dimensional plan of the game. As this increases the number of possibilities the player has and reduces the number of limitations for the developers it became so popular.

This did not shadow the 2D games completely, but the popularity of those dropped and it slowly started to be related with “indi” or small companies that developed them because the ease of it as 3D games are more complex and a high performance is hard to achieve.

As many 2D games used artificial parallax, the use of this technique also dropped with the number of games developed. That made the potential research and development of this technique lower and that is when I realized there was still room to discover new applications for a relatively old technique in the video game history.

3. DEVELOPMENT TOOLS

3.1 Unity

Unity [5] is a cross-platform game engine developed by Unity Technologies, which is primarily used to develop video games and simulations for computers, consoles and mobile devices. It supports 2D and 3D graphics, drag and drop functionality and scripting through C#. Within 2D games, Unity allows importation of sprites and an advanced 2D world renderer.

Figure shows this framework logo and the Unity official website

Figure 1. Unity logo. [*https://unity.com*](https://unity.com)

Other possible alternatives to this engine, that could fulfil the same goal of creating the game are:

a) Unreal Engine

Unreal Engine [6] is a 3D computer graphics game engine developed by Epic Games. Initially developed for PC first-person shooters, it has since been used in a variety of genres of games and has seen adoption by other industries, most notably the film and television industry. Written in C++, the Unreal Engine features a high degree of portability, supporting a wide range of desktop, mobile, console and virtual reality platforms.

Figure 2. Unreal logo. [*https://www.unrealengine.com/en-US*](https://www.unrealengine.com/en-US)

This is a very potent engine which also uses C++, but for this project is not needed that much. Also, the Unity engine is much easier to handle without experience and will take less time to learn than Unreal.

3.2 C++

C++ [7] is a cross-platform language that can be used to create high-performance applications and gives programmers a high level of control over system resources and memory. It is one of the world's most popular programming languages.

Can be found in today's operating systems, Graphical User Interfaces, and embedded systems and is an object-oriented programming language which gives a clear structure to programs and allows code to be reused, lowering development costs. It is also portable and can be used to develop applications that can be adapted to multiple platforms.

3.3 GIMP

GIMP [8] is a cross-platform image editor available for GNU/Linux, macOS, Windows and more operating systems. It is free software, you can change its source code and distribute your changes like photo retouching, image composition and image authoring.

3.4 Visual Studio

An *integrated development environment* (IDE) is a feature-rich program that supports many aspects of software development. The Visual Studio IDE [9] is a launching pad that you can use to edit, debug, and build code, and then publish an app. Visual Studio includes compilers, code completion tools, graphical designers, and many more features to enhance the software development process.

4. DEVELOPMENT

4.1 Requirements

a) Functional requirements

- Functional game loop to not stuck the player.
- Parallax levels and mechanic with some challenge that requires movement.
- Make the player focus on the background or parts with the implementation of the parallax scrolling.
- Good player movement so it does not interfere with the tests.
- Clean animations so it gives a depth feeling.

b) Non-Functional requirements

- Performance to not downgrade the parallax effect.
- Scalability to be able to implement more levels if needed and make them quick.
- Availability to let the players play the game at any moment.
- Maintainability so can be easily updated if needed.
- Development Standards to minimize code errors.

4.2 Prototypes

Before implementing the parallax effect to the background, I had to design the levels and how they worked. To do that I considered that it had to have movement to be able to use the parallax. Therefore, I had many options, but as the game itself and the mechanics are not the most important part I decided to go for a classic side-scrolling game with enemies. That is because the implementation of the game is simple, they usually have parallax scrolling and the players will easily understand it.

a) Test levels

First, for the testing levels I had to do five different levels, but with the same layout as I wanted to make the players notice the changes of the background between those. This are the different designs:

- Pure platformer (level goes upwards)
  - Background does not follow the camera as with the verticality will feel strange.
  - Vertical only parallax can be applied to solve the previous problem.
  - More complex to play than a normal side-scrolling platformer without much height.

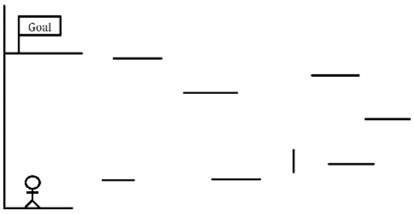


Figure 3. Pure platformer prototype

- Pure platformer with enemies (level goes upwards):
  - Background does not follow the camera as with the verticality will feel strange.
  - Vertical only parallax can be applied to solve the previous problem.
  - More complex to play than a normal side-scrolling platformer without much height.
  - Enemies add more difficulty.

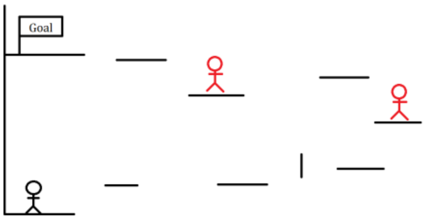


Figure 4. Pure platformer with enemies prototype

- Pure action game (level goes sideways):
  - Background follows the camera.
  - Beat all enemies to pass the level.
  - Vertical axis not used (camera will not move up even with jumping).

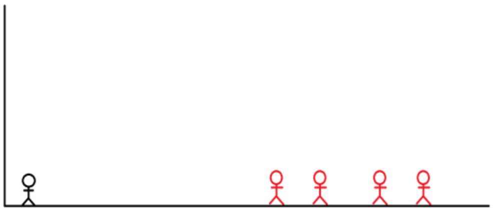


Figure 5. Pure action game prototype

- Action game with some platforming (level goes sideways):
  - Background follows the camera.
  - Beat the boss to pass level.
  - Vertical axis used.
  - First enemy weak to let the player test the combat.
  - Some platforming to test the parallax vertically too.

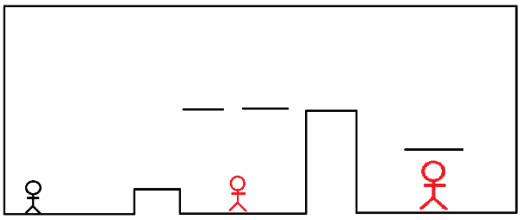


Figure 6. Action game with some platforming prototype

I considered the ups and downs of each prototype and decided that the levels that went upwards were too much of a stretch to test what I wanted too and may not give the best results. Then, I had two options, but the one with only enemies would be worse to study the negative effects and all the other aspects I wanted too, so I chose the last one.

b) Parallax mechanic game

After the implementation and study of the test levels, I started the prototype of the game with parallax as a mechanic knowing with method implement it with. I had two main ideas, do it as a puzzle where you must align the different backgrounds to solve some problem, or a side-scroller where you must change between the background and the level to dodge some elements and enemies.

- Puzzle mechanic:
    - Move the player to solve different puzzles
    - Player has not much mobility
    - Background moves slow

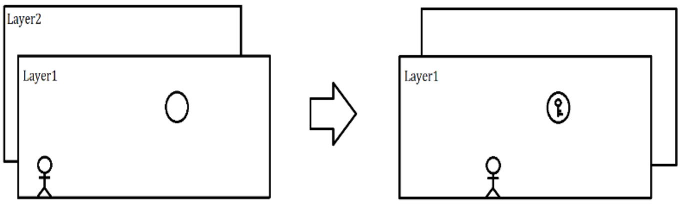


Figure 7. Puzzle mechanic

- Side-scroller background swap mechanic:
    - Survive until the end of the level dodging the enemies and obstacles.
    - Player has mobility but only controls one part (autorun).
    - Background moves fast.
    - As one layer moves faster than the other, the player must calculate the speed to jump at the right time.

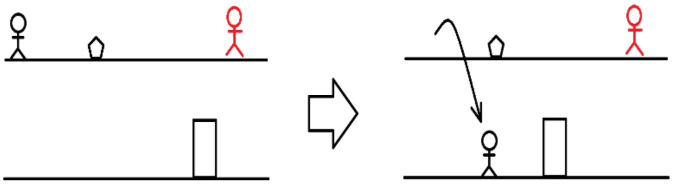


Figure 8. Side-scroller background swap mechanic

So, before the implementation I asked 5 players of the previous test about which game would they play. Four of them said that the side-scroller was clearer and more intuitive, and that was like the previous levels they played. That was a very important aspect to take into account, so I decided to proceed with the second option.

4.2 Development of the game

After the process of selection done in the previous part, I had a clear idea of what to build, but not how. First, I searched for some free Unity assets that could help me relieve the amount of work that was not related with parallax, the main part of the project.

Once that was done, I implemented the parallax for each method and created a level for the tests.

Then, for the parallax mechanic level, I chose the method with the better results from the form and created two ground layers for the mechanic. After changing the animations and the animator conditions to match the parallax, I did the level and created and modified different scripts for the obstacles, enemies and game loop. One important add was the pause menu and the different buttons.

a) Assets

- Metroidvania Controller:

  2D character controller system that includes sprites, animations, sounds and scripts for the movement, enemies and other behaviours.

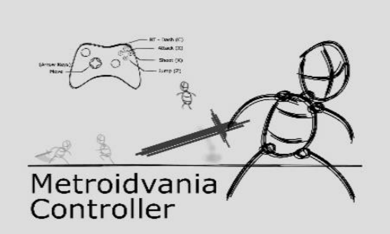


Figure 9. Metroidvania Controller image

- Free 2D Cartoon Parallax Background:

  4 different backgrounds, divided by layers.

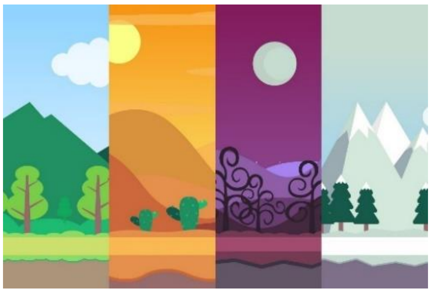


Figure 10. Free 2D Cartoon Parallax Background image

b) Test levels

To begin I created a scene and added the main camera and player that I got from the assets. Then I dragged the background images to the scene and started to implement the parallax with the animation method.

In order to do that I added a material to the images and created a script named “ParallaxAnimation”. This script gets the velocity of the player, a speed that I use to control how fast the layer moves (parallax factor) and the background material. Then, I calculate the offset that the layer has to move with:

```
offset = (player.velocity.x * 0.1f) * speed * Time.deltaTime;
```

With this offset I added it to the material offset:

```
material.mainTextureOffset += offset;
```

The problem with this is that the image wrap mode was in “Clamp” and this happened:

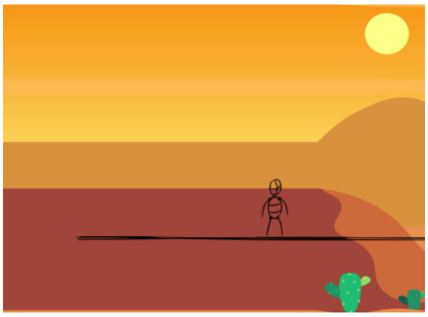


Figure 11. Clamp error

To fix that the wrap mode had to be in “Repeat”.

Next was the “Layer Level”, which I added the background layers with different depth and created “ParallaxLayer” script. In the script multiply the parallax effect for the camera position respect its start.

```
pos.x += parallaxEffect * (cam.transform.position.x - startCamerapos.x);
pos.z = transform.position.z;
```

To make it “infinite” I added a condition that if the layer is going to be out of the càmera, it sums the length of the sprite to the start position, to move it to the center again.

```
float temp = cam.transform.position.x * (1 - multiplier);
if (temp > startpos.x + length){startpos.x += length;}
else if (temp < startpos.x - length) {startpos.x -= length;}
```

To not let a space without image to the sides, I added one copy of the layer to each side. And the parallax looks like this:

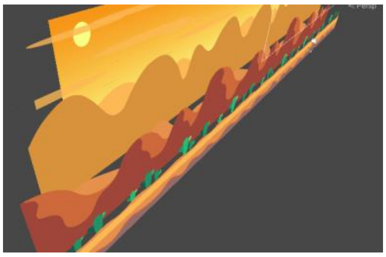
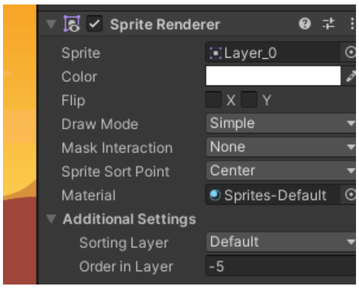


Figure 12. Layer level parallax

Next is the “SpriteLevel”, which is very similar to the layer one but there is only one plane. Then, I assigned to each layer and order in the “Sprite renderer”.



```
Sprite Renderer
Sprite                 Layer_0
Color
Flip                   X  Y
Draw Mode              Simple
Mask Interaction       None
Sprite Sort Point      Center
Material               Sprites-Default
Additional Settings
  Sorting Layer        Default
  Order in Layer       -5
```

Figure 13. Sprite order in layer

And the parallax in this method looked like this:

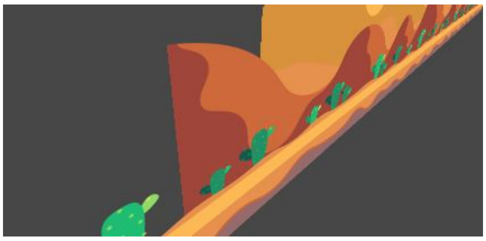


Figure 14. Sprite level parallax

For the last method I used the same parallax script as the previous one. However, this simulates the rasterization many old games used that divide the layer in horizontal strips. Therefore, I divided the image in one layer with several strips and moved them at different speeds. It looked like this:

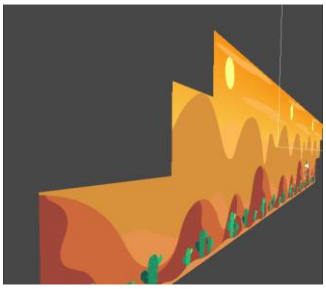


Figure 15. Raster level parallax

Once the background was fully implemented, I created the level layout as the prototype, a text to know which level are you at and added a scene loader script to change between each of the 5 levels once the boss was defeated.

Final look of the levels shown in figure 16.

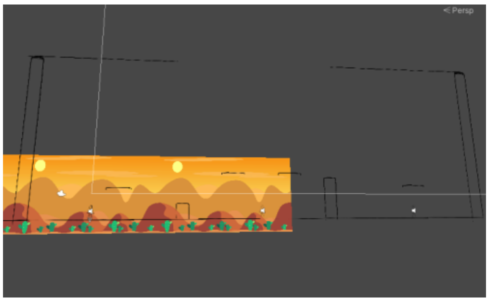


Figure 16. Level layout

c) Parallax mechanic game

The first step to create the game with the parallax mechanic is the same as the test levels, implement the camera and player. Then I imported the “SpriteParallax” script that gave better results in the previous levels.

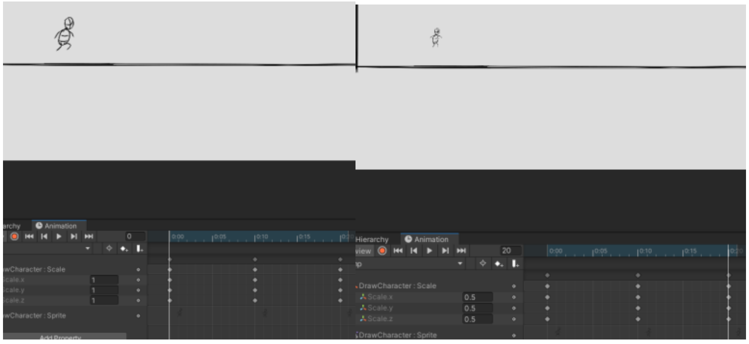
Then, I created to objects that would be the ground for the near part with more parallax and the far one with less. I added the script to both of the objects and changed the player movement script so it runs automatically after 2 seconds from the start, and change between layers jumping with the spacebar.

```
time += 1f * Time.deltaTime;

if (time < startTime)
{
      time += 1f * Time.deltaTime;
}
else
{
      horizontalMove = runSpeed;
}

if (Input.GetKeyDown(KeyCode.Space) && isNear && !isEnd)
{
      animator.SetBool("isShrink", true);
      controller.nearJump = true;
      jump = true;
}
```

I created the variables “isNear” and “isFar” to know which layer the player is and created an animation that shrinks the player to make it look like the player is far.



wCharacter : Scale
Scale.x 1
Scale.y 1
Scale.z 1
wCharacter : Sprite

DrawCharacter : Scale
Scale.x 0.5
Scale.y 0.5
Scale.z 0.5
DrawCharacter : Sprite

Figure 17. Shrink animation

When all the animations created where need I added them to the animator and created the variables “isShrinked and “isEnd” inside it to create the conditions between animations.

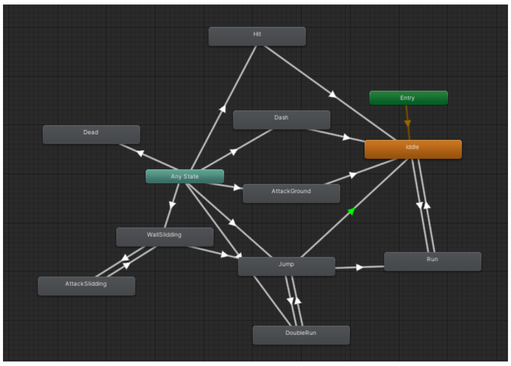


Figure 18. Original animator

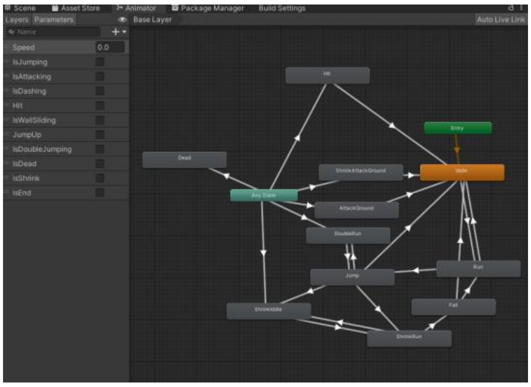


Figure 19. Remodelled animator

Next, I had to let the player switch between near and far ground, but the problem is that if the higher ground had the collider active it collisioned with the player. So, I deactivated the far ground and made a trigger that when is entered activates that ground.

```
if (other.CompareTag("Player")){
    if (ground.enabled) {
        ground.enabled = false;
        player.isFar = false;
        player.isNear = true;
    }
    else {ground.enabled = true;
        player.isNear = false;
        player.isFar = true;
    }
```

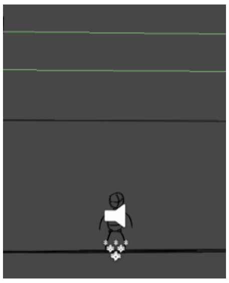


Figure 20. Ground deactivated

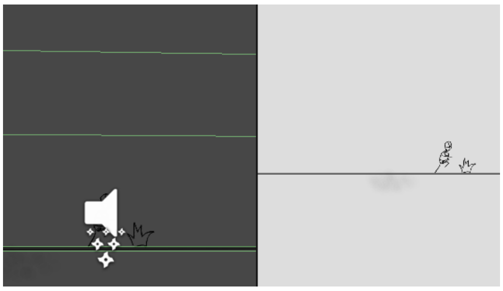


Figure 21. Ground activated

As I wanted to make the game different when the player is near and far, I only let the player shoot when far and attack when near.

```
if (Input.GetKeyDown(KeyCode.X) && canAttack && player.isNear)

if (Input.GetKeyDown(KeyCode.X) && canAttack && player.isFar)
```

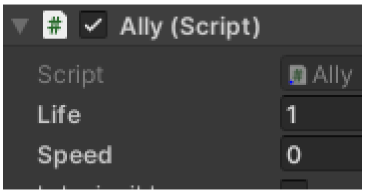
I also had to modify the speed of the projectile as the shrink animation made the direction half and that made the projectile go twice as slow.

```
if (transform.localScale.x < 1)
{
      direction = new Vector2(transform.localScale.x * 2, 0);
}
```

When I finished with some bugs in the coding, I started building the level and creating the obstacle script, that on collision damages the player.

```
if (other.CompareTag("Player"))
{
        player.ApplyDamage(damage, transform.position);
        Destroy(gameObject);
}
```

I noticed then that the rest of the movement features, like double jump wall jump, wall sliding, where not necessary and removed them. After that, added some enemies to the level removed their range attack and made them static setting their speed to zero.



Ally (Script)

Script Ally

Life 1

Speed 0

Figure 22. Enemy speed to 0

I also added a trigger, with the “KillZone” trigger that the asset gives, in the end parts where you can fall.

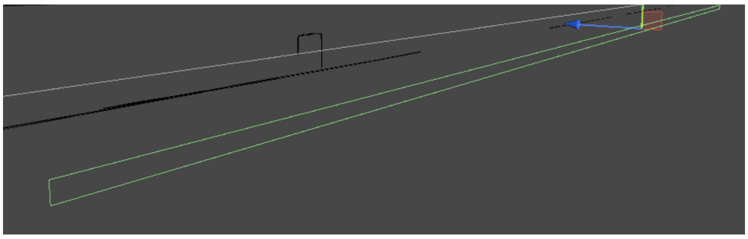


Figure 23. Kill zone trigger

To close the game loop, I created a trigger at the end and a script attached to it that stops the player

```
        if (other.CompareTag("Player")){
            player.isEnd = true;
            player.runSpeed = 0.0f;
            animator.SetBool("isEnd", true);}
```

The end just stops the player, because I implemented a pause menu where the player can: resume, restart or quit.

```
if (Input.GetKeyDown(KeyCode.Escape) && pauseMenu.activeSelf == false){
            Cursor.visible = true;
            audioMute.changeMute = true;
            Pause();}
```

The other methods are called when clicked on the buttons implemented in the canvas by clicking.

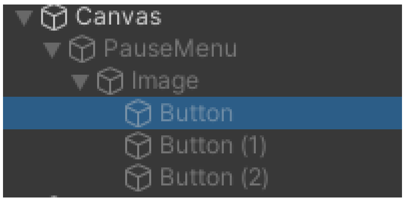


Figure 24. Pause menu canvas

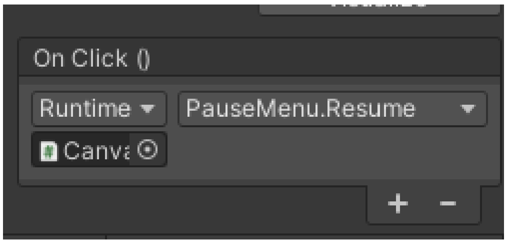


Figure 25. Click event pause

Now I only had left the implementation of the background with the parallax. I opened the sprites for the grass, leaf and rocks with Gimp and created an image for the ground parallax.

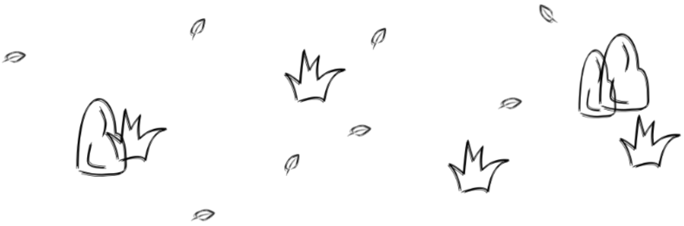


Figure 26. Homemade background layer

Then created an object with the image that follows the player, and one more to each side, with a new script "ParallaxBackground", that adds the infinite behaviour that I did for the test levels.

The finished level looked like this:

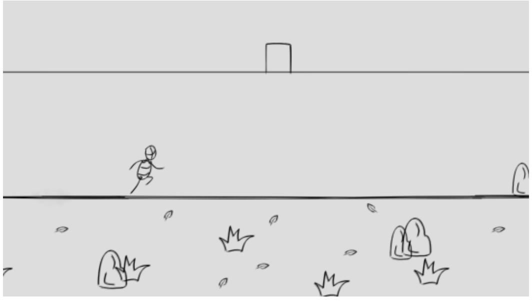


Figure 27. Finished parallax mechanic level

```
The link to the github the project is uploaded:
https://github.com/TrueMarcaBlanca/ParallaxGame
```

5. TESTING RESULTS

The evaluation for the tests levels was done via Google Form and it was divided in three different sections.

The first one was about the gameplay, asking whether if the background with the parallax affected that in some way or not. The first questions “Did the background interfere with the game mechanics?” and “Did the background movement affect yours at any point?” were aimed to asked for the interferences in the gameplay from the parallax that the users perceived. On the contrary, the next question “Did you feel the player movement better at any point?” was asking for improvements in the gameplay. I asked the players to point which level in case their answer was positive.

On the second part, I focused on the negative effects on the user and the 3D perception. I first asked for some ache or sickness during the playthrough of the game or after. The las two questions of this section “Did you get distracted and lost focus of the screen or missing something due to the background?” had a similar purpose as the previous one, but “Which level did you feel more depth?” was focused on extracting information about which implementation did the players preferred.

For the last section the first three questions “Do you think this method used to give depth can be used as a game mechanic”, “Do you think this could interfere with the rest of the gameplay?” and “Do you think that this can be used to something more than give depth to the background?” are questions to get information about their thoughts prior the game with the parallax mechanic. And for the last one, “How much expectations do you have to play a game with parallax as a mechanic?”.

5.1 EVALUATION METHOD

a) Test Levels Results

In the test levels questionnaire, the Level one refers to the one without parallax, the second to the animation method, the third to the raster method, the third

The first questions were “Did the background interfere with the game mechanics?” and “Did the background movement affect yours at any point?”

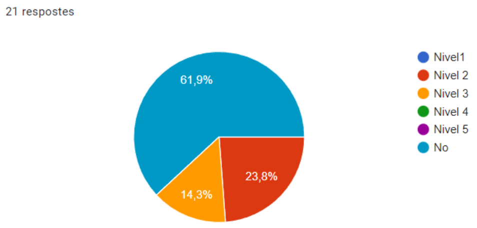


Figure 28. Test level first section first question

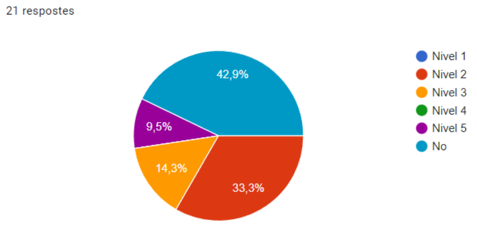


Figure 29. Test level first section second form question

Most of the players did not have problems with it interfering with the gameplay itself, but the animation method level made the modify in some way their movement, which a red flag for that method.

The last question of this section was “Did you feel the player movement better at any point?”.

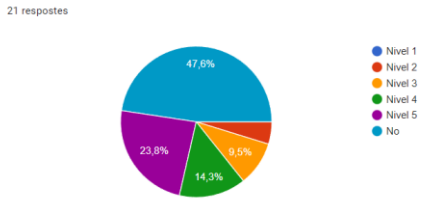


Figure 30. Test level first section third question

Almost half of the answers were that they did not feel any improvement in the movement just by the parallax effect (47,6%), but a quarter of them did in the sprite method level and that is a decent amount to consider it relevant (23,8%).

For the second section I asked “Did you feel any ache or sickness due to the background movement? Dizziness, headache, nausea, etc…”.

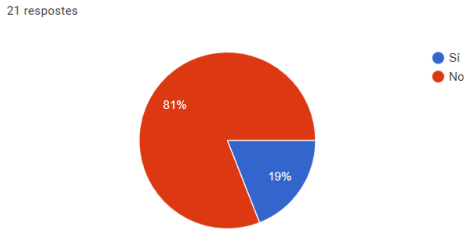


Figure 31. Teste level second section first question

Almost none of the players felt any illness during the test.

For the second question I asked them “Did you get distracted and lost focus of the screen or missing something due to the background?”

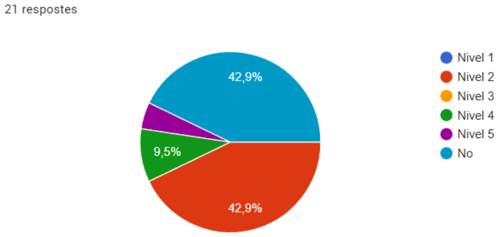


Figure 32. Test level second section second question

Almost half of the players said they did not have any problems, but the same amount did in animation level.

And for the last question of section 2 “Which level did you feel more depth?”.

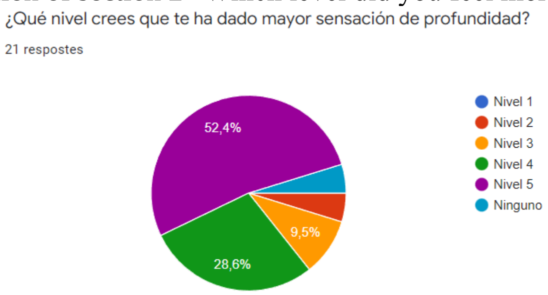


Figure 33. Test level second section third question

The sprite method got more than half of the answers (52,4%)

For the last section the first three questions “Do you think this method used to give depth can be used as a game mechanic”, “Do you think this could interfere with the rest of the gameplay?” and “Do you think that this can be used to something more than give depth to the background?”

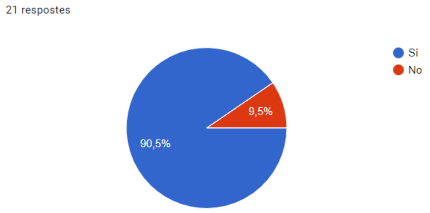
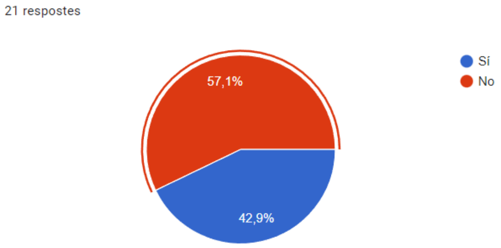


Figure 34. Test level third section first question



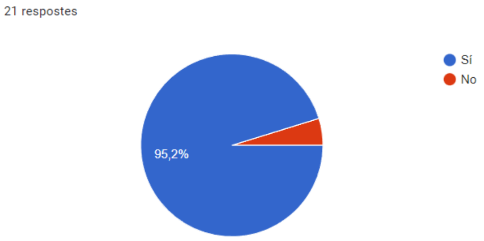


Figure 35. Test level third section second question *Figure 36. Test level third section second question*

Almost all players agree that parallax can be used as a mechanic and many other aspects (>90%), but they are not sure if that can interfere with the rest of the gameplay as the answers are near the half.

For the last question “How much expectations do you have to play a game with parallax as a mechanic?”

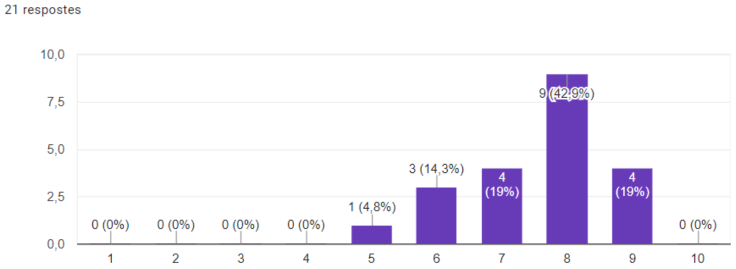
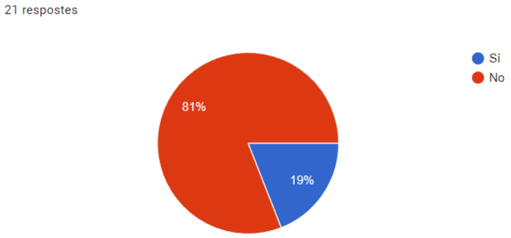


Figure 37. Test level third section fourth question

The players have a relative high expectation as most answers are concentrated around the 8.

b) Parallax mechanic game results

The two questions of the first section are the same as the previous questionnaire but applied to the game.



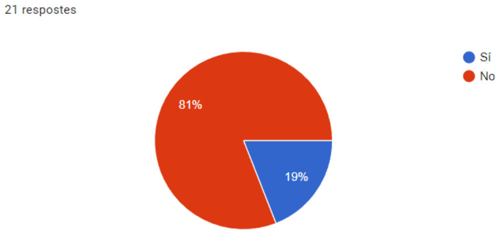


Figure 38. Game first section first question *Figure 39. Game first section second question*

The majority of the did not find any problems in the gameplay.

In the second section the two first questions were the same, but the third was “How do you grade the depth sensation you felt during the game?”.

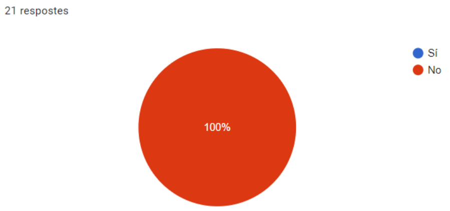


Figure 40. Game second section first question

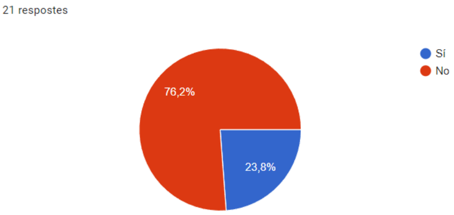


Figure 41. Game second section second question

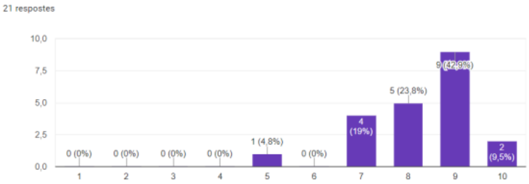


Figure 42. Game second section third question

The last section had two questions only, the first is the same as the other test, but the last on was “Considering the expectations of the previous form, how do you grade the game?

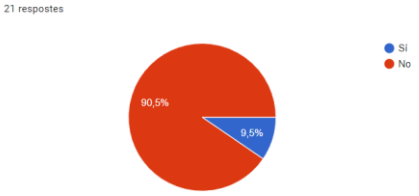


Figure 43. Game third section first question

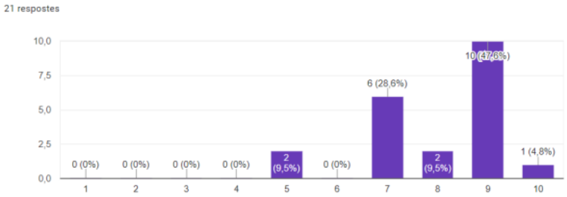


Figure 44. Game third section second question

6 Conclusion

From the results of the first test, we can extract that there were some problems with the parallax regarding the gameplay, being the animation method the one with worst results, But the animation and raster method got pretty good feedback. Regarding the negative effects on the player, not many users had problems and the sprite method got the most answers for the most depth sensation with more than half, but the animation method gave the players many distractions with a 42,9% of them saying so. Furthermore, almost everyone though that parallax has many more uses and high potential to be a mechanic, but they are not sure if that could interfere with the rest of the gameplay.

If we take the results of the second form and compare with the previous one, they improved a lot with a high depth sensation, but as some grades of the game compared to the expectations went up some went a bit down, despite being near of the previous grade.

To sum up, the animation method was the worst rated by the players, even giving some problems to the gameplay and them. The layer method, despite being similar to the sprite method, was the second worst rated from the four methods. The raster method had better responses than I thought it will at first, but the one that got the best feedback by far was the sprite method.

With that information, I worked on the development of the game. From the results of the form, I concluded that the implementation method was very important to make the user have a pleasant experience. However, as some of the players rated it better than the expectations and some did not, I really think that there is potential for this kind of game but maybe the implementation was not the best or the game category was not the most popular.

BIBLIOGRAPHY

[1] Hainey, T., Connolly, T., Stansfield, M., & Boyle, E. (2011). The differences in motivations of online game players and offline game players: A combined analysis of three studies at higher education level. Computers & Education, 57(4), 2197-2211.

[2] Lucas, J., & Pultarova, T. (2022, 11 enero). *What Is Parallax?* Space.Com. Available: https://www.space.com/30417-parallax.html

[3] P. (2021, 2 octubre). *Parallax Scrolling in pixel-perfect 2D Unity games*. Pav Creations. Available: https://pavcreations.com/parallax-scrolling-in-pixel-perfect-2d-unity-games/

[4] Starloop Studios. (2022, 5 junio). *3D Vs. 2D: The Eternal Battle to Develop Video Games*. Available: https://starloopstudios.com/3d-vs-2d-the-eternal-battle-to-develop-video-games/

[5] freeCodeCamp.org. (2021, 28 abril). *Unity Game Engine Guide: How to Get Started with the Most Popular Game Engine Out There*. Available: https://www.freecodecamp.org/news/unity-game-engine-guide-how-to-get-started-with-the-most-popular-game-engine-out-there

[6] Unreal Engine | The most powerful real-time 3D creation tool. (2022). Unreal Engine. Available: https://www.unrealengine.com/en-US

[7]C++ Introduction. (2022). w3Schools C++ Introduction. Available: https://www.w3schools.com/CPP/cpp_intro.asp

[8] About GIMP. Official webpage. Available: https://www.gimp.org/about/

[9] T. (2022, 30 abril). *Overview of Visual Studio*. Microsoft Docs. Available: https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2022