UNIVERSITAT JAUME I

Erin’s adventure: arena mode

Exploring the mechanics of fluidity and combo system

Technical report of the Final Degree Project

Carlos Fuente Merino

Abstract

This Degree Final Project is aimed to explore differents combat styles with a solid combo system with multiple possibilities. To potentiate that mechanics, they will be accompanied with lots of animations that will make easier the understanding of what happens in the game in every moment.

This mechanics has been integrated in Erin's Adventure: arena mode, a game in 2.5D with lateral scroll and combat mechanics. This project has been developed with Unity 5.

In this document is described the game creation process of the game previously named. This includes the part of game design, development and art creation, among others.

The document ends with some conclusions related with the game development and what it was learned in it.

KeyWords

Game

Combo system

Animations

Weapons

Development

INDEX

INDEX

FIGURES INDEX

FIGURE INDEX

ABSTRACT

This paper will try to justify my concept of DFP. The concept consists In a game developed in Unity3D with graphics in 2.5D that mixes a Metroidvania gameplay style with combat based in combos. The game will stand out for having dynamics fights with animations that highlight the fluidity.

1 TECHNICAL PROPOSAL

INTRODUCTION

The project will be developed in Unity3D and will have graphics in 2.5D in lateral scroll (*This War Of Mine*). The mechanics will be based on combos fight, where the player will face a large number of enemies and will have to change weapons every now and then, providing the fighting with a lot of fluidity and variety. Each type of weapon, all melee, will have different mechanics, forcing the player to adapt quickly to the new weapon obtained. Weapons will be obtained from defeated enemies and can be caught by the player without cutting the gameplay. The game is set in the near future, where artificial intelligence begins to have real power.

The reason why the project is going to be developed is because during the years in the Design and Development of videogames career has never been developed any big video game, all has been remained as concepts or demos. Then, the project will be a game, with all the sections of a game (art, programming, planning ...).

In addition, the project aims to visit that part of the world of videogames that explore the mechanics of combat. Finally, with this game it will be explored all the subjects in the degree and it will be learned to develop a large-scale game.

RELATED SUBJECTS

- VJ1223 Arte del videojuego
- VJ1227 Motores de Juego
- VJ1218 Narrativa hipermedia y análisis de videojuego

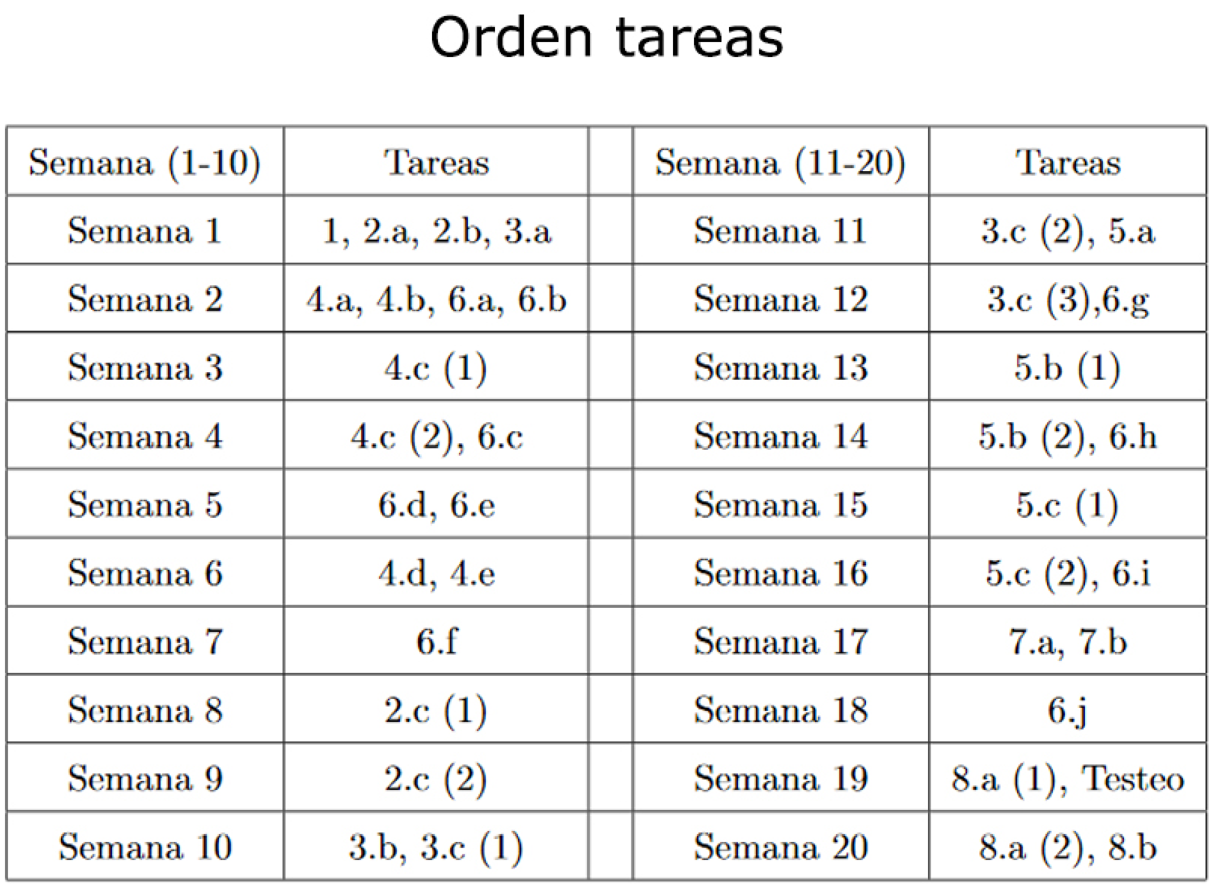
OBJECTIVES

- Achieve fluency in combat and animations
- Design a game with a start and an end.
- Implement a demo with 15 minutes of play.

Tasks

1. GDD (6h)
2. Design game's narrative (36h)
   a. History of world (2h)
   b. History of the game (4h)
   c. Specific dialogs of the game (30h)
3. Game design(48h)
   a. Define mechanics (2h)
   b. Design tutorial level(8h)
   c. Design rest of levels (38h)
4. Visual design, models and animation of characters (53h)
   a. Define art style (1h)
   b. Design and model of the main character (8h)
   c. Animate main character (24h)
   d. Design and model of other characters (8h)
   e. Animate other characters (12h)
5. Set design (47h)
   a. Define art style (1h)
   b. Design and model a standard set(24h)
   c. Design and model objects and assets of the game (22h)
6. Programmation(82h)
   a. Program player movement (4h)
   b. Program camera movement (2h)
   c. Program combo system (12h)
   d. Program weapons (12h)
   e. Program characters animation (4h)
   f. Program enemy AI(12h)
   g. Program NPC AI (6h)
   h. Program others (6h)
   i. Program tutorial level (8h)
   j. Program other levels (16h)
7. Sounds (8h)
   a. Research and introduction of the game music (4h)
   b. Research and introduction of the game sounds effect (4h)
8. Project presentation (20h)
   a. Creation of the memory (16h)
   b. Design project defense (4h)
   c. Defend the project (20 min)

Order to do the tasks [Fig.1]



Orden tareas

Semana (1-10)	Tareas		Semana (11-20)	Tareas
Semana 1	1, 2.a, 2.b, 3.a		Semana 11	3.c (2), 5.a
Semana 2	4.a, 4.b, 6.a, 6.b		Semana 12	3.c (3),6.g
Semana 3	4.c (1)		Semana 13	5.b (1)
Semana 4	4.c (2), 6.c		Semana 14	5.b (2), 6.h
Semana 5	6.d, 6.e		Semana 15	5.c (1)
Semana 6	4.d, 4.e		Semana 16	5.c (2), 6.i
Semana 7	6.f		Semana 17	7.a, 7.b
Semana 8	2.c (1)		Semana 18	6.j
Semana 9	2.c (2)		Semana 19	8.a (1), Testeo
Semana 10	3.b, 3.c (1)		Semana 20	8.a (2), 8.b

Figure 01

Figure 01: Tasks order.

EXPECTED RESULTS

The expected result for this project is a 15 minutes demo in which all the work done during the course is shown, along with a memory that corroborate it.

It is expected to get a fluid gameplay that invite the player not to leave the fight, though the life counter of his character was very low, with mechanics that incite to that. Also it tries to have different mechanics for each type of weapon accessible in the game. The number of weapons will be a total of three: a sword, an axe and a spear. It is also to expected have 2 bosses, one being weaker than the other. The controls will be comfortable and intuitive and that feeling will be maximized with a very accessible and well-presented tutorial level. Finally, the project will have two game modes, one focused on history and other focused on a battle arena, which will generate enemies in waves, so that the player practices against them and get as far as the player can. The story will be short and concise.

In the artistic section it is expected to have an accessible style to all the public. The number of animations in the main character is going to be very high, giving the character great dynamism and realism. The rest of the characters will not have such a high number of animations, but they will also have a high number. Sounds will be specific to each weapon and space.

TOOLS

Unity 3D: Will be used as game engine

3D Max: It will be used as a modeling and animation tool for the characters.

Photoshop: It will be used as a tool to create textures and the UI.

The game is going to be programmed in the C # language.

REFERENCES

[1] Unity 3d. https://unity3d.com/es, February2017.

[2] 11 bit studios. This War of Mine. https://www.youtube.com/watch?v=OQQw3pZACfk, February 2017.

[3] Adobe. Adobe Photoshop CS6. http://www.adobe.com/es/products/photoshop.html, February 2017.

[4] AutoDesk. 3d max. http://www.autodesk.es/products/3ds-max/overview, February 2017.

[5] Nintendo. Metroid: Zero mission. https://www.youtube.com/watch?v=7UnKzWIHioE, February 2017.

[6] UJI. Grado en diseño y desarrollo de videojuegos. http://www.uji.es/estudis/oferta/base/graus/actual/videojocs/, February 2017.

2 DESIGN

ABSTRACT

This chapter contains the game design document of the developed game, including information about mechanics, art, some influences, sounds or mindset.

Gameplay

The gameplay is fast and dynamic, where the player fights against a lot of enemies with three weapons with differents combos and play styles: a sword, a spear and an axe. Weapons break easily, forcing the player to change continuously his mindset to adapt the new gameplay. To take new weapons the player have to defeat enemies. In their death, they will leave their weapons and the player will be able to take them if he/she wants.

To potentiate that dynamic combat the player and the enemies will have a lot of differents animations. That is other of the game pillars.

That gameplay is developed in an arena mode, a game mode that puts the player against a big number of enemy waves. The other planned mode is the story mode. In this, the player has to advance in a city and a factory to save both and complete the story. This mode isn't developed.

The arena mode is a mode in which the player is put into a kind of coliseum. The player has to defeat an infinite number of enemies waves. The waves are sent periodically.

The gameplay takes inspirations from many games. Beginning in the combo system, it was looked for games that had an optimized combat system. One of the games that helped most was *Skullgirls* [1] [Fig.2] by M2, a 2D fighting game that uses a fight system based on combos. But the influences don't stop there, for the development of the game it also can be named other games, like *Injustice 2* [2] or *Street Fighter* [3]. The objectives to have been watched these games was to learn about combo systems and about how to do that for the project. But what it was discovered was that it doesn't exist games with a combo system similar that it is needed for the project.

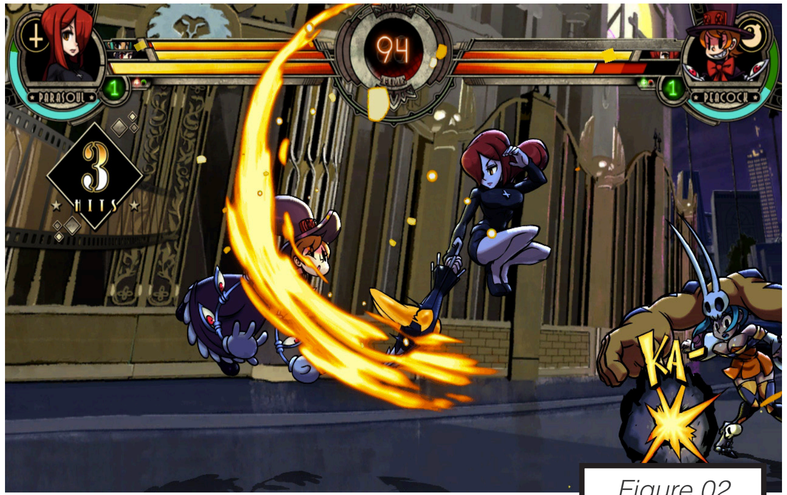


Figure 02

Figure 02: Gameplay screenshot from SkuyllGirls by M2.

Hollow Knight [4] is another influence for the project.It has a combat system that encourages the player to continue in battle rather than retreat. It gives the player some profits in life form if he/she continues fighting. Furthermore, in *Hollow Knight* [Fig.3] exists the concept of Arena too. In the Arena the player has the possibility to fight versus enemy waves. In the project the player have benefits in form of weapons and other buffs, like life or extra damage.

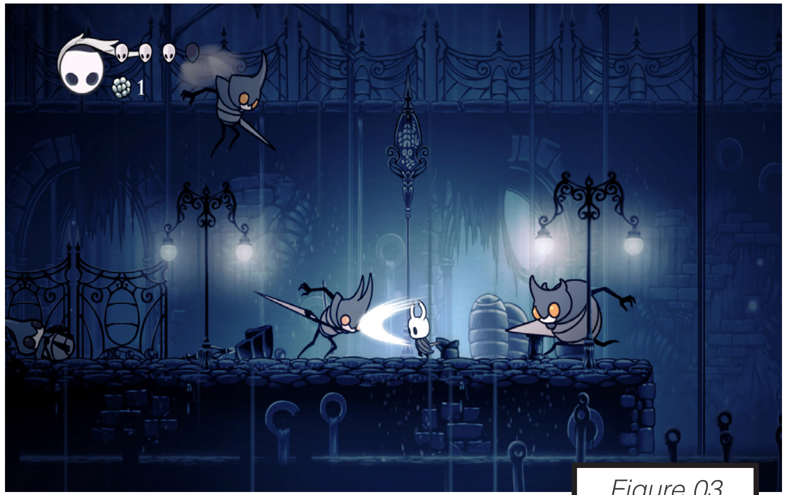


Figure 03

Figure 03: Gameplay screenshot from Hollow Knight by Team Cherry.

GAME DESIGN

The last influence of the project is *The Legend of Zelda: Breath of the Wild* [5], with its weapons system.

Figure 04

Figure 04: Gameplay screenshot from The Legend Of Zelda: Breath of The Wild by Nintendo.

But that is not a normal influence, because the idea of the weapons life in the project arrived before the *The Legend of Zelda: Breath of the Wild* [Fig.4]. Even so, their gameplay helped to define the final weapons system too. The difference between that game and the game of the project is the duration. In Breath of The Wild, the weapons have a considerable duration compared with the project weapons.

Mindset

While the player plays the game, he or she will have a sense of power. That is because the player will be fighting and winning a big number of enemies. At the same time, the player will be a bit stressed, due to the continuous weapons change.

TECHNICAL

Screens

1. Main menu
   a. Story
   b. Arena
   c. Controls
   d. Exit
2. Story
3. Arena
   d. Game
   e. Pause
6. Controls

Credits

Controls

The controls are predefined by the game. It have controls for keyboard and Xbox Controller, but it's preferable to play it with the Xbox controller. The player will be able to do some actions as:

- Moving: The player can move the character to the left or the right with the left stick or the arrows key.
- Jump: The player can make the character jump with the button X or the space key.
- Two attacks: The player can attack with the buttons A and B or the keys A and S. That buttons work to make combos.

Figure 05: From top to bottom: Example of dash, of parry and of take in the project.

Figure 06: Controls of the game, over a Xbox Controller. It can be seen better in the section "Big Figures".

- A dash: The player can make a dash with the button LB or the key Q. A dash is a fast and short movement of the player in one direction [Fig.5].

- A parry: The player can make a parry with the button RB or the key W. The parry leaves the player invulnerable for a few seconds

- Take weapons: The player can take weapons with the button Y or the key D. The take action is used to take the stage weapons.

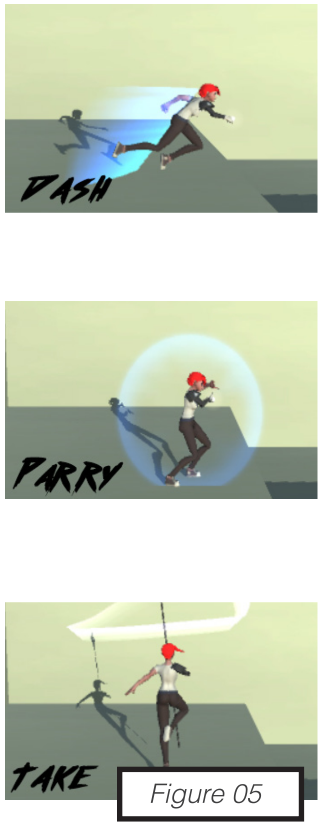


Figure 05

- Pause the game: The player can pause the game with the button Menu or the key P. In the pause the player can see the current possible combos.

With that actions the player could do the basic movements of normal games and a lot of combos. That is possible for the combinations of the attack buttons (A and B) with the movement directions (UP and DOWN).

The controls distribution in the Xbox controller is really thought-out too [Fig.6]. As can be seen in the figure all the movement options are distributed on the left controller side. With that, the player will know and learn really fast how to move in the game. Furthermore the interactions with the environment are in the upper right controller side, and the fight buttons are in the most easy place to use with the right hand. With that distribution the controls are telling the player he/she have to fight.

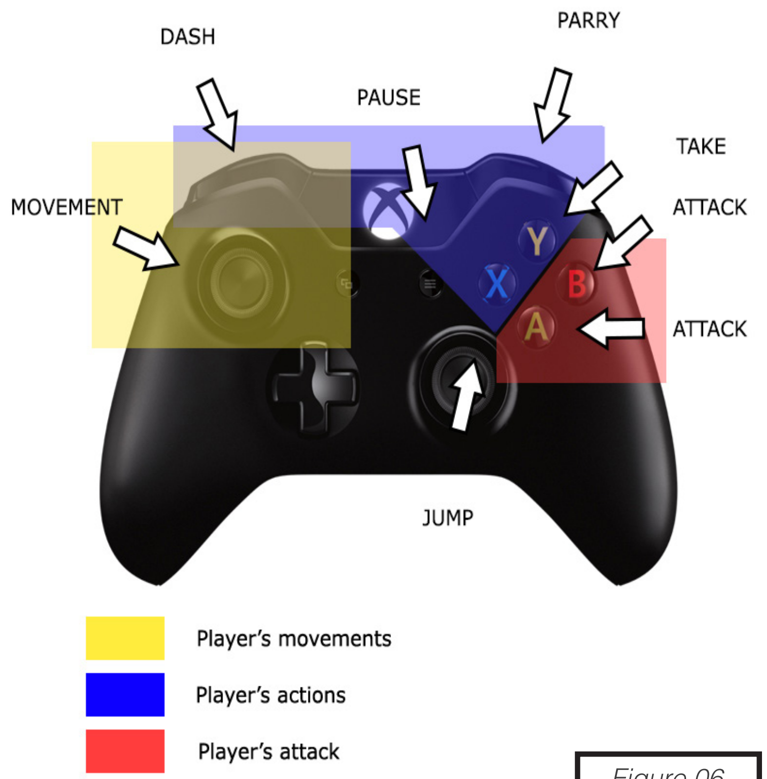


Figure 06

Mechanics

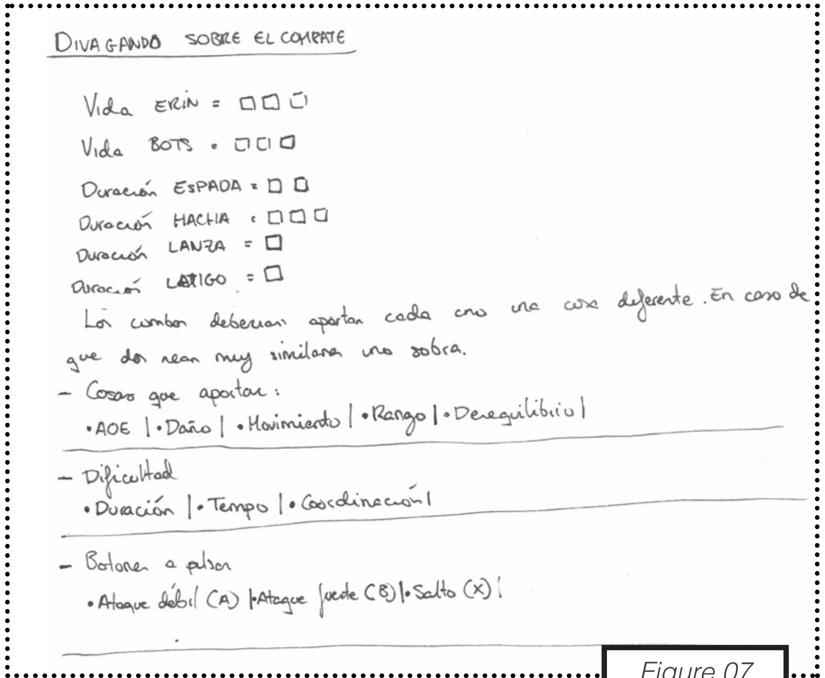
There are some key mechanics in the game.

The first is a combo based combat. In it, the player can do different combos with every game weapons. All the weapons have different play style and combos. The combo system in the game works like a tree of actions. The player can begin a combo and decide in the middle of it if he wants to go in one way or another. Every part of a combo is called “combo step” (comboAttack in the programming section). That allows the player to adapt his/her actions depending of the environment. Furthermore, if the player feels threatened, he/she can use the parry action to dodge the danger and he/she will have an additional special combo and some extra milliseconds to continue the combos. Talking about combos, these depend of some factors. First of all, the previous action. The previous action opens the player other actions. Second, the buttons combination and the timing with it. All these characteristics make the combo system really solid and with a lot of possibilities. Finally, the damage sended to the enemies (or to the player, that works in both directions) works with the animation. That means that the damage is only executed if the weapon collides with the objective and it will collides if the player, or the enemies, make the correct animation in the correct position.

The weapons have an interesting mechanic too. Additional to the different gameplay, it will be talked about it later, the weapons have their own life [Fig.7]. The lifes represents the weapons durability. If that life is equal to zero the weapon is destroyed. The amount of life is very small for all weapons, and that reinforces the mechanic of the player adaptation. The weapon lose a life every time the combo is broken. The combo can be broken by several reasons: to finish a combo, to fail the buttons or combos tempo or to get damage in the middle of a combo.

The player and enemies work with a life classic system. Both of them have some lives and they lose one or more of them every weapons hit. If the lives arrive to zero in enemies case, they leave their weapons to the player. And in the player’s case, the game finish.

The weapons left by enemies are “Free Weapons”. These objects are floating in the air and turning on themselves. If the player are near one of it, the free weapon render a line between them and the player will be able to use the object. If the player take it, the free weapon disappear and the player will be equipped with the free weapon.


DIVAGANDO SOBRE EL COMBATE

Vida ERIN = □□□

Vida BOTS = □□□

Duración ESPADA = □□

Duración HACHA = □□□

Duración LANZA = □

Duración LATIGO = □

Los combos deberían aportar cada uno una cosa diferente. En caso de que dos sean muy similares uno sobra.

- Cosas que aportar:
  - AOE | • Daño | • Movimiento | • Rango | • Desequilibrio |
- Dificultad
  - Duración | • Tempo | • Coordinación |
- Botones a pulsar
  - Ataque débil (A) | • Ataque fuerte (B) | • Salto (X) |

Figure 07

Figure 07: Concept of the lives of the player, enemies and weapons.

Figure 08: Concept of the sword combos.

Weapons gameplay

As was said previously, every weapon has different combos and gameplay. To do a combo step the player should press a minimum of one attack button and it could have additionally a direction with the left stick. The gameplay of every weapon is described in the next lines.

- **Punch**

The player can be without weapons. In this situation the player have to fight with her own hands. That “weapon” doesn’t have a real mechanic or gameplay. It only have one combo and it’s thought to get another weapon.

- **Sword**

The sword is the basic weapon of the game. It have some easy combos and some difficult combos. The weapon life is high, with 3 points of life. This makes the sword the essential weapon to begin with. It has the smallest range.

The damage of every combo step is low, but it is incremented if the combo continues. That simple mechanic potentiates the idea of finish the combos. The sword has 5 basic combos and one extra combo with the parry [Fig.8]. Every combo have a special detail that distinguishes them from others:

1. The first is really easy to do, with only press the principal button. With an easy rhythm it’s done and it’s a buckle combo (that means that the combo can do it as long as the player like).

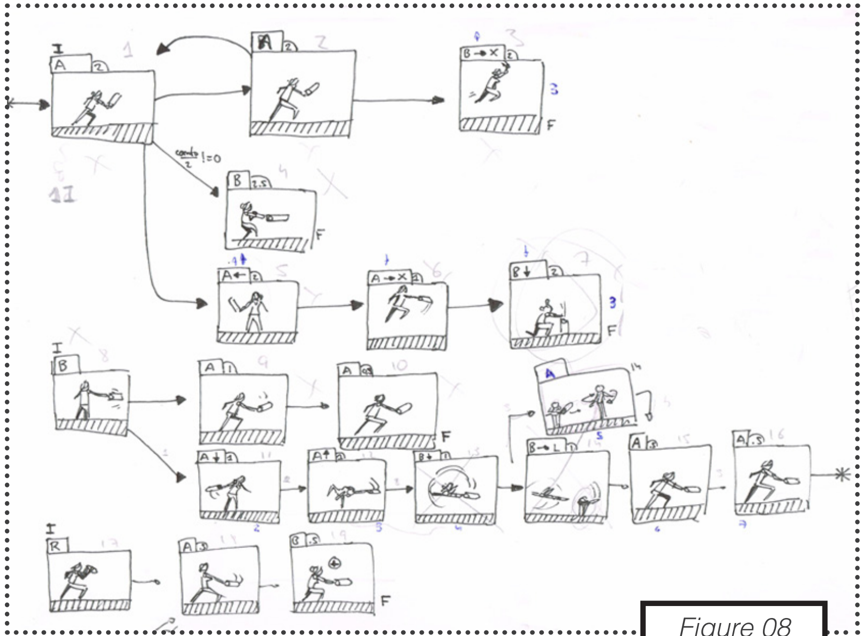


Figure 08

   And furthermore, it have a finisher, with high damage to the enemies.
2. The second one is easier yet. It only have two steps and is really accessible.
3. This combo allows to attack backwards and return to the front to kill the first enemy.
4. This combo is really easy too. But this time it doesn’t begin with the button A, it start with the button B.
5. The last combo is the most difficult. It is long and with a big increment of the damage in all the steps of the combo (6 steps). It allows to attack in all directions, but is really complex.
6. The special combo can be made when the parry is used in the middle of a combo. If it is done correctly increases player’s health by one.

TECHNICAL

Figure 09: Concept of the axe combos.

- **Axe**

The axe is slow but with high damage. It has easy combos but with few steps. That forces the player to hit the combos. The weapon has 4 points of life. It has the characteristic that its attacks are based on the button B, unlike the sword that made it in the A.

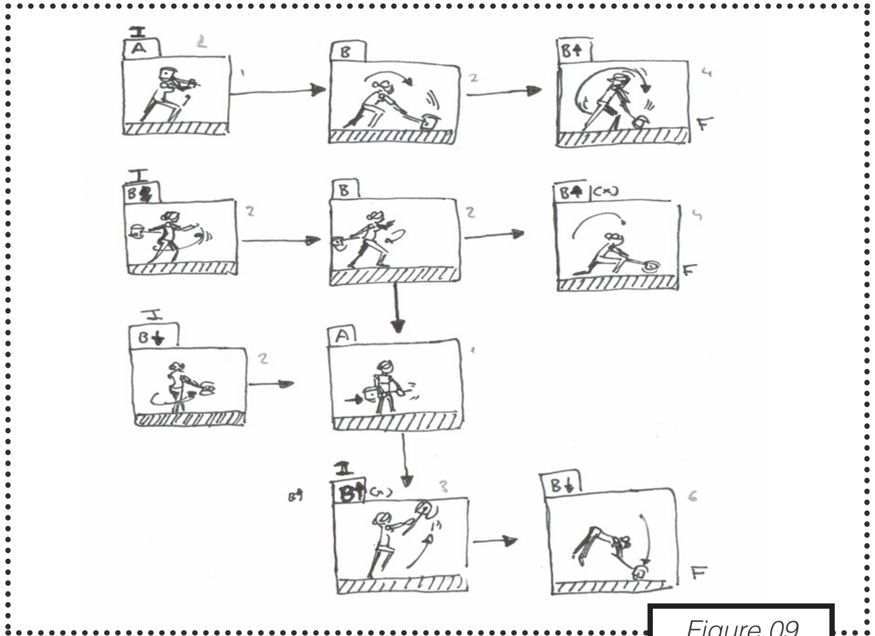


Figure 09

The axe has four combos and a special combo with the parry [Fig.9]. Every combo have a special detail that distinguishes them from others:

1. The first is easy and intuitive for the player, but it's not really strong. It begins with the button A.
2. The second is the basic combo of the axe. It has 2 ways to advance. The first is a finisher similar to that found in the previous combo. The other is a part of next combo.
3. That combo allows to attack in all the directions and finish with a basic attack without a lot of damage (that can be done with the previous combo too). Furthermore, you can continue it with the next combo.
4. The last combo has the stronger damage of all the game and is really easy, but there's a lot of time in which the player will be unprotected.
5. The special combo can be made when the parry is used in the middle of a combo. If it is done correctly increases weapon's health by one

- **Spear**

The last weapon is the spear. It only has one point of life and it doesn't do much damage, but it has a lot of range and only two combo. One of them is an infinite combo. This weapon is, in addition, really fast doing damage and with its special combo the damage of the weapons is duplicated.

As stated above, the spear only has two combos.

1. The main combo has differents ways and possibilities that allow the player to follow the combo as needed.
2. The second and last combo isn't really usefull, but it has a characteristic higher jump, ideal to explore new paths in the map.
3. The special combo can be made when the parry is used in the middle of a combo. If it is done correctly multiply weapon's damage by two.

ARTIFICIAL INTELLIGENCE

The artificial intelligence of the game is focused on the robots. It is really simple but effective, thanks to the amount of combos that has and the different weapons. That gives the player the feeling of diversity.

The robots have five states: Sleep, Stop, Move, Fight and Death.

The first state happens when the player is far of the robot. It doesn't do anything, only wait. The second state happens when the robot is still preparing to attack. Is the best moment to attack him. The robot enters in Move state when it goes toward the player. It stops always in front of the player. The distance between the player and the robot depends of the robot current weapon's range. When the robot is in fight mode is the moment to go back for the player, because the robot is ready to attack with a random combo of its repertory. It can't move during this state. The last mode is Death, and, obviously, represents when the robot is death. When that happens, the robot spawn its weapon as free weapon. It means that the player can take the robot weapon now.

The robot combos are different for every weapon [Fig.10]. Every combo has some common actions: Anticipation, action and pause. The anticipation tells to the player than the robot will attack soon. The action is unstoppable and it is when the enemy is more dangerous. In the pauses the player can attack the enemy. If the player do it the pause becomes longer.

The robot weapon also works as a way to show the zone or wave difficulty. That is because every weapon has their own combos and ranges. It makes easier to beat a robot with a sword than a robot with a spear.

Figure 10: Concepts of the Artificial Intelligence combos.

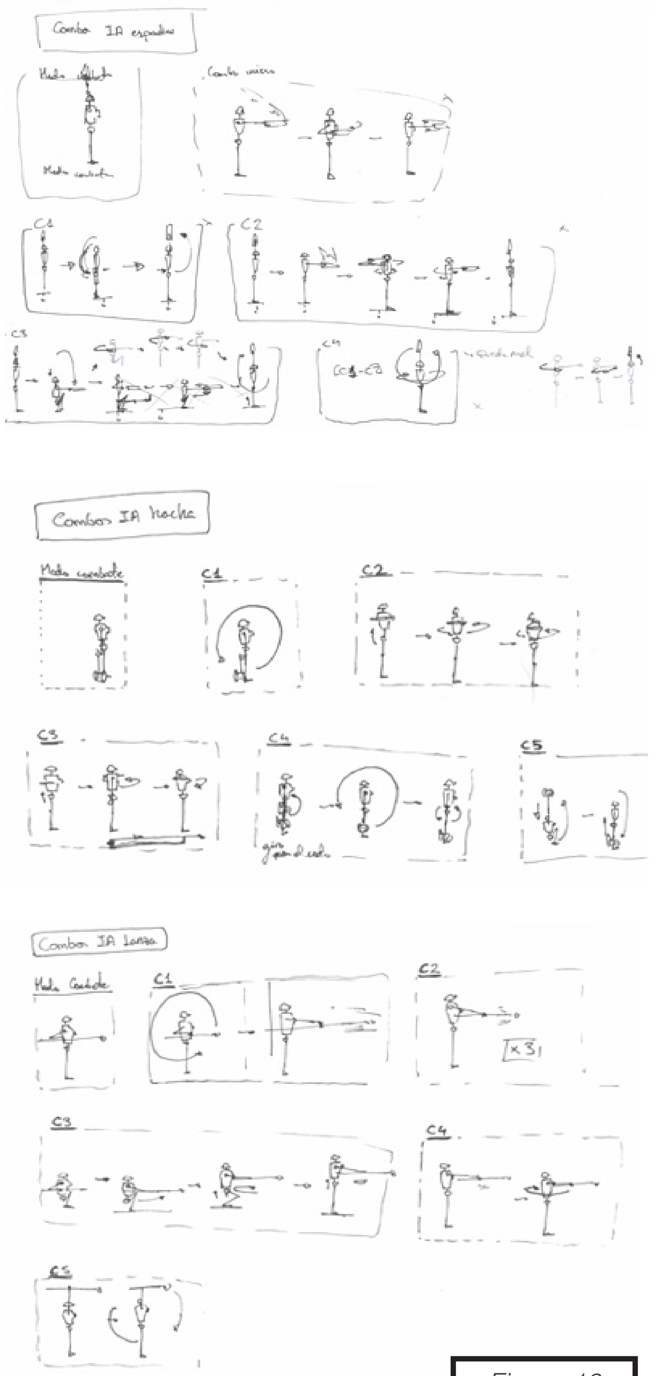


Figure 10

ARTIFICIAL INTELLIGENCE

In the arena mode there is a wave controller. It controls when enemies appear. It is designed under some rules:

- There will be 4 spawn points in the environment.
- If the player kills all the enemies on the set, the next wave will begin in five seconds.
- After X seconds of the current wave last enemy appears the next wave begins. That is as maximum. Being X the next formule result.

$$\boxed{X = 3 * \text{number of enemies of the current wave}}$$

- If this number is less than 20 seconds, this time will be 20 seconds.
- The first wave creates a simple robot equipped with a sword.
- At the beginning a free weapon sword appears to the player.
- In the next nine waves the enemies number on the waves increment.
- After ten waves, the tenth wave is repeated but the enemies appear with random weapons.

NARRATIVE

The story of the game is as follows:

Erin, the main character and who is controlled by the player, arrives to Dao, a military city. Dao is involved in an investigation relationated with the input of military tactics on a number of robots and an artificial intelligence (AI) who controls them automatically. The last days, the city Major detected suspicious action from the AI and calls out to the The Triangle organization (a kind of United Nations in the world of the game). Erin is sent by The Triangle and she has to discover and repair the problems in Dao. The city have a big factory.

During the game, Erin goes into to the city sneaking in the factory, because the doors of Dao were closed. This is the first level of the game. When she goes through the zone, Erin arrives to the city. She, and the Player, begins to know the city and the story of the game. The Major recommends Erin to talk with the AI center. She takes the big factory's elevator and goes to talk to the AI. It confirms the rebellion and throws Erin to the City again. But now, the city is under the robots' chaos and destruction. Erin returns to the AI's room finding other ways and she fights and defeats the AI in a final fight.

The arena mode happens in the second visit to the elevator, this time accompanied by some non friendly robots.

LEVEL DESIGN

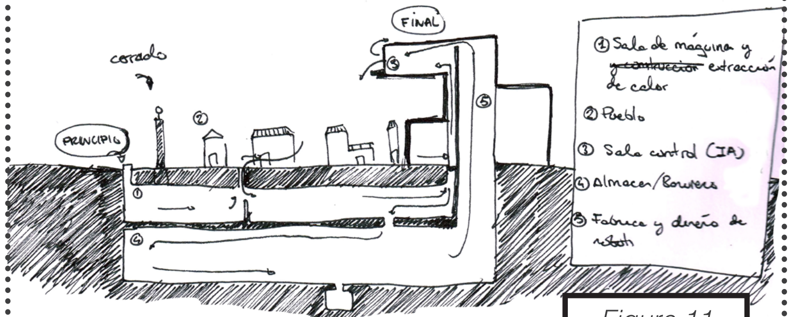

FINAL
cerrado
PRINCIPIO
① Sala de máquina y ~~y contrucción~~ extracción de calor
② Pueblo
③ Sala control (IA)
④ Almacen/Bowers
⑤ Fabrica y diseño de robots

Figure 11

Figure 11: Concept of the DAO mapping, the city of the game.

Machine room (I)

It has a metallic look, cluttered and dirty [Fig.12]. This first level function is to work as a tutorial to the player. It teaches basic controls and combos and the player learns how to fight versus robots. They are strategically positioned to reinforce the player's learning curve. In this case the robots aren't fighting for the rebellion, they fight because the player is an intruder in the factory.

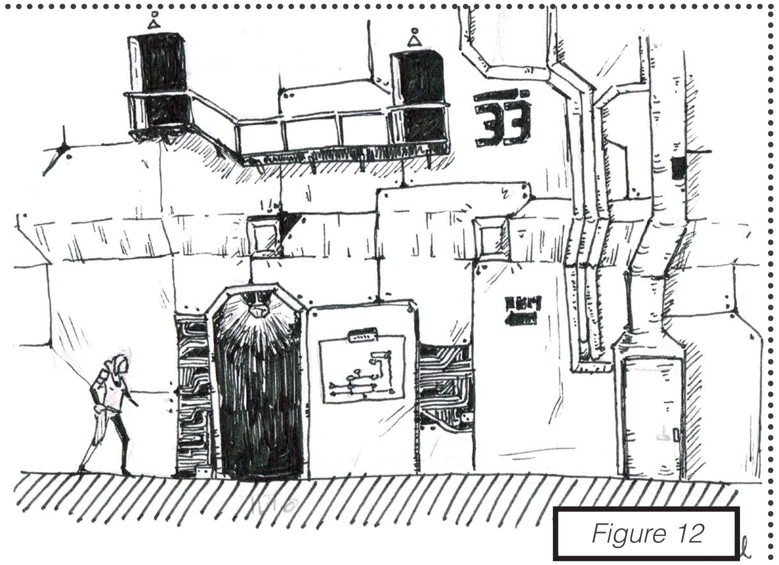


Figure 12

Figure 12: Concept of the Machine room level.

City (I)

Clean and military look [Fig.13]. The level aim is to present to the player the game story, and learn some combos.

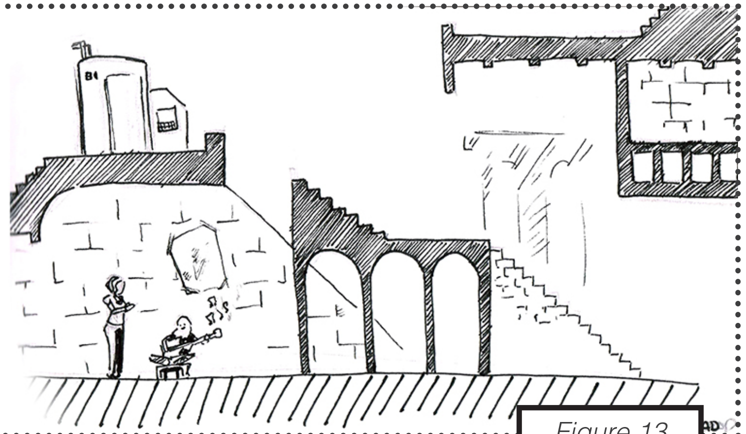


Figure 13

Figure 13: Concept of the City level.

Control Room (I)

Clean and white look. A simple room with the AI, where the rebellion begins. The objective is present the final boss and carry on with the story.

City (II)

The same city as before but, this time, it's destroyed and chaotic. In this level the player will fight versus some robots in a known territory, the city. The both cities huge contrasts will involve the player in the game story.

LEVEL DESIGN

Machine Room (II)

Metallic, cluttered and clean look [Fig.14]. A classic game level, with some robots to fight.

Elevator

White, empty and clean aspect. This level function is to put the player in a little room with a lot of enemies spawning to fight against her.

Control Room (II)

The same room as before. This room is to fight versus the final boss.

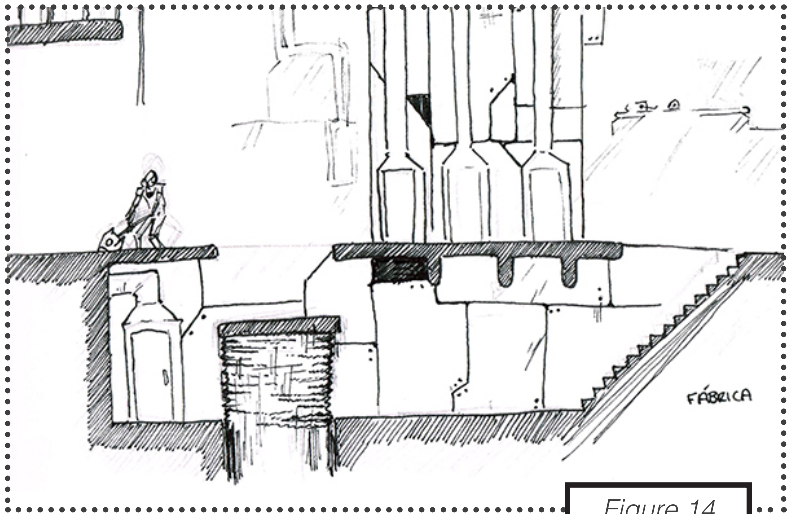


Figure 14

Figure 14: Concept of the Machine room (II) level.

GRAPHICS

In the artistic section, the game takes influences from many games, for example, it was influenced by *This War of Mine* [6] [Fig.15] by 11 bit Studio and *Assassin's Creed Chronicles China* [7] [Fig.16] by Ubisoft Montreal. These influences helped to define the game as a 2.5D game and not as a 2D game. And their animation style and scenarios inspired a lot of the models, animations and environments of the project. Their realistic style, their particles and animations helped to choose the course of the project in the graphical style.

The game happens in a military city and in a factory. That forces a metallic colour palette, without natural environments. Furthermore during the game reign the straight angles, been the main character the source of curves.

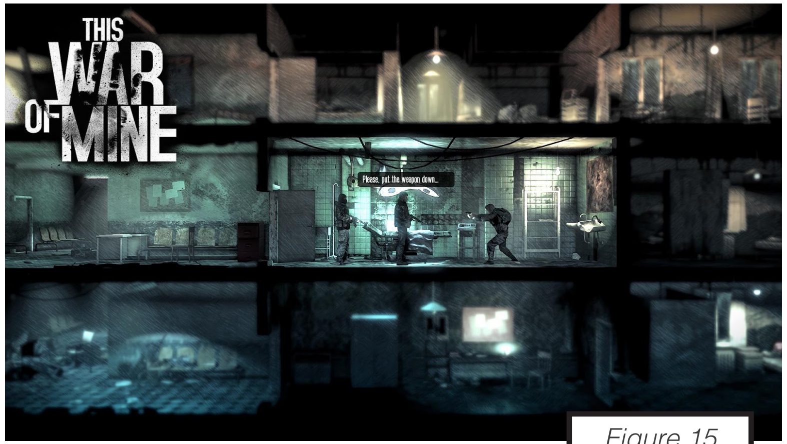


Figure 15

Figure 15: Gameplay screenshot of This War of Mine, by 11 bit Studio.

The style will be realistic, but simplified, with a lot of plane colours and some textures.

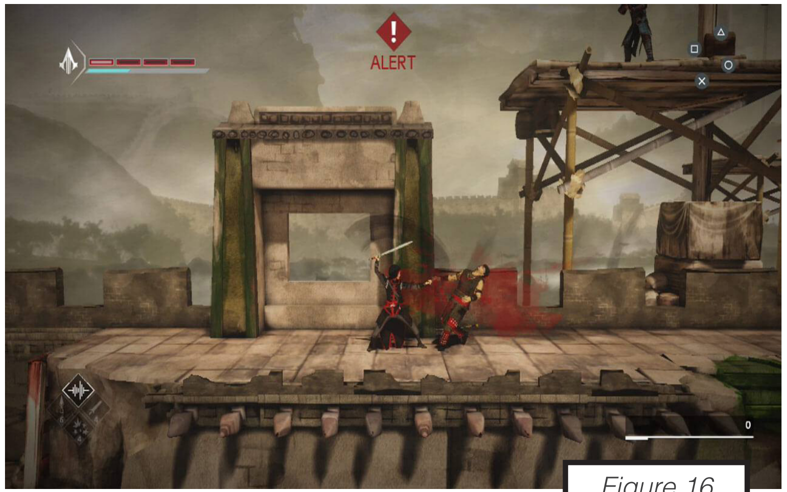


Figure 16

Figure 16: Gameplay screenshot of Assassin's Creed Chronicles China by Ubisoft Montreal.

The interface will be simple and uncluttered [Fig.18], with only vital information: the life or the player, the life of the current weapon and the current attacks on combo. Furthermore, the enemies will have over them an indicator with their current life. This interface is inspired in the game *Hyper Light Drifter* [8] [Fig.17] by Heart Machine.

The game will have many particles to do more readable the actions in the screen. Will be the next:

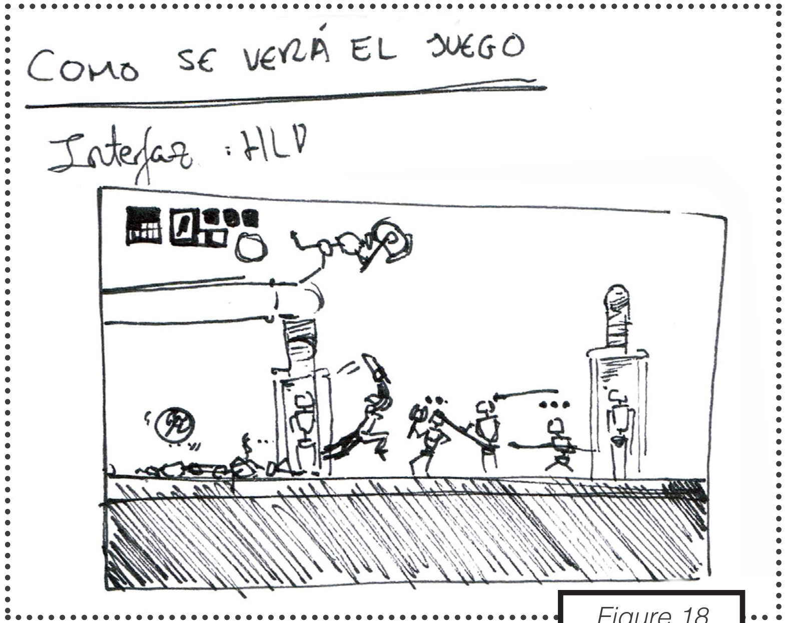


Figure 18

Figure 18: Concept of a theoric game screenshot.

- Hit of the weapon particles.
- Dead of a player weapon particles.
- Dead of the player particles.
- Dead of an enemy particles.
- Particles for the weapons movement when they can hit.
- Dash particles.
- Parry particles.

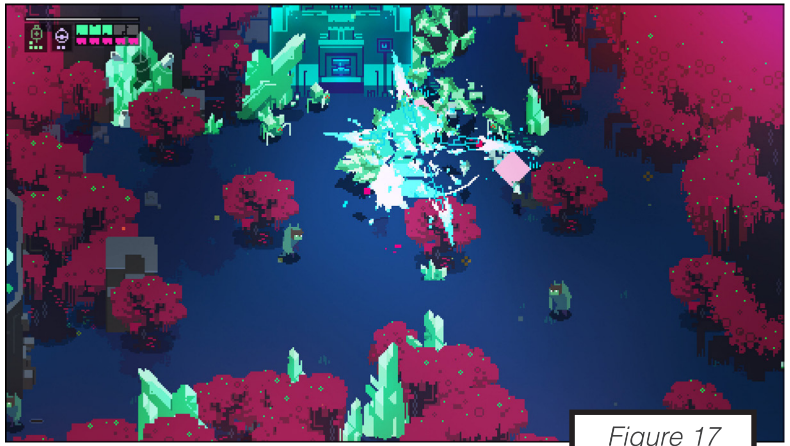


Figure 17

Figure 17: Gameplay screenshot of Hyper Light Drifter by Heart Machine

GRAPHICS

As it has been said previously, the animations are one of the game pillars. The player will have a different animation for every combo and action. Furthermore, the idle animation and the run animation are different for every weapon. In addition, the idle state has an extra animation if the player remains a lot of time without fighting. There will be, in total, more or less 29 animations only for the player (counting a combo as

GRAPHICS

one and not as the number of steps in the combo). The robots will have a lot of animations too. One different for every combo and action. And, again, a different animation for walk and idle depending of the weapon. There will be, in total, 25 animations only for the robot. That means that the game will have a total of 54 animations for the characters, counting a combo as one.

SOUNDS AND MUSIC

The sounds and music haven't got an especial inspiration. That is because the game developer doesn't have a lot of music idea. All the music will be taken of online free pages of effects and sounds like Newgrounds [9], Freesfx [10] or Freesound [11]. That means that the music aspect will not highlight in the game and will stay only to fill the game silence.

The game will have some SFX to clarify the actions in the game, similars to the particles:

- Hit of the weapon sound.
- Dead of a player weapon sound.
- Dead of the player sound
- Dead of an enemy sound.
- Sound for the weapons movement when can hit.
- Pauses sound.
- The game will have a simple music song.

3 DEVELOPMENT

ABSTRACT

This chapter will explore the game development.

It will show the planification changes, the programmation of the characters, AI, weapons and scenes, among others. The project changes a lot during its development and in this chapter it will see why.

Remember that in this chapter isn't included the art or animation development. It will be explained in the next chapter.

DESTROYING THE ORIGINAL PLANIFICATION

In the technical proposal it showed an optimistic planification and objectives [page 10]. All that changed a lot in the first month of developing. The hopeful duration of the tasks in the project fastly where broken seeing the time wasted in the modeling and animation of the main character, exceeding a lot the time planned. This fact forced to change development plans and reconsider the project objectives.

With that in mind, the original plan totally changed: Now the objectives are as follow :

- Achieve fluency in combat and animations.
- A playable demo.
- A game with comfortable and intuitive controls.

As can be seen, the goal is not longer get a game with narrative, with a beginning and an end. That is because the developing of the scenarios and their assets, like NPCs or checkpoints, were going to steal a lot of developing time, and they were not going to potentiate the combat mechanics or the game animations. That was the reason to change the idea to do an arena mode instead of a story mode. The demo is still an objective, but without the 15 minutes condition. The new objective for the demo is to be playable.

With there new objectives and the new perspective of the task timing, the planification changed. Now, 30 hours of the narrative design (2.c) and 46 hours of the playable design (3.b and 3.c) had to be distributed. 50 hours of that goes to the animation part. The rest to the programming part for emergencies. To advance in the game and know if the gameplay was good the combo system programmation was converted in a priority and this task occupied more that the planned hours, due to its continuous adaptation.

The new planification was not fulfilled. The first problems with the combo system, the modeling and animation of the main character, the lack of pressure to deliver the project and the external practices of the developer modified a lot the time to invert in the project. Furthermore, the vacations like Fiestas de Magdalena, Pascua or the end of school trip cut a lot the dynamic of work during the developing of the project. All of this converge in a chaotic last month with a lot of time used in the game and not enough sleeping.

All of this really doesn’t affect in the total hours used on the project. There are probably more than 300 hours and it is the best project I could do in these conditions, but the final stress probably has affected negatively in the game project.

Failure to carry out with planning has been the biggest project mistake.

ORIGINAL STORY AND MECHANICS AND THEIR EVOLUTION

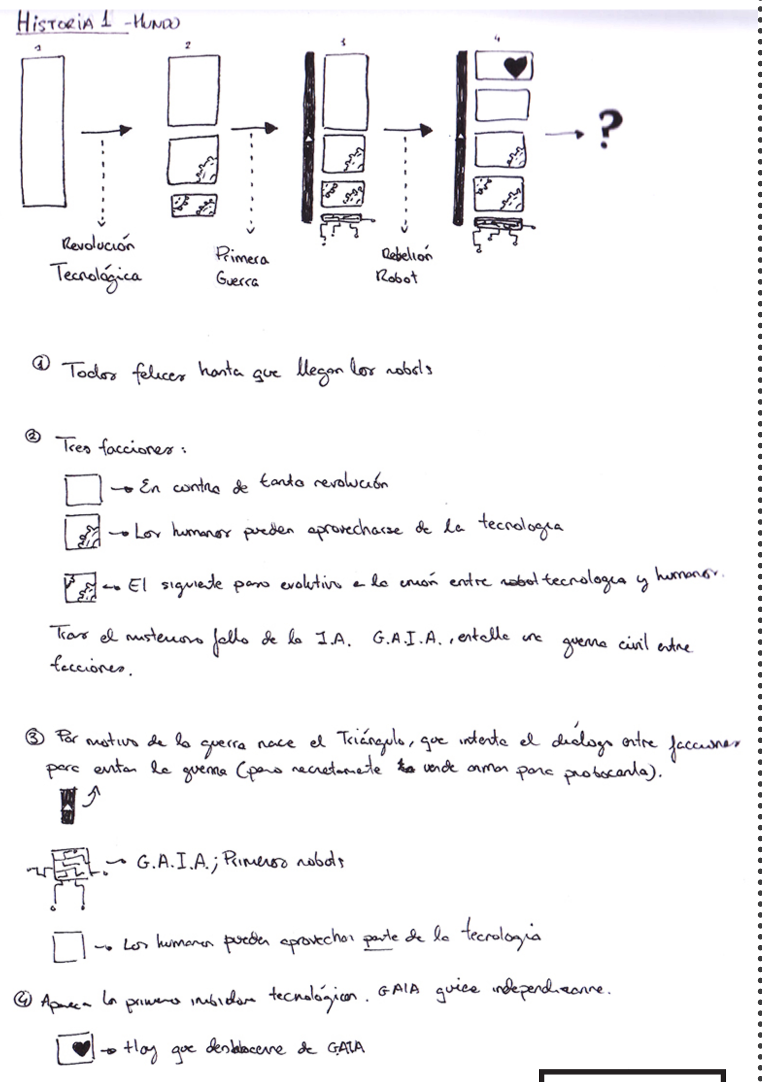
The story changed a lot between the first thought (December 2016) and the final result.

First of all to say that in the game was always prioritized gameplay before the story, so the story was a mechanic slave.

In the project really first idea the game will have the final dynamic combat with, additionally a stealth mechanics and a mechanic with the differents tribes relations in the game. The stealth was not going to have a new idea for the videogame world, but it, with the relation mechanic, was going to have an interesting result. In concept, every tribe have a relation with the other tribes, positive or negative. That means that if the tribe 1 is friend of the tribe 3 and enemy of the tribe 4, if the first see someone in his territory of the tribe 4, he will attack him and probably the tribe 3 NPCs will help him. With that map of relationships, the player will be able to steal the dress of every enemy to simulate been someone of a tribe and create chaos or be unnoticed in the differents game areas. Furthermore, every tribe was going to have their own culture, with their own costumes, sets, weapons and fight style.

With all that in mind the game's story was going to be the next [Fig.19]:

In a region of the world, a pre-robotic country live in a normal peace. But then a technological revolution arrives. That new technologies was an inclusion of the robots in the diary life. The biggest part of the robots was under the control of a central A.I. called G.A.I.A. It was a great advance for the world, but not everyone thought the same. It divided the nation in three factions: The first, contrary to the robotic revolution. The second thought the robots were a breakthrough. And the third thought with a real union between humans and robots creating cyborgs. These three differents thoughts united to a mysterious G.A.I.A fail, over time, ended up with the First War between factions, a civil war.



① Todos felices hasta que llegan los robots

② Tres facciones:

→ En contra de tanta revolución

→ Los humanos pueden aprovecharse de la tecnología

→ El siguiente paso evolutivo a la unión entre ~~robot~~ tecnología y humanos.

Tras el misterioso fallo de la I.A. G.A.I.A., estalla una guerra civil entre facciones.

③ Por motivo de la guerra nace el Triángulo, que intenta el diálogo entre facciones para evitar la guerra (pero secretamente ~~la~~ vende armas para provocarla).

→ G.A.I.A.; Primeros robots

→ Los humanos pueden aprovechar parte de la tecnología

④ Aparece la primera inhibidora tecnológica. GAIA quiere independizarse.

→ Hay que deshacerse de GAIA

Figure 19

Figure 19: Concepts of the story first version.

[Subtractive design is the process of removing imperfections and extraneous parts in order to strengthen the core elements. Make everything as simple as possible, but not simpler]

Figure 20: Concepts of the vehicles to travel between cities.

ORIGINAL STORY AND MECHANICS AND THEIR EVOLUTION

That war wasn't really big. It fastly ended with the creation of an organization called The Triangle. That was a mediating organization to avoid conflicts between tribes. But it had an occult intention, related with the sale of weapons. It really wasn't resolving conflicts, it was creating conflicts. Furthermore the country gobern was broken and it began to work like the classical Greece, with state cities, controlled by the factions [Fig.20]. In addition, G.A.I.A. began to have its own thought and, bit by bit, it began to want their rights.

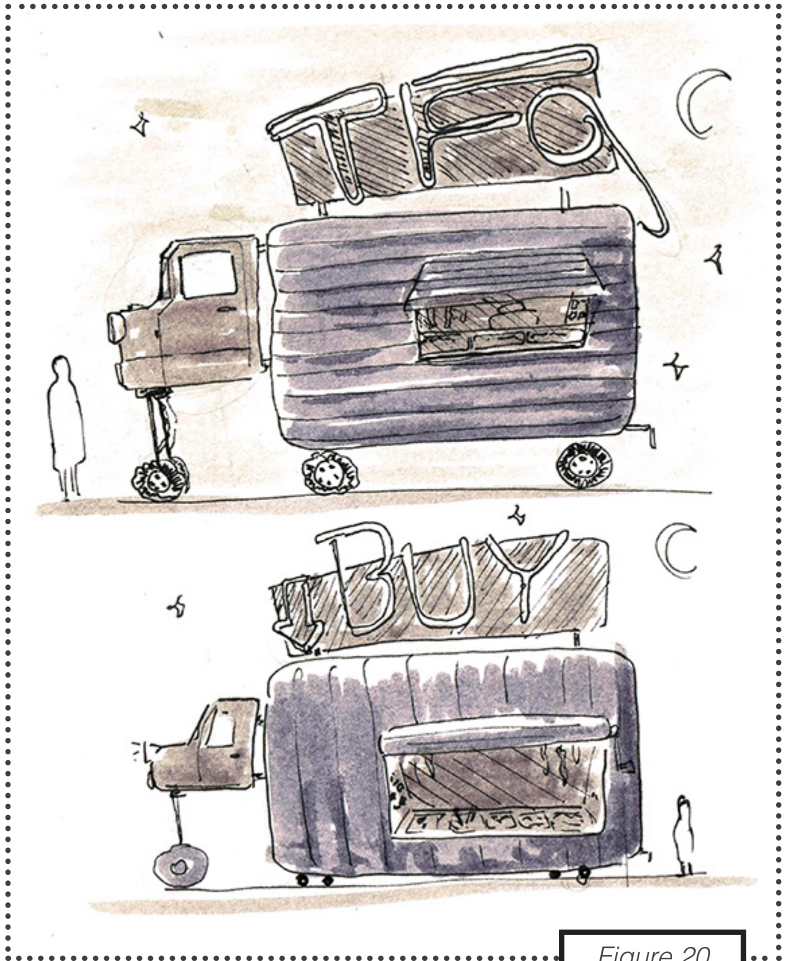


Figure 20

All that bring them to the next step in their world: The Robot Rebellion. After that rebellion, G.A.I.A form its own faction with their intereses. Furthermore the most classical faction were divided in two. One that thought in the little utilities of the machines. And the other that had an amish thought, totally against G.A.I.A.

In that point the game begins. Every faction has its own relations, culture, identifying colour and form. In the next figures can be seen some concepts.

Every tribe was going to have their own city and the player will be able to travel between them too.

But with the arrival of the technical proposal (February 2017), the planification showed that there was too much content in the game to be performed in only 300 hours. In this situation the project begins a "subtraction design phase".

The stealth mechanic was deleted for one main reason, it creates a dissonance with the fight system. The fight system was thought to encourage the player to fight and it's exactly the opposite function of the stealth mechanic. That forced to deleted the stealth mechanic. With that out, the tribes relation was totally unnecessary and was deleted too. Without tribes, the cities wasn't necessary, neither the cities or the dresses. Without that and with an alone main mechanic, the combo fights, the current game story had to change. Now the story had to be more simple, only as a background. This finished with the final game story written in the previous chapter [page 23].

ORIGINAL STORY AND MECHANICS AND THEIR EVOLUTION

During the game, Erin goes into to the city sneaking in the factory, because the doors of Dao were closed. This is the first level of the game. When she goes through the zone, Erin arrives to the city. She, and the Player, begins to know the city and the story of the game. The Major recommends Erin to talk with the AI center. She takes the big factory's elevator and goes to talk to the AI. It confirms the rebellion and throws Erin to the City again. But now, the city is under the robots' chaos and destruction. Erin returns to the AI's room finding other ways and she fights and defeats the AI in a final fight.

Now all the game happens in a simple city without differents cultures. To include the weapons mechanics was introduced the concept of combat robots. The possible discussion of the importance of technology and its good use was, sadly, eliminated. But with the only implementation of the Arena mode the story really doesn't matter and it is only a guide to the art and conceptual development. The context is one of the most important section on a game.

COMBO SYSTEM

The game has had always three base weapons: a Sword, an Axe and a Spear. Their gameplay idea don't change a lot at the pass of the time, the sword is standard, the axe is slow and the spear is fast. But it was a common idea to put a fourth weapon. Several weapons were thought, like a pistol, a whip or daggers. Finally all of them were canceled because they didn't give a characteristic gameplay. The final weapons are the three previously named and the player's fits. The weapons mindset didn't change a lot from the beginning, but the possible combos changed [Fig.21]. That is because the first combos were thought with the old mechanics. Furthermore, originally the combos have more buttons to activate them (not only A, B and Up and DOWN). Originally the combos will be able to activate with all the directions in the left stick, the jump button, the A and B and the special actions, the dash and the parry. Finally all that was simplified to give more accessible gameplay to the player.

Figure 21: Evolution of the axe combos. It can be seen better in the section "Big Figures".

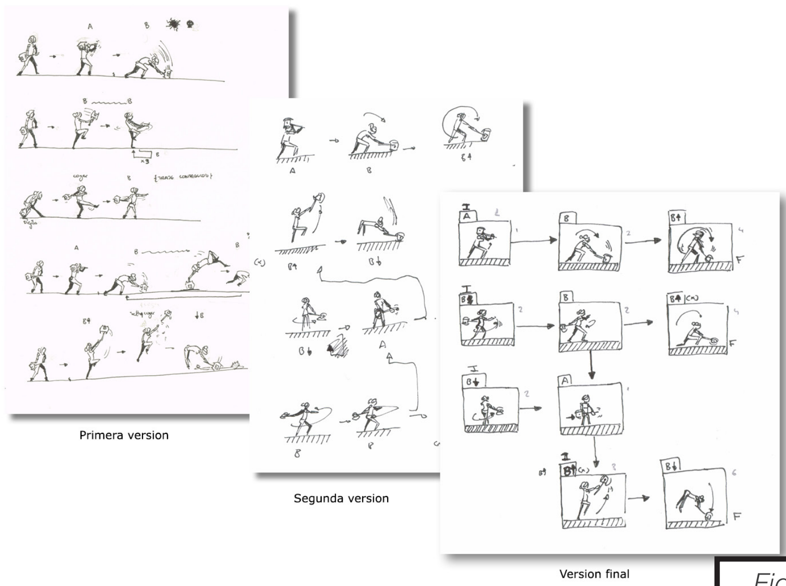


Figure 21

The operation of the combos system works through four scripts: the weaponSystem script, the ComboManager script, the ComboSystem script and the animationCombo script.

ComboManager

The ComboManager script contains the base of the combos, and the class comboAttack, which is the combo step. Only one object has that script, the GameManager. In that script is defined all the combos of all the weapons. It contains two important functions.

- **comboOf-”Weapon”:**

  That returns a list with all the weapon combos. They can be called in the comboSystem.

```
public List<comboAttack> combosOfPunch()
public List<comboAttack> combosOfSword()
public List<comboAttack> combosOfAxe()
public List<comboAttack> combosOfSpear()
createCombosOf-”Weapon”:
```

In these functions are programmed the lines with the combo steps creation, called as comboAttack, and their relations with other comboAttacks to create the tree possibilities in the combos.

```
private List<comboAttack>createCombosOfPunch()
private List<comboAttack>createCombosOfSword()
private List<comboAttack>createCombosOfAxe()
private List<comboAttack> createCombosOfSpear()
```

ComboAttack

Every comboAttack has some properties that define it. First of all the main button to activate it, the A or B button. Second the direction, UP, DOWN or NULL. After that comes two boolean, one for know if is an initial comboAttack (that kind of comboAttacks can be used to begin a combo) and other to know if it is a final comboAttack. Both booleans are initialized as false. The animation that goes with the combo step is defined with two integers, the combo Id, and its position in the combo. After that, comes the numeric characteristics: the damage of the combo step and the reaction time to continue the combo after the previous step end. Also there are some additional variables to define extra behaviours, like a jump or the special actions. In the next code it can be seen how is a combo defined. In this case is the combo 2 of the axe:

```
//Creación de los combos
comboAttack c2_1 = new comboAttack (“B”, DIRECTION.
NULL , time);
c2_1.setInital (); //es inicial
c2_1.setAnimation (2, 0); //animacion
comboAttack c2_2 = new comboAttack (“B”, DIRECTION.
NULL, time);
c2_2.setAnimation (2, 1);
comboAttack c2_3 = new comboAttack (“B”, DIRECTION.UP,
true, time);
c2_3.setAnimation (2, 2);

//Ponemos los hijos
c2_1.setChilds (new List<comboAttack> (){ c2_2 });
c2_2.setChilds (new List<comboAttack> (){ c2_3,c3_2});

//Los añadimos a la lista
combosList.Add (c2_1);
combosList.Add (c2_2);
combosList.Add (c2_3);
```

The comboAttack class has only one relevant function (the rest are constructors, getters and setters). That is the check function

- **Check:**

 It verify if the combination of the key, the direction and the timing, sended by the ComboSystem, is suitable to activate that combo step.

ComboSystem

The ComboSystem is an script of the Player. In this is defined the fight system main block. Here is interpreted the information sent by the CharacterController (the script that controls the actions of the player) to know if it has to activate a combo.

In this there are some important attributes. First of all some informations of other scripts, like the ComboManager, or the other scripts of the player (BasicCharacter script, CharacterController script, the current weapon of the player and CharacterAnimation script). To the right system work are needed two comboAttack Lists. The first with all the weapon combos (it is thanks to the comboOf-Weapon functions of the ComboManager) and the second list with the actual possibilities of the player to continue or begin combos. To control the player timing there is a float that controls the time between comboAttacks with a minimum and a maximum value to reset the combos if it is necessary. There is an Integer that controls the sum of comboAttacks done correctly. Finally there are three booleans to control better the combo: comboBool to check if the player is in the middle of a combo, canContinue to know if can continue a combo or is in the middle of the animation and isFinal to know if the current comboAttack is the end of the combo. The combo System has some important functions.

- **Update:**

 It controls the time in the script. With the time, the function controls the booleans state, depending on the time and themselves. If the time arrives to the maximum time, the active combos list is restarted with the comboAttacks in the list with the total combos that have the boolean IsInitial in true. This is done by beginCombos function.

- **BeginCombos:**

Search in totalCombos list the comboAttacks with the boolean isInitial in true and save it in activeCombos.

- **SetWeapon (weapon):**

Update the totalCombos list with all the combos of the new weapon received.

- **CheckCombos**:

 The main function of the script is CheckCombos. This function receives the key pressed by the player. It interprets the current player state and combo state to know what comboAttack has to do and if the player could do it. First of all it checks if the pressed key is the parry. If it is that, the function does an action and finish. Then it checks if the pressed key is the dash. If it is that, the function cut the combo totally, the weapon lost a life and the active combos are rebooted with begin-

Combos. Then it checks if the weapon is ready to do the next combo testing the current canContinue state. If it is false the combos are reseted as before. Then, finally arrives checking the current comboAttacks actives. In a for bucle, the function goes over the active comboAttacks using its own check function. If the check is correct the code interprets the comboAttack information(if it is final, if it has to do the player jumps or others). Furthermore update their own attributes (the Combo steps number completed in a row, change the booleans) and active the correct animation. If any comboAttack from the active combos are checked correctly it means that the player fails with the buttons combination and then it calls to the function beginCombos.

```
public void CheckCombo(string key){
     [...]
     if (!canContinue) {
        comboBool = false;
        controller.getWS ().canContinue (combo-
Bool);
        controller.getWS   ().activate   (combo-
Bool);
       numCombo = 0;
       controller.lostWeaponLife ();
       beginCombos ();
       return;
        }
       DIRECTION direction = character.getDirec-
tion ();
       foreach (comboAttack cA in activeCombos)
{
       if (cA.check (tempo, key, direction)) {
            tempo = 0;
            comboBool = true;
            canContinue = false;
       controller.getWS ().activate (comboBool);
        controller.getWS ().canContinue (canCon-
tinue);
       if (cA.onAir) {
           character.Jump (cA.jumpStrong);
          }
          cAnimation.combo (true);
          controller.getWS ().damage = cA.damage;
         Animation.changeCombo(cA.getAnimation());
           numCombo++;
           activeCombos = cA.getChilds ();
           isFinal = cA.finalCombo;
           return;
        }
     }
     //Si no es ninguno de los cA correctos
     comboBool = false;
     controller.getWS ().activate (comboBool);

     numCombo = 0;
     controller.lostWeaponLife ();
     beginCombos ();
 }
```

AnimationCombo

The animationCombo script is a little code applied to every animation (in every comboAttack). It changes the canContinue value of the ComboSystem to true at the end of the animation.

WeaponSystem

Finally there is the weaponSystem script. It is a code attached to the weapons. It controls the weapon colliders. If the player is in a combo the colliders stay activate. If not the colliders are inactive. In addition, it controls their particles, as a trail to draw the weapon trace or a particle system for the weapon hits. It controls to, obviously, its life and damage.

CHARACTERS

In this project the characters are the type of objects who receive damage, can move and jump. There are two kind of characters able to do this: the player, and the robots. The character's base is in the script BasicCharacters. The characters have always some objects and components:

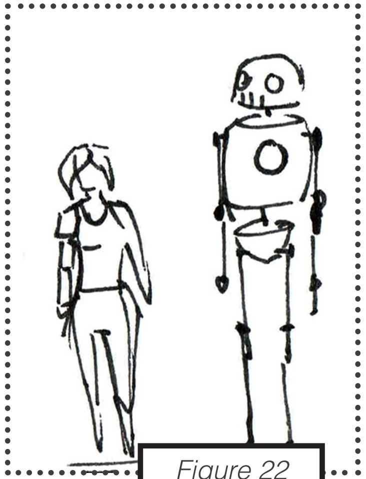


Figure 22

Figure 22: Concepts of characters.

Render

This object contains the character's meshes. All those meshes are Skinned Mesh Renderer type. That permit to Unity3D animate their vertex with the other Render's part, the skeleton.

The skeleton is in charge to do the animations done in 3d Max. It is composed by all the human body bones (simplified).

Furthermore, the render includes on the right hand bone three invisible objects: the weapons. They are attached to that bone to move with it.

CheckFloor

This object has a simple collider under the render to check if the characters is on the floor or not. It is calculated in the BasicCharacter script.

BasicCharacter

This script is the base of the characters. It controls the jump, movement and damage of a character. The class has some main attributes. References to the character's rigidbody, the CheckFloor object and CharacterAnimatorController. Parameters floats or integers as the character's life, jumping force and speed movement. The character's type (player, NPC or enemy). And some booleans to know if the character is in the ground, moving, turning, dead or others. It has some important functions:

- **Move:**

This function receives a vector 3. This vector is applicated over the character to move it.

- **Jump:**

This function receives a float value and it applies a velocity to the rigidbody.

- **getDamage:**

This functions receives an integer and the function subtracts it from the character life. In addition, it puts the character invulnerable for some milliseconds, and the character can't receive damage during that time.

- **OnTriggerEnter:**

OnTriggerEnter is called when other Collider actives the character's trigger. If that Collider is tagged as "Weapon", it calls to getDamage function.

Animator

This component is an Interface to control the Mecanim animation system. Every character have their own Animator Controller [12].

CharacterAnimationController

This is a script to simplify the communication between the animator and the scripts. All the game animators Controller are similar, they have the same parameters. With that in mind, the script simplify the functions to edit that parameters and it does that more readable.

The main character

The player controls the main character. It is an object with a lot of components and objects [Fig.23].

Character Controller

The main script. It has references to all its objects, it controls their situation and it interprets this. With that, the player can control the main character with his/her inputs. It has references to all its weapons, its BasicCharacter, Render, ComboSystem, CharacterAnimationController and some effects. It has booleans and timers to controls action as the dash, the parry and the movement. It has some important functions:

- **Start:**

Initialize the biggest part of the components and booleans.

- **Update:**

Update the timers, sends the movement to the BasicCharacter, update the animation and recives the inputs. If it is an Xbox controller connected, it calls to UpdateController. Else it calls to UpdateKB.

- **interpreteMove:**

Interpret the input movement information to make the numbers friendlier for the game.

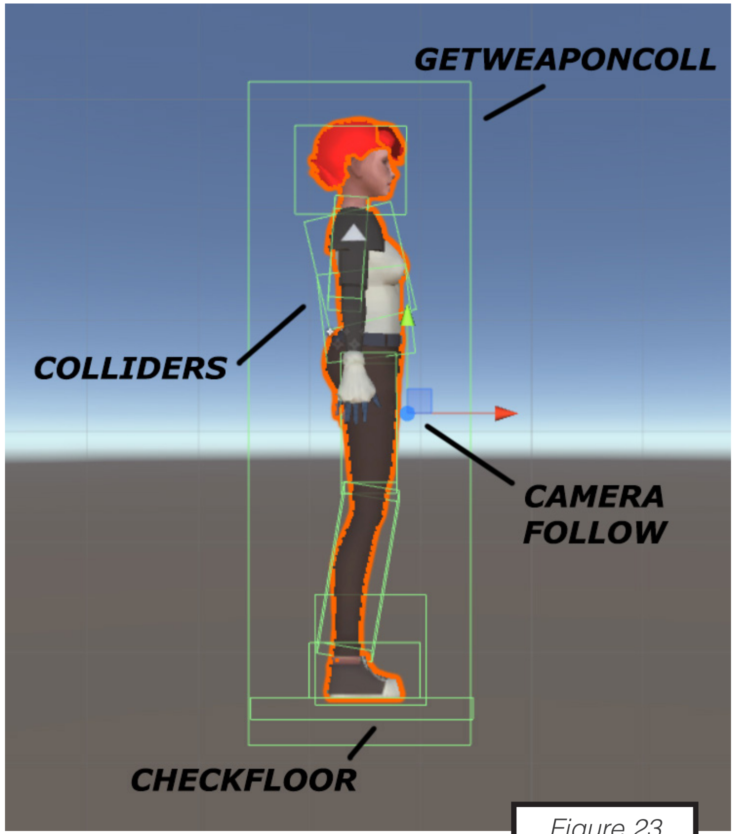


Figure 23

Figure 23: Colliders of the main character.

- **UpdateController:**

It receives the inputs of the Xbox Controller and response to that.

- **UpdateKB:**

It receives the inputs of the KeyBoard and response to that.

- **getFreeWeapon and changeWeapon:**

The first is used when the player uses the Take action. It search if there is a near free weapon. If it is true, it calls to changeWeapon and changes the player's current weapon to the nearest Free Weapon.

- **lostWeaponLife:**

This function changes the life of the current weapon and destroys it if the life arrives to 0.

- **Parry and parryAct:**

Both functions are the mechanism of the parry. It makes invulnerable the player during the parry duration

- **Dash and dashMove:**

Both functions are the mechanism of the dash. It moves fast the character on the initial direction of the player for few seconds.

ComboSystem

It is explained in previous pages [page 35].

BasicCharacter

It is explained in previous pages [page 37].

CharacterAnimationController

It is explained in previous pages [page 38].

DashTrail

A visual effect object to make more visible the dash movement. It is explained in next pages [page 47].

ParryEffect

A visual effect object to make more visible the parry action. It is explicated in next pages [page 47].

CameraFollow

An object towards which the camera looks.

GetWeaponColl

A collider to make easier to the freeWeapons render a line between the player and themselves.

Render

The main character's render is special. It is because it have specific colliders to do more realistic it.

Animator Controller

It is one of the most game complex systems. It controls the animations and transitions of the player. It's divided in some sub-state machines. In the base layer there are four sub-state machines and one state.These sub-states divide the possibles animations and simplify their communication. The following figure shows a part of the animator controller [Fig.24]. The parameters are self explanatory.

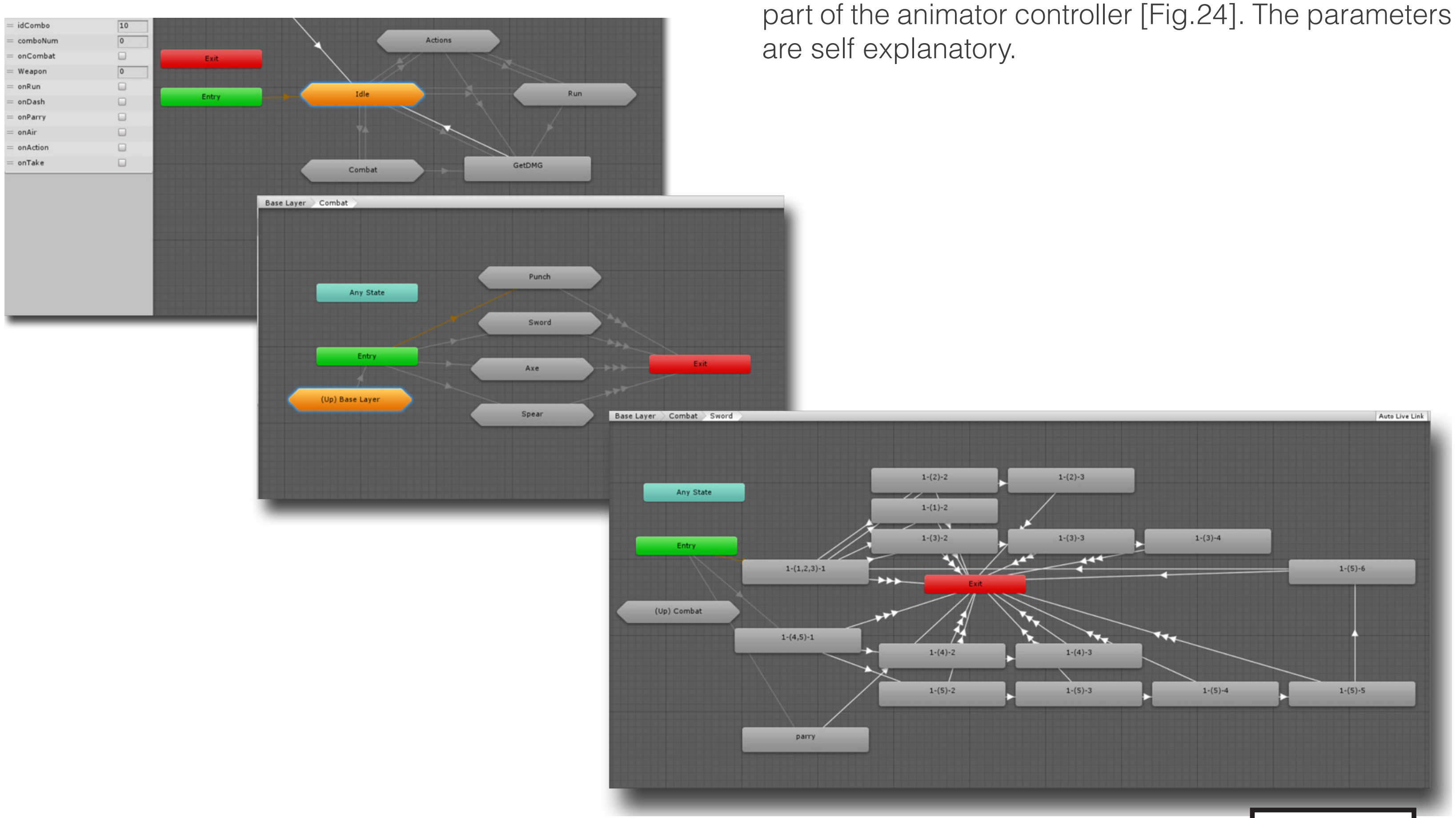

idCombo	10
comboNum	0
onCombat	☐
Weapon	0
onRun	☐
onDash	☐
onParry	☐
onAir	☐
onAction	☐
onTake	☐

Figure 24

Figure 24: Differents states and substates on the Animator of the main character. It can be seen better in the section "Big Figures".

The Enemies' Artificial Intelligence

The Artificial Intelligence (AI) represents all the robots of the game. All of them are under the same workflow, but changing usually the IABasic variable's weapon. It has a lot of components too [Fig.25].

BasicCharacter

It is explained in previous pages [page 38].

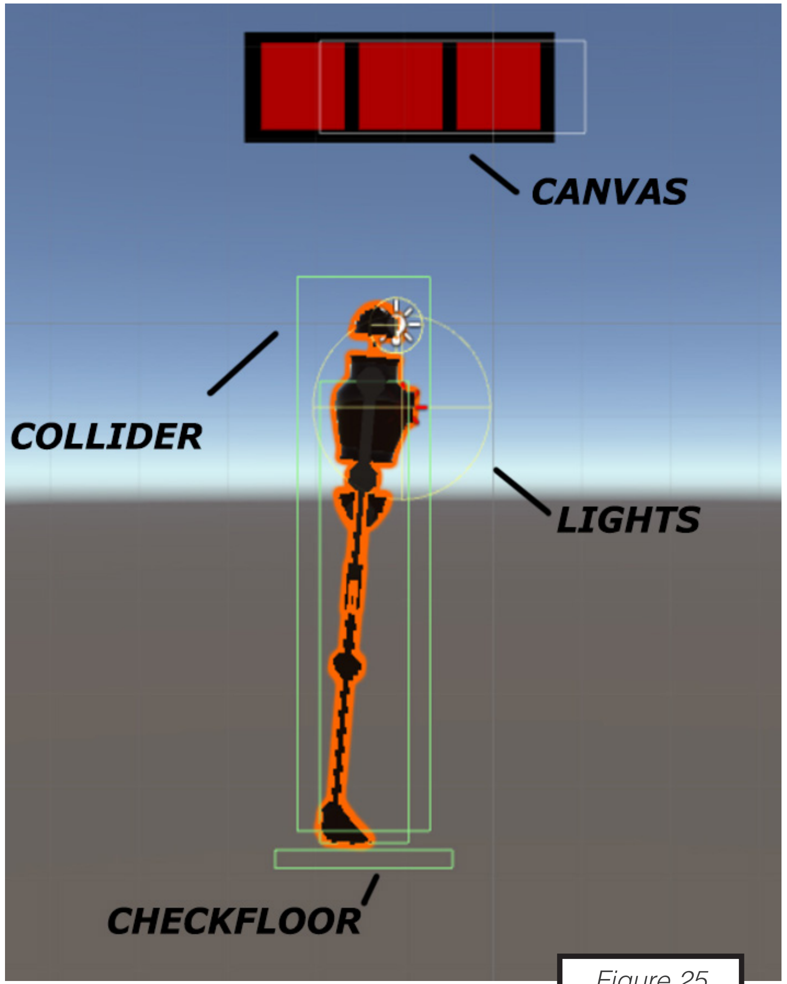


Figure 25

Figure 25: Colliders of a robot.

CharacterAnimationController

It is explained in previous pages [page 38].

Render

It is explained in previous pages [page 37].

Canvas

The robots have their own canvas above them. In it is showed their current life.

IA Basic

The main script of this kind of characters. It controls the mindset of the robot. The script has references to a lot of robot's components. Also it has a lot of control attributes and important functions:

- **Start:**

This function is used to initialize its resources.

- **setWeapon:**

This function changes the character's current weapon, doing visible the selected weapon and changing the distance to set in front of the player (more distance if the weapon has more range).

- **Update:**

This function calls to the principal action functions of the script periodically.

- **updateAttack**

updateAttack looks the robot's current state and if it can attack it does that. To attack, it generates a random number to activate the animation of one of the combos prepared for him.

- **updateAnimation:**

This update, with their attributes and the help of its CharacterAnimatorController, the animator controller.

- **Death:**

This function is called when the lives of the robot arrives to 0. It applies some visual effects, creates a freeWeapon with its own weapon and, finally, destroys himself.

- **checkMove:**

checkMove is the main function to move the object. It checks the distance to the player. If the player is far from the character doesn't detect the player and it stays still. If the player is very close, the character enters in fight mode, ready to the battle. If none of the above happens the character moves into the player's direction.

- **checkJump:**

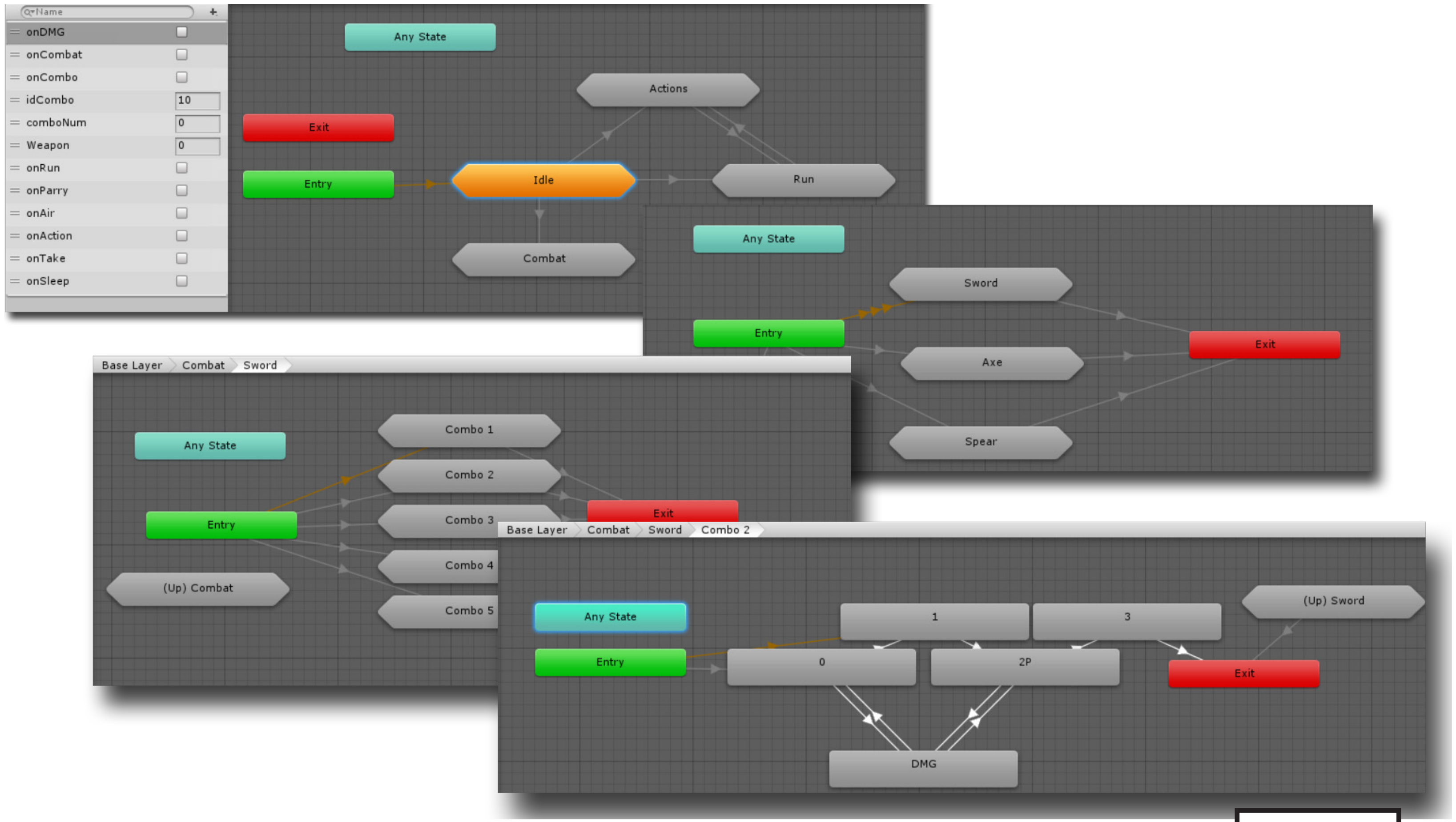
This function makes a raycast forward to know if there is a thing to be jumped. If the raycast collides with a floor the script calls to the BasicCharacter's jump function.

```
void checkJump (){
        RaycastHit hitInfo;
        Vector3 mA = new Vector3 (0, m_Render.lossyS-
cale.y/2, 0) ;
        Vector3 dir = Vector3.zero;
        if (m_Right)
            dir = Vector3.right;
        else
            dir = Vector3.left;
              if (Physics.Raycast (transform.position
- mA, dir, out hitInfo, m_JumpCheckDistance)) {
          if (hitInfo.transform.tag.Equals (“Floor”))
{
                m_Character.Jump (1);
            }
        }
    }
```

Animator Controller

It controls the animations and transitions of the robot. Is divided in some sub-state machines. It is simpler than the player's animator controller. In the base layer there are four sub-state machines and one state. These states divide the possibles animations and simplify their communication. The following figure shows a part of the animator controller [Fig.26].

Figure 26: Differents states and substates on the Animator of the enemies. It can be seen better in the section "Big Figures".



onDMG	☐
onCombat	☐
onCombo	☐
idCombo	10
comboNum	0
Weapon	0
onRun	☐
onParry	☐
onAir	☐
onAction	☐
onTake	☐
onSleep	☐

Figure 26

CHARACTERS

WEAPONS

The weapons are other of the game pillars [Fig.27]. The weapons are objects for the characters. They can be in two states: Attached to a character or floating on stage, prepared to be taken by the player. Depending of the states, the weapons are really differents.

Attached weapons

This kind of weapons are always with a character, usually in their right hand position. These weapons have their own colliders and their WeaponSystem. The weapons have a type to detect as damage the collision of an object if the type is different to who recibe the collision. The type is assigned depending of their parent: if it is the player, the type is player, else it is enemy.

Furthermore, the weapons have some visual and sounds effects. A trial to draw the weapon trace. A particle system for the weapon hits. This last is rendered when the weapon collides with the player or with a robot.

Free Weapons

The free weapons are objects thoughts to be caught by the player. After ten seconds the objects self-destruct. They have some important components. First of all, three transforms that have the meshes of the weapons. A sphere collider to detect if the player is near them. If the player collides with the sphere, a line is rendered between the freeWeapon and the playe, with a Line Renderer. Also, it has a plane with an effect to see better the weapon. This objects have two scripts.

FreeWeapon:

This script manages the object. First of all controls the time to be prepared to the self-destruction after 10 seconds. And it prepares the line render when the player collides with the freeWeapon.

FloatingEffect:

This is an effect applicated to the object. With it, the transform turns on itself and bounce up and down.

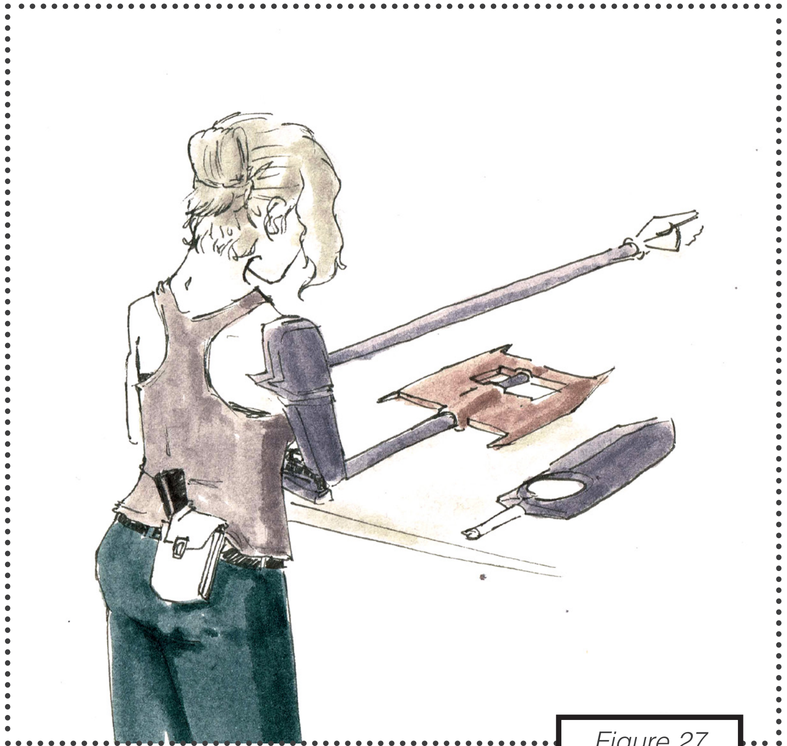


Figure 27

Figure 27: Concept of Erin picking out a weapon.

SCENES

In Unity3D, scenes contain the game's objects and work as levels for a game. The project has 2 scenes. The Menu and the Arena.

Main Menu

This scene is the starting point of the game. In it you can enter in the arena mode, see the control and leave the game. It works with the Unity UI objects. The scene has four buttons.

Story button

This is only representative, doesn't work

Arena button

This button leads to the scene Arena, to the Arena Mode.

Controls button

This activates and deactivates a sprite with the controls of the game.

Exit button

This button close the game.

The scene has one script manager, the MenuScript script. This permits the management of the menu with the mouse and with a Xbox Controller.

Arena

The Arena scene is the scene where the arena mode happens. In it is programmed the wave system, and the player can play it. The scene have some objects for its operation:

Setting

On it are all the objects that define the environment of the game: the floor, platforms, walls and lights.

Main Camera

This object is the scene's camera. It has a script, CameraController. This script allows to follow the player with fluidity. Furthermore the camera shakes when the player get damage.

Game Manager

This contains the scripts of the game. It has the ComboManager, the VariableManager (a script with some common values, accessible for all the objects), the pauseScript and the WaveController.

pauseScript

This script manages the game pause. In the pause the time is stopped and the script shows the combos of the current weapons. It helps the player to know what kind of things can be done.

SCENES

WaveController

It is explained in next pages [page 46].

HUD

The HUD is the interface of the game. It is explained in next pages [page 48].

An initial Free weapon

It is explained in previous pages [page 44].

The player

It is explained in previous pages [page 38].

WAVE SYSTEM

[Brackeys: How to make a Wave Spawner in Unity 5 - Part 1/2]

The wave system manages the waves and all their relations. It applies the rules established in the chapter 3 [page 23]. To do it, it has 4 empty objects, which work as spawn points, and the WaveController script. The wave controller is based in a script of the youtube channel *Brackeys* [13].

It has some important attributes as a list with the waves, its state and some timers. It controls the arena's HUD too. The important functions are the next:

- **Start:**

Initialize some values and start the coroutine initArena.

- **initArena:**

It shows the arena's HUD with "WELCOME TO THE ARENA MODE " written in the text attribute. After 2 seconds it fade out the HUD.

- **updateHUD:**

It shows and fade out the HUD with "WAVE" + the number of waves written in the text attribute.

- **Update:**

It updates the timers and the conditions necessaries to begins a new wave.

- **SpawnWave and SpawnEnemy:**

Spawns the next wave. Spawns the next enemey in a random spawn point.

- **SpecialSpawnWave and SpecialSpawnEnemy:**

They are used in the first wave. The first needs to instantiate a specific enemy in a specific spawn point and this function manages it.

Effects

[Sirhaian'Arts: Unity VFX Tutorials - 04 - Basics (Hit)]

[imn nam: Game effect tutorial - Sword Attack]

There are some visual effects in the game to do easier understand what is happening in the screen [Fig.28]. All of them have been made to follow the tutorial of the youtuber *Sirhaian'Arts* [14].

Hit

It is used when a weapon hits a character. It is composed by 3 particles system: a glow, sparks and a shockwave.

GetDMGRobot

It is used when a robot gets damage. It is composed by 4 particles system: a glow, a shockwave and 2 rays.

Parry effect

It is used when the player uses the parry. It is composed by a shockwave.

Dash trail and weapons trail

They are used when the weapons are activated. They are taken from a video of *imn nam* [15].They are composed by A MeleeWeaponTrail script and two objects to limit the begining and the end of the trial.

DeadWeapon

It appears when a weapon lives is equal to 0. It is composed by 3 particles system: a glow, sparks and a shockwave.

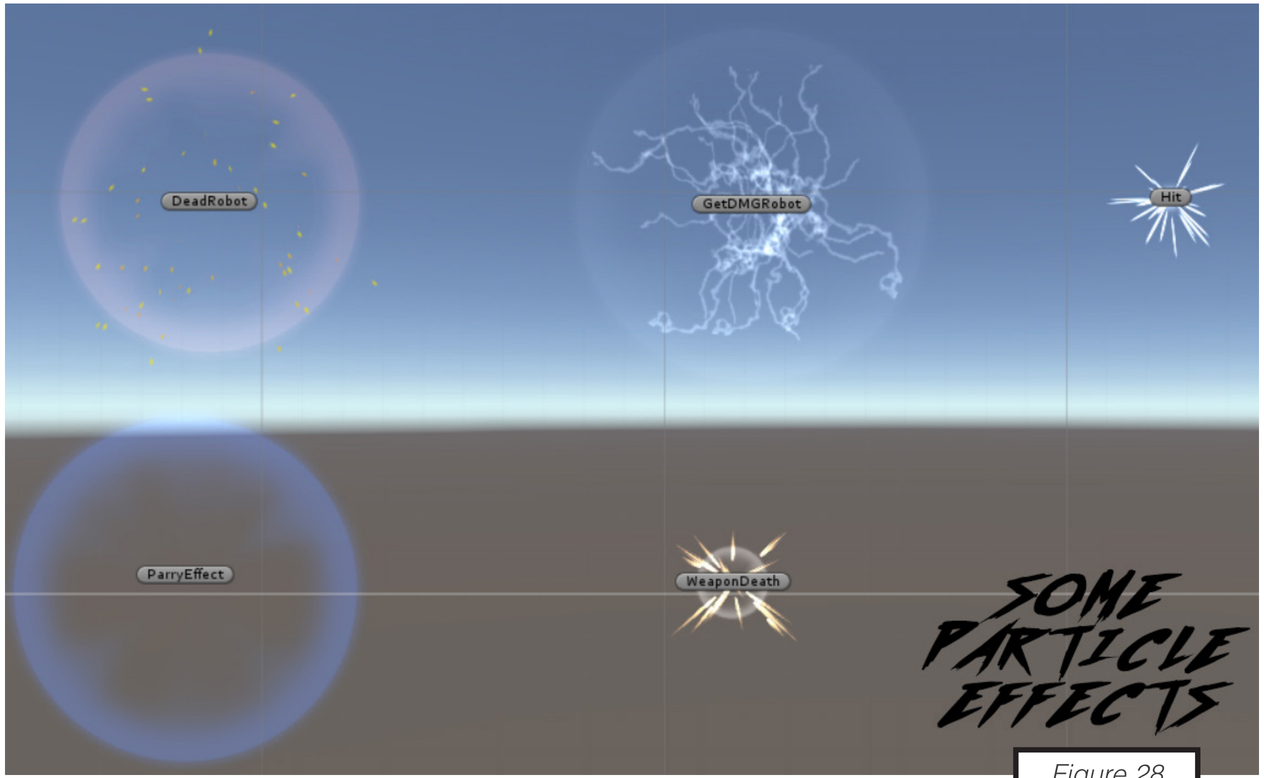


Figure 28

Figure 28: Examples of particles inside the game.

Interface

The function of the interface is to show important information to the player without being annoying for the player. The game interface can be divided in some parts:

Player information

It shows the player's lives, the current weapon and the current weapon lives [Fig.29].

Robot information

It shows the robot's lives. It is always over the character.

PauseHUD and deadHUD

This is showed when the player pause the game or he/she die, respectively [Fig.30].

Figure 29: Description of the user interface.

Figure 30: Example of the dead screen.

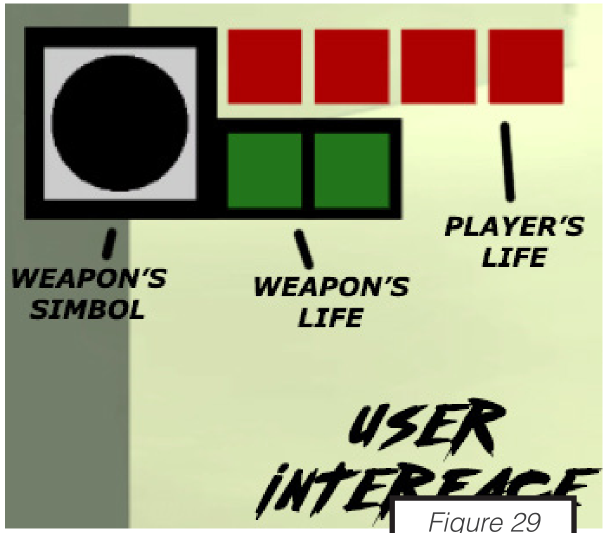


Figure 29

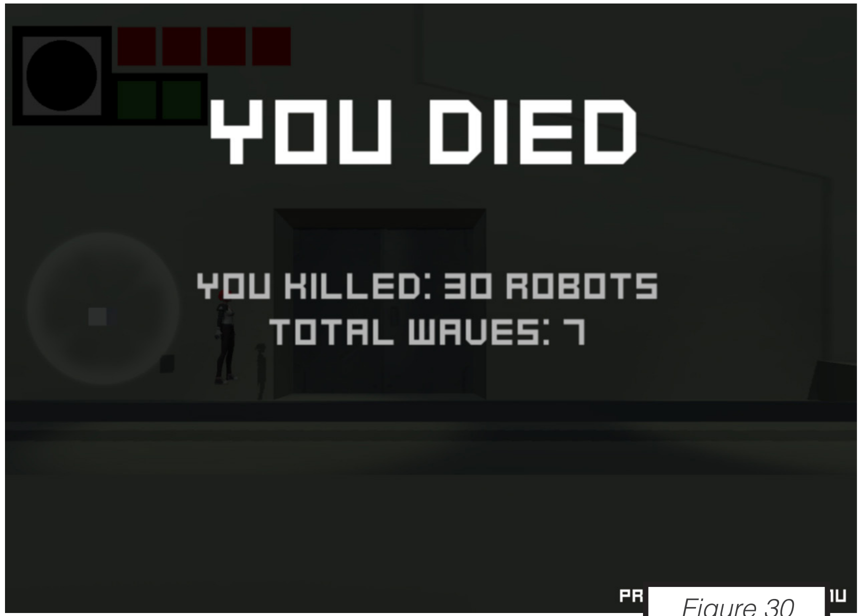


Figure 30

4

ART

ABSTRACT

This chapter includes all the art developed for the project. The art has some conceptual designs of the old narrative and others for the current styles, Erin's personality and differents designs of the robots.
It also includes the models and references to modeling the characters and their textures unwrapped and old texture designs.
Finally it includes an animation section. In it is commented the rigging and animation of the main character and robots.

ART STYLE

The game's art highlights in its quality. The game art hasn't got a personal art style (like *The Legend Of Zelda: The Wind Waker* [16] by Nintendo, or *Limbo* [17] by PlayDead) but it has got a solid art style and hasn't got big failures. The characters are realistic. The models have few polygons without being a low poly style. The animations are realistics and naturals, marking the postures to make them more readables.

CONCEPT ART

Erin

The first concepts are from Erin, the main character. Since the beginning, Erin was thought as a woman and not a girl [Fig.31]. She was really defined since the first concept. She is sure of herself and self-sufficient.

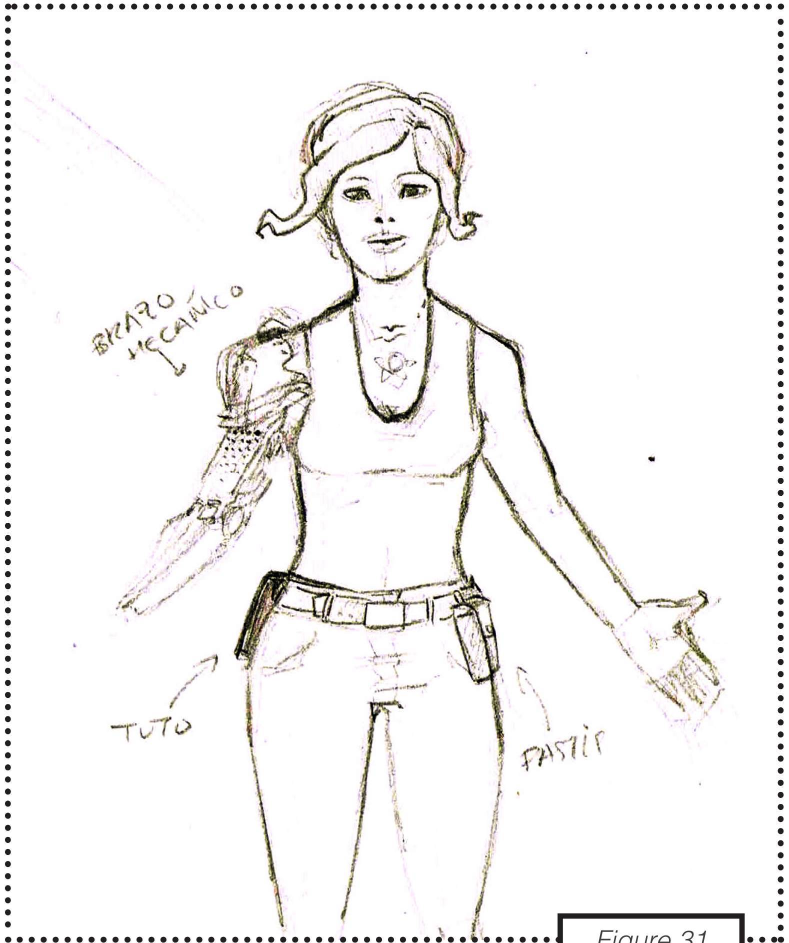


Figure 31

Figure 31: Concept of Erin. FIrst drawing.

CONCEPT ART

Erin has a mechanic arm and it was designed too. The final model has a simplified version of it [Fig.32-33].

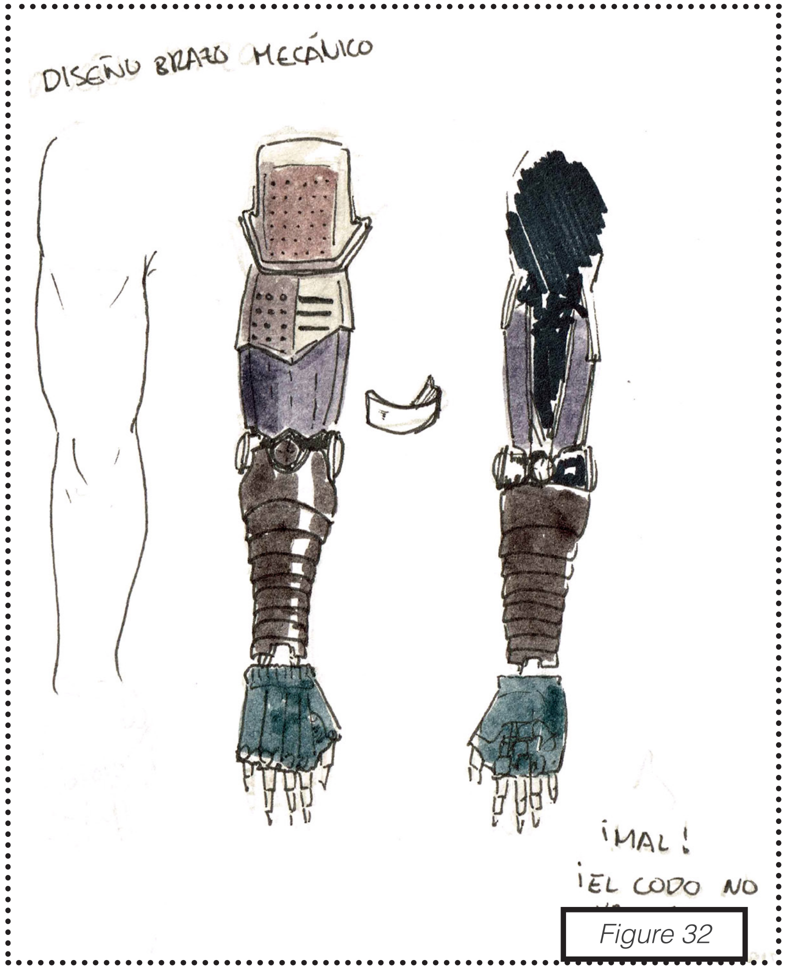


Figure 32

Figure 32: Concept of Erin. Arm design.

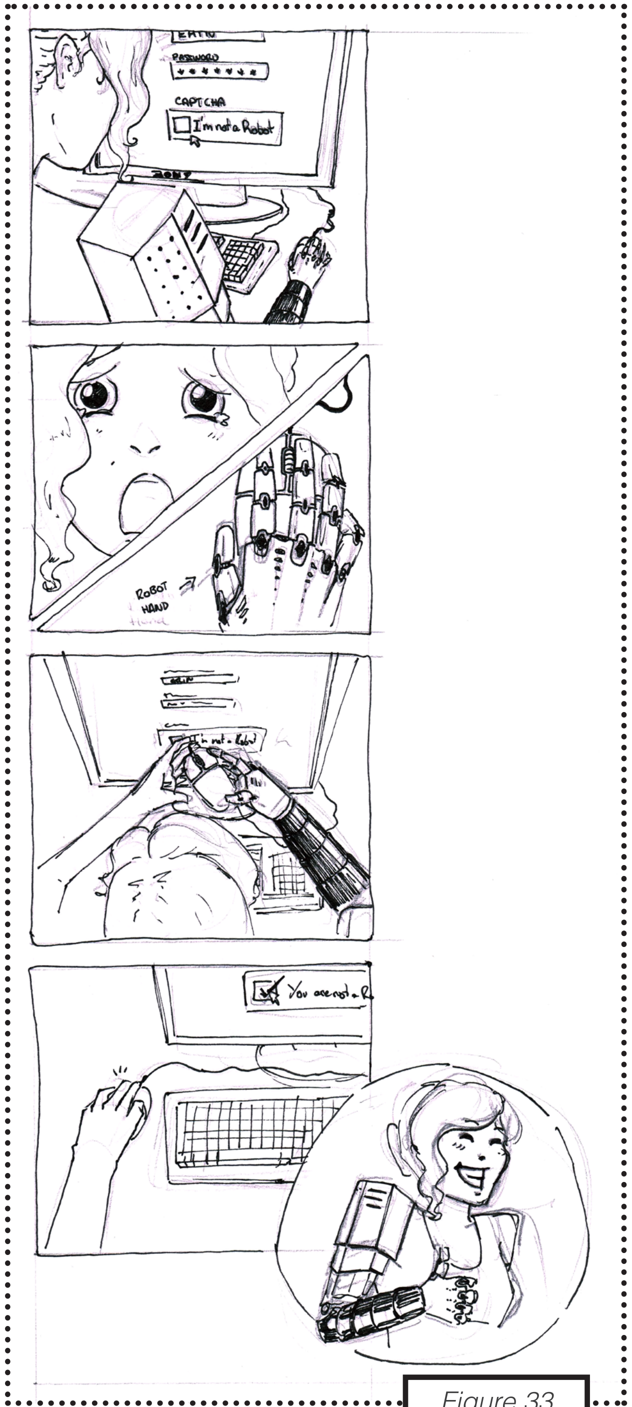


Figure 33

Figure 33: Concept of Erin. Mini-comic exploring Erin's personality.

Figure 34: Concept of Erin. Differents situations of Erin.

Figure 35: Concept of Erin. Differents situations of Erin

Figure 36: Concept of Erin. Differents situations of Erin

Figure 37: Concept of Erin. Differents situations of Erin

Erin has some concepts in other situation too [Fig.34-38].

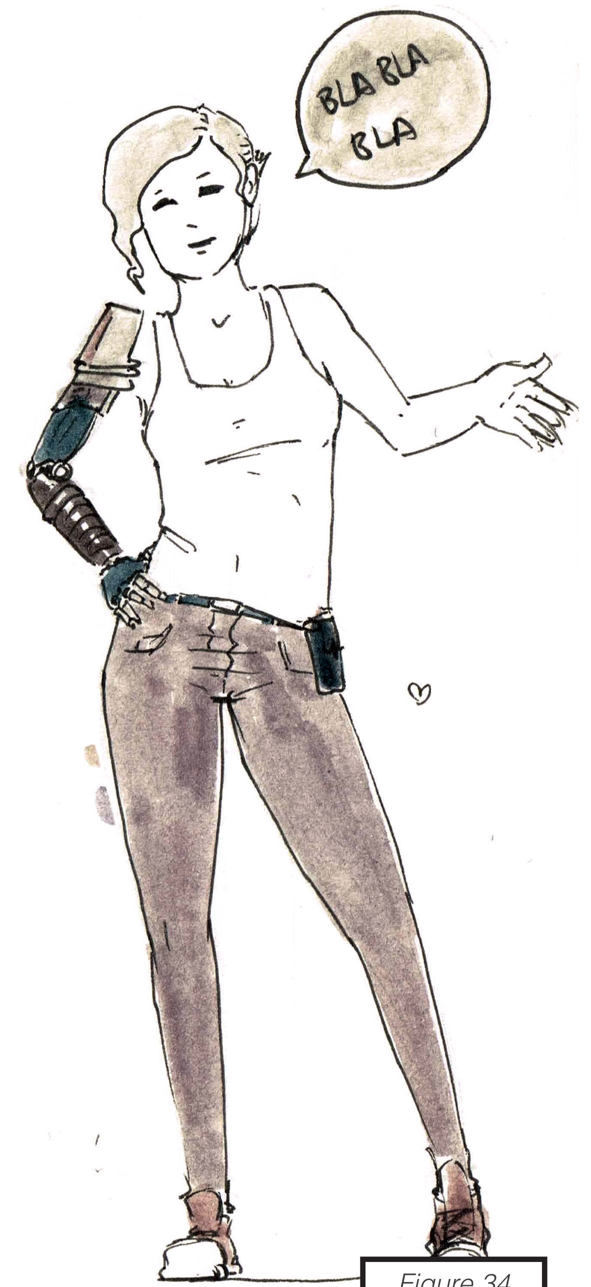


Figure 34

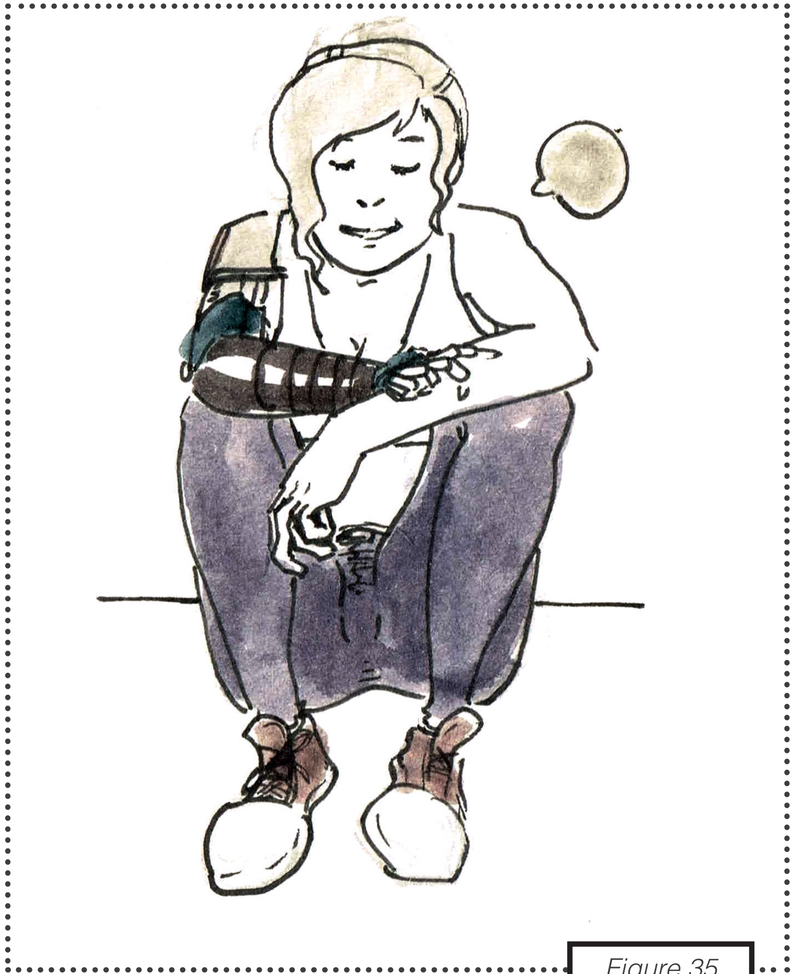


Figure 35

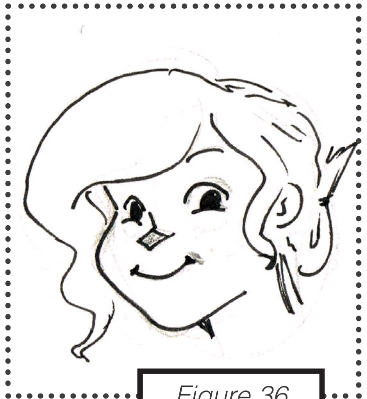


Figure 36

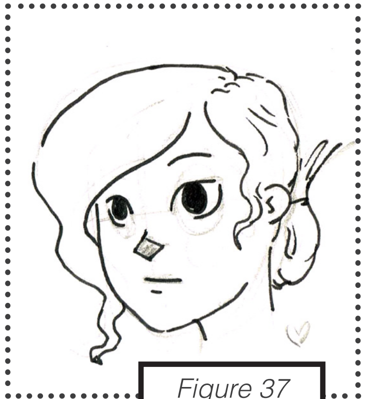


Figure 37

Figure 38: Concept of Erin. Reparing their own arm.

CONCEPT ART

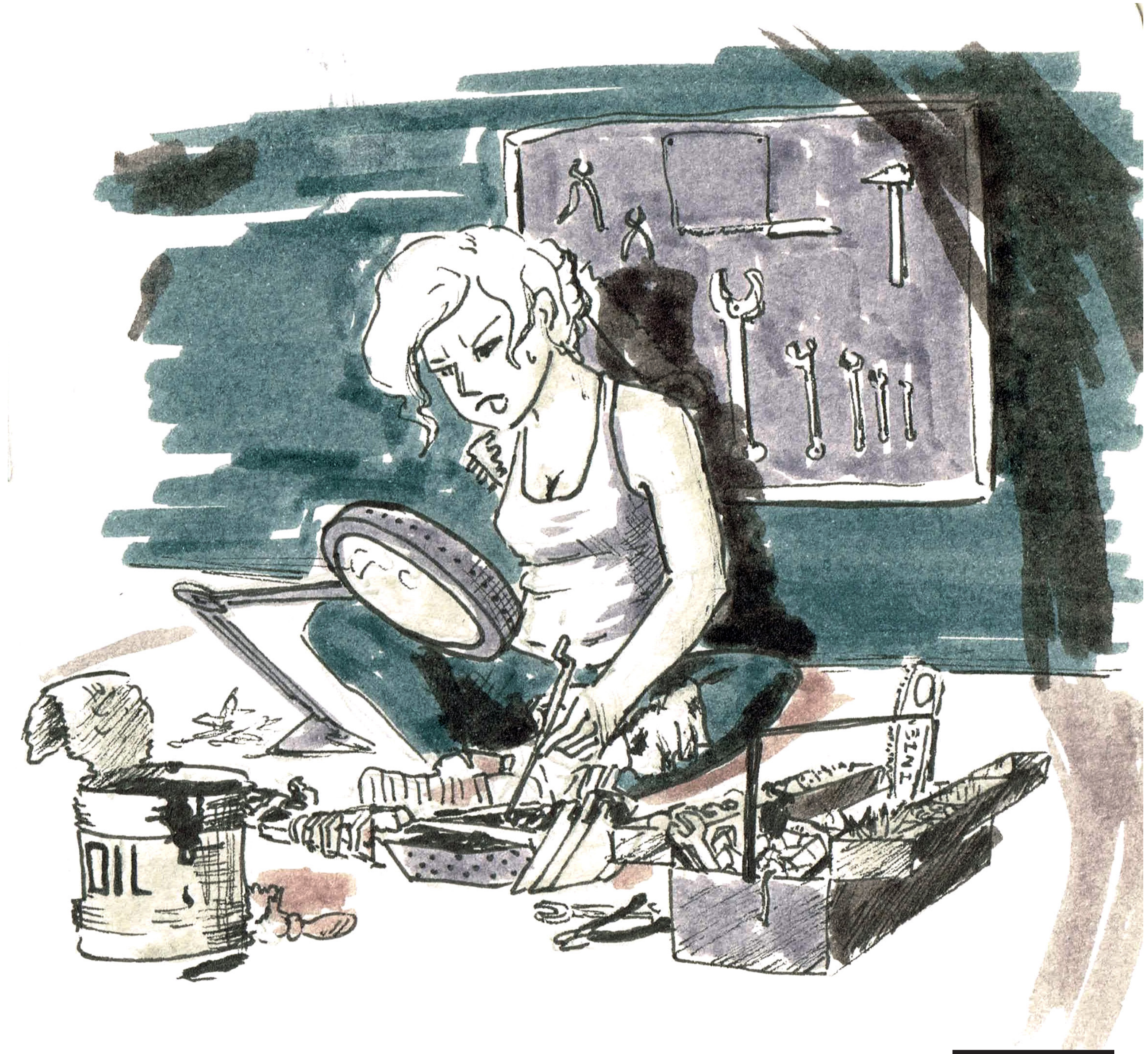


Figure 38

Finally, to know the Erin's colours, it was done a colour test in a digital version [Fig.39-40].

Figure 39: Colours tests of Erin.

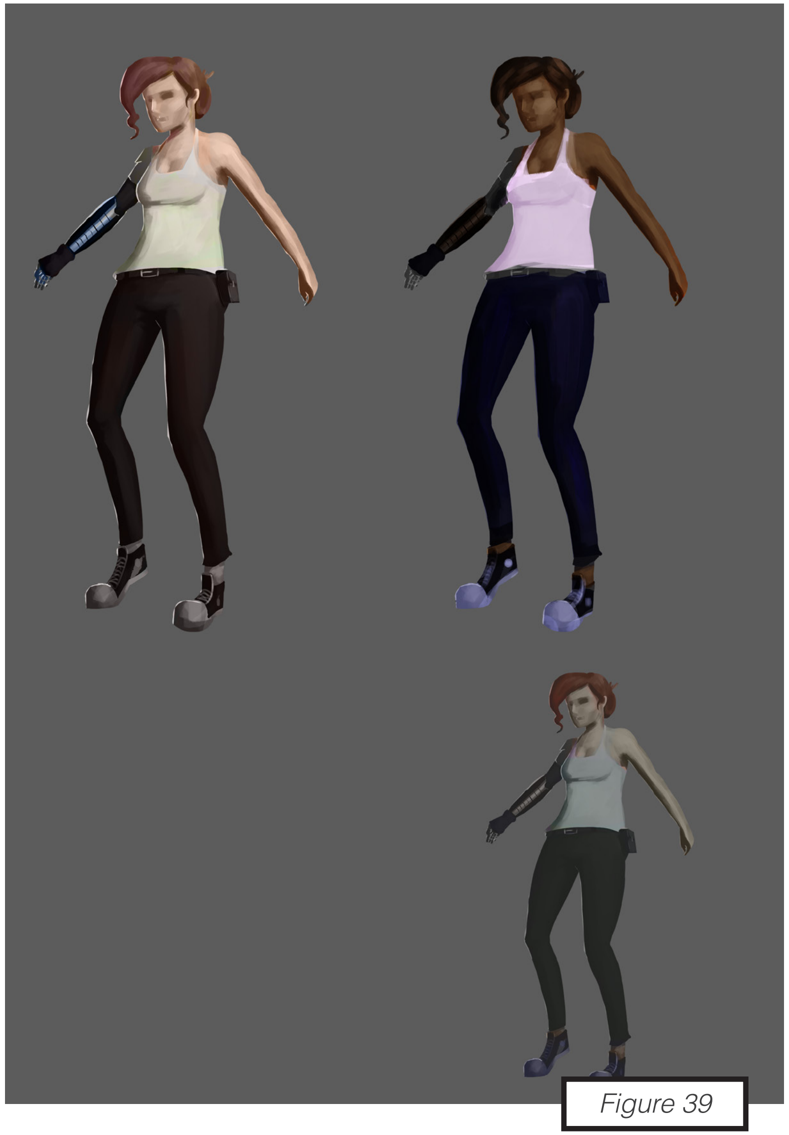
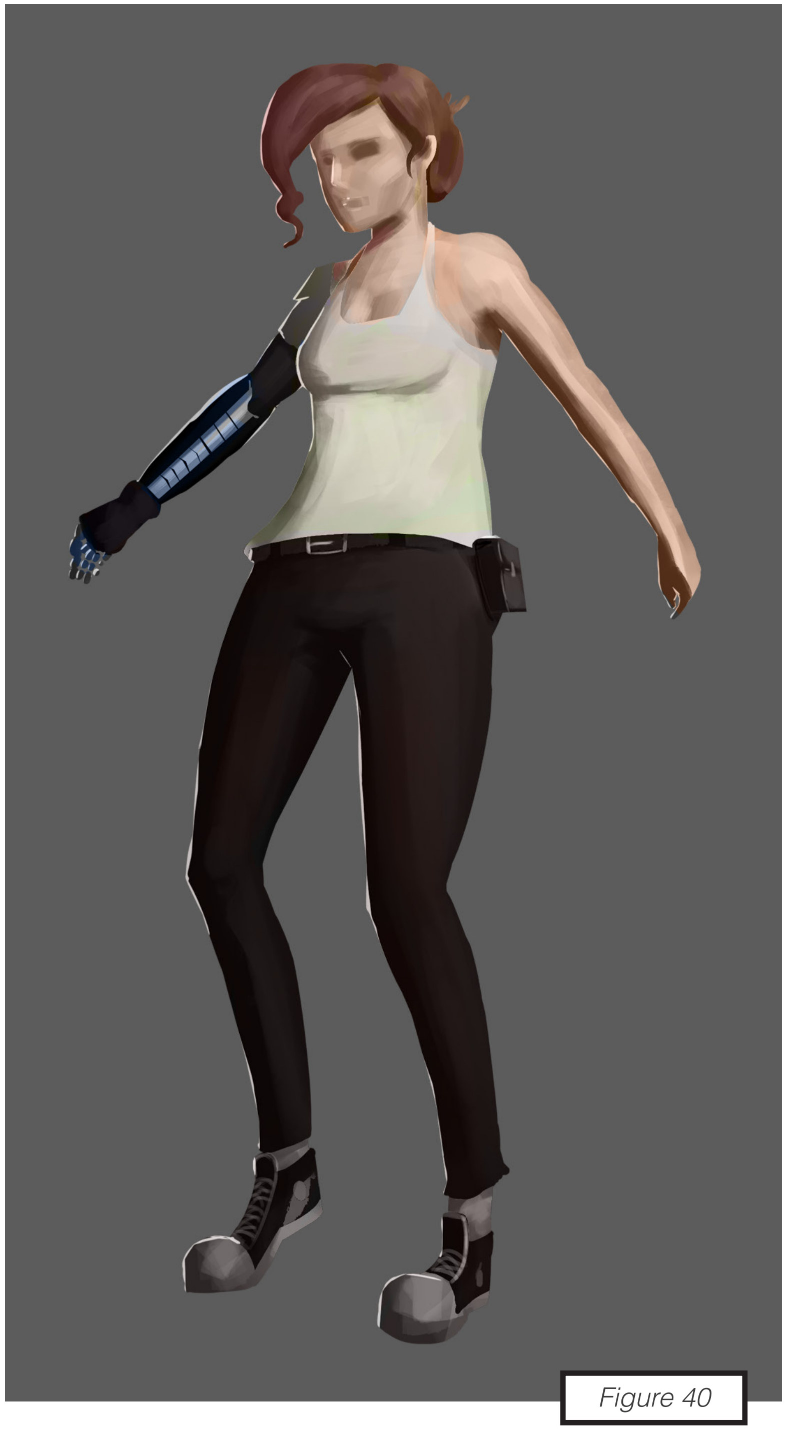


Figure 40: Definitive colours of Erin.



The selected was the next.

The influences of Erin were the next (Triss from *The Witcher 3: Wild Hunt* [18] by CD Projekt Red) [Fig.41].

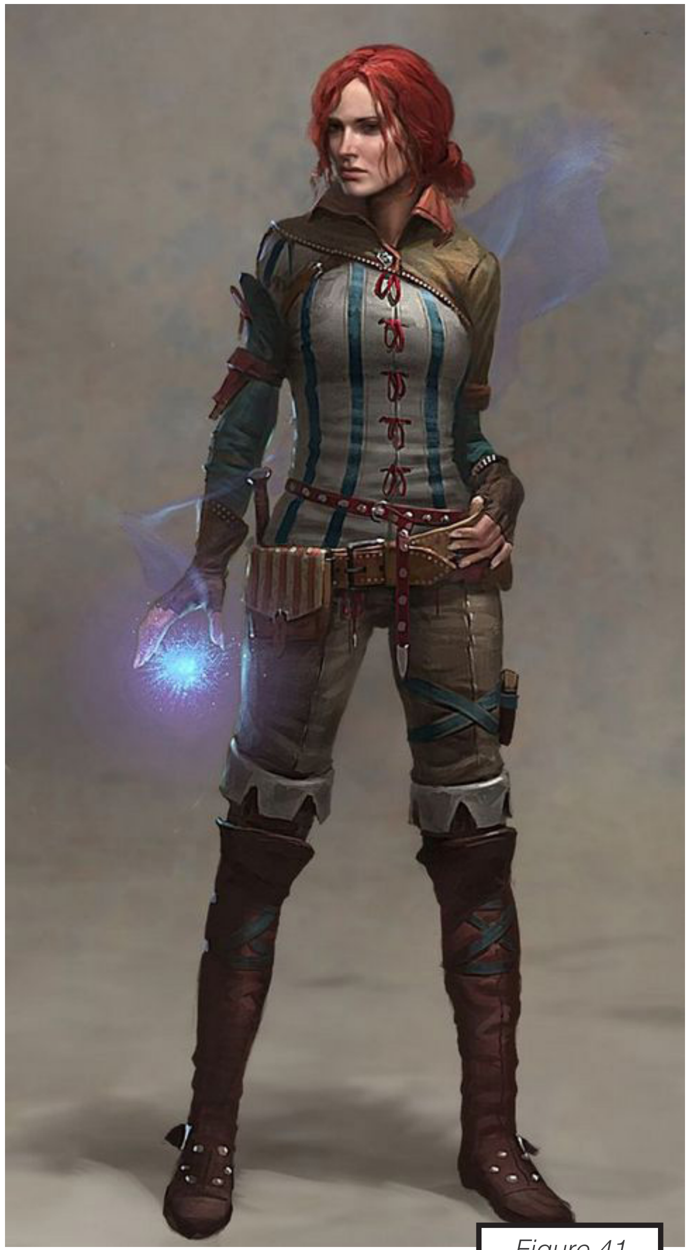


Figure 41

Figure 41: Triss from the Witcher 3: Wild Hunt by CD Projekt Red.

Robots

The robots were really difficult to design. They had many designs and influences, like *Star Wars* [19] (by George Lucas) or *ART 88/44* [20] (by Ulises Lafuente) [Fig.42-43].

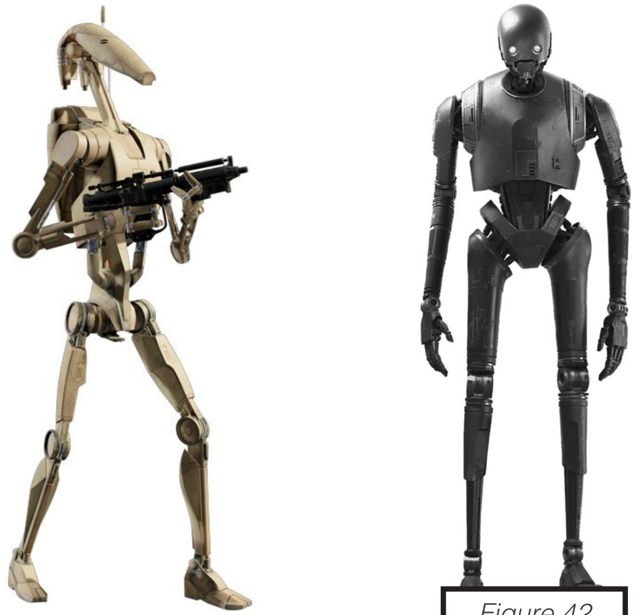


Figure 42

Figure 42: Some combat drones from Star Wars.

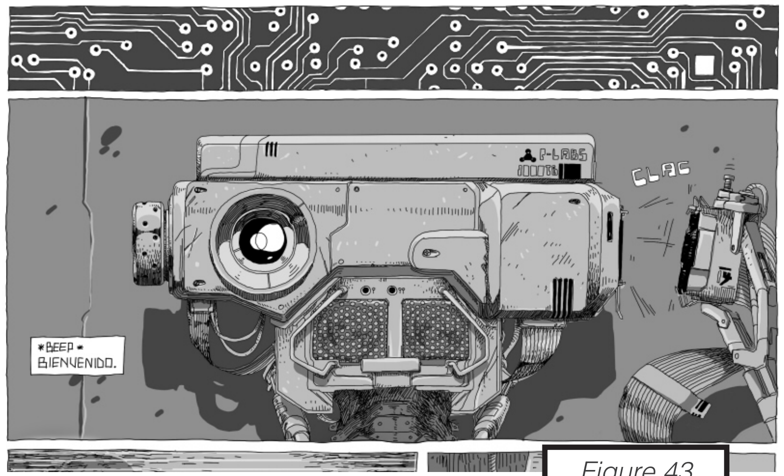


Figure 43

Figure 43: A robot of ART 88/44 by Ulises Lafuente.

The design changes a lot through time, been at the beginning more organic and complex [Fig.44].

Finally it was decided a more simple form. It was done to make the robot easier to animate and modeling [Fig.45].

Figure 44: Concepts of some robots. First desgins.

Figure 45: Concepts of some robots. Last designs.

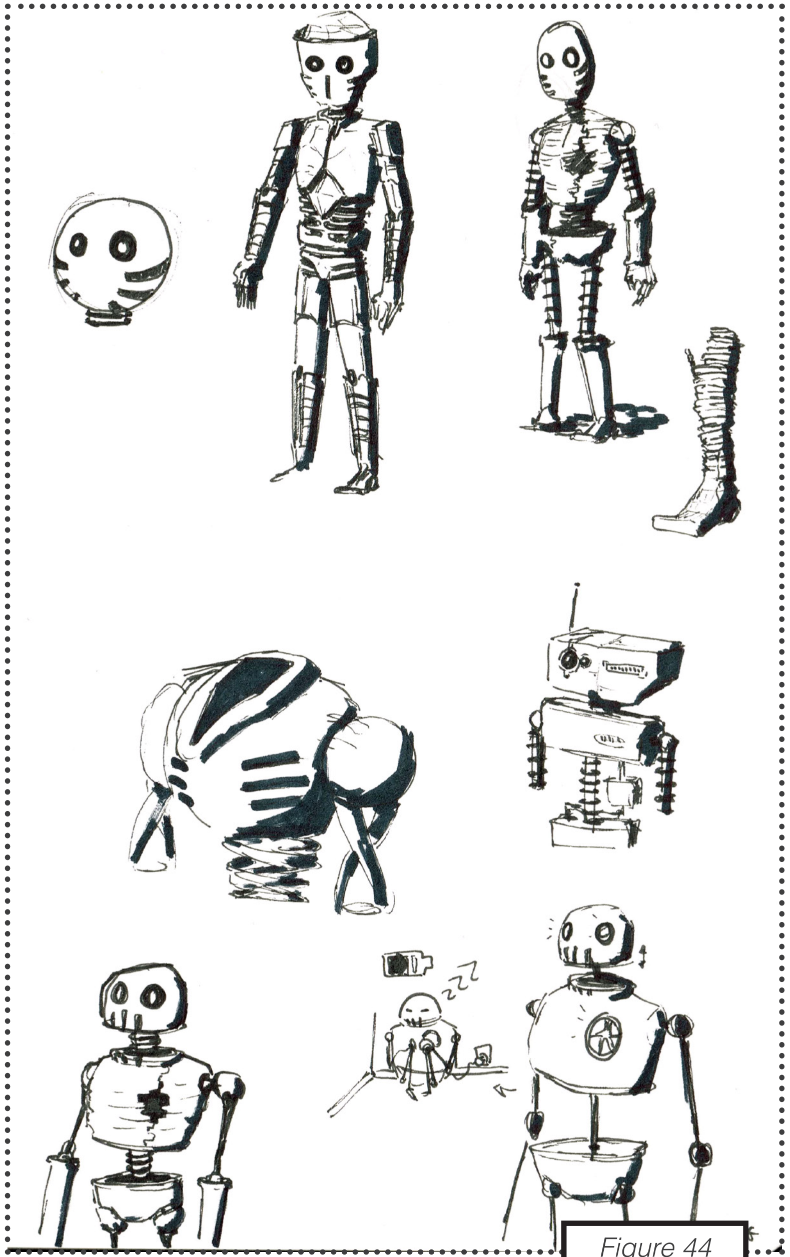


Figure 44

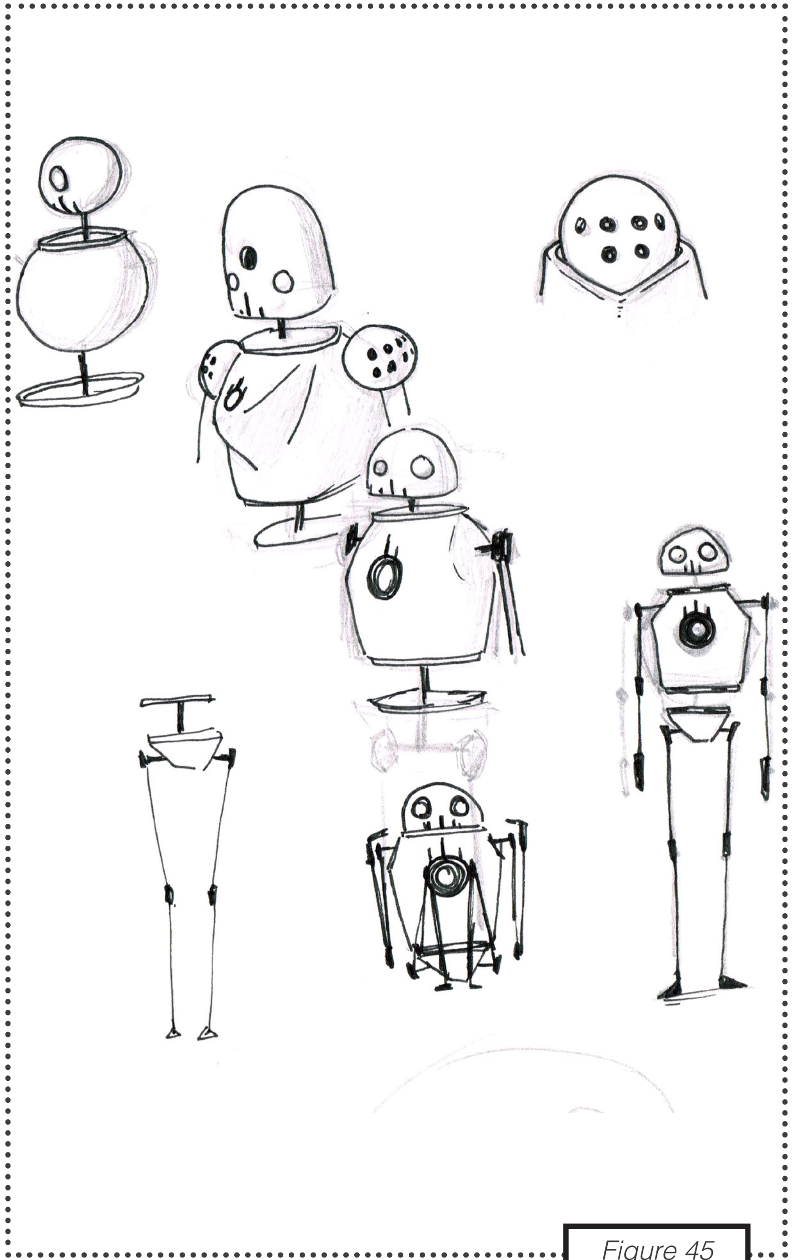


Figure 45

Story and Gameplay

As it was said in chapter 3, the story changed a lot since the beginning to now. There are some initial concepts of tribes and NPC that were deleted [Fig.46-48].

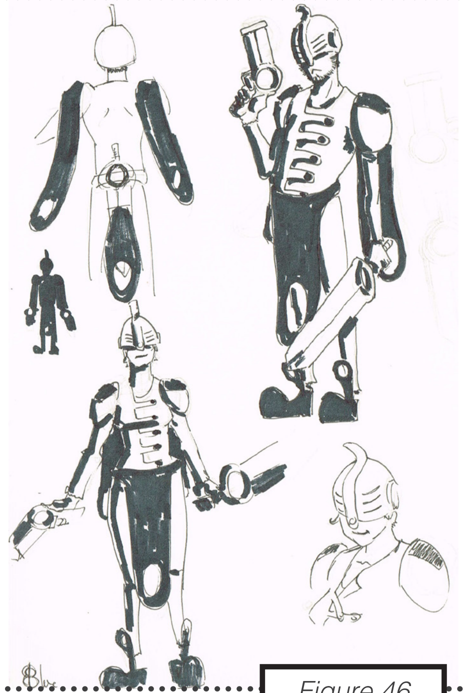


Figure 46

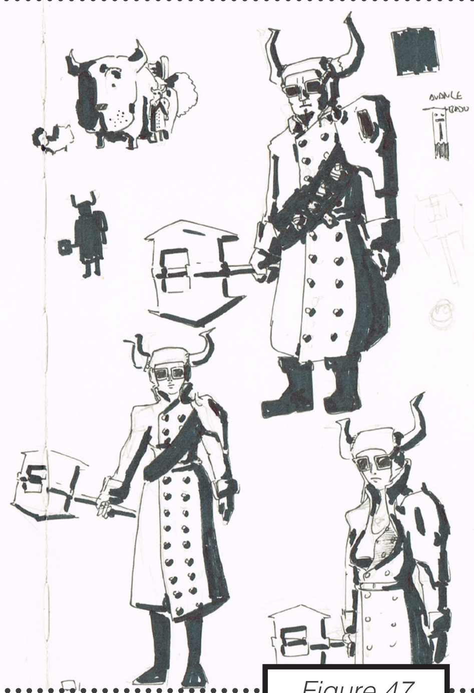


Figure 47

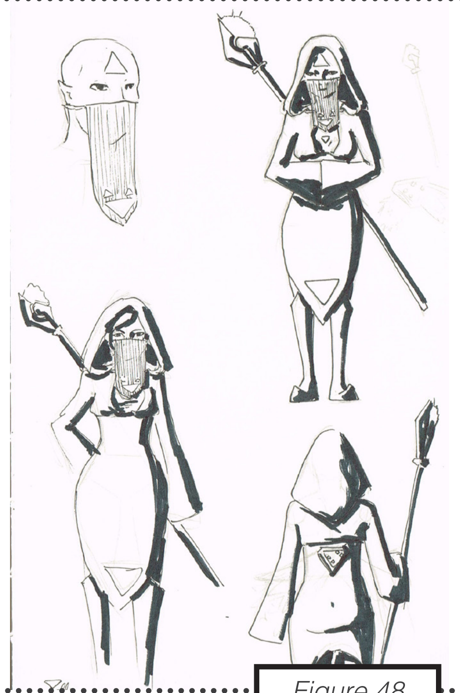


Figure 48

Figure 46: Concepts of Tribes. Desgins that helped to define the sword style.

Figure 47: Concepts of Tribes. Desgins that helped to define the axe style.

Figure 48: Concepts of Tribes. Desgins that helpedw to define the spear style.

As it was said in previous chapters, the combos changes a lot from the first idea. The evolution of the sword and spear combos can be seen in the next images [Fig.49].

Figure 49: Evolution of the designs of the sword and spear combos. It can be seen better in the section "Big Figures".

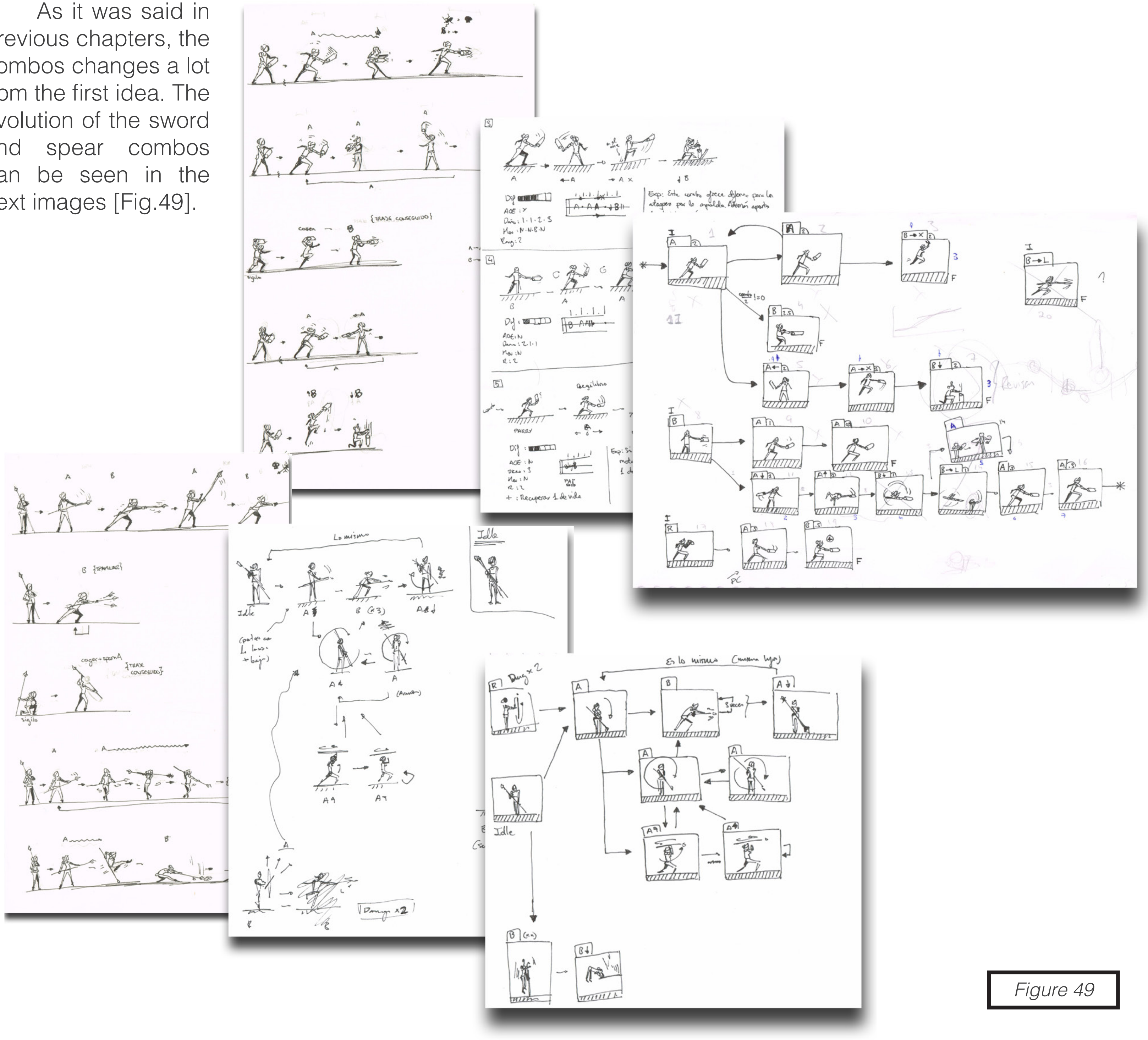


Figure 49

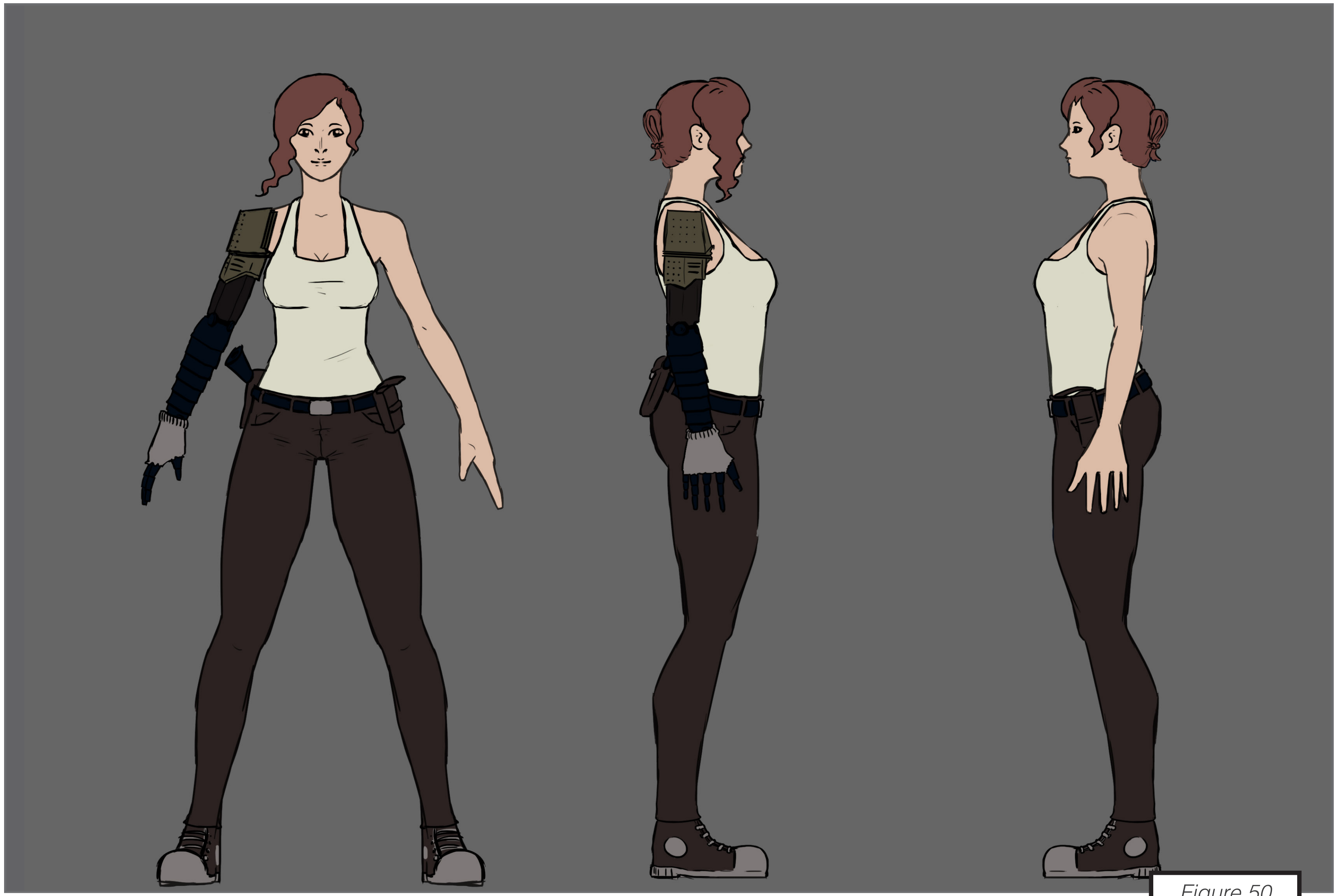
MODELING

The project has some 3D models (all made with the program 3D Max)

Erin

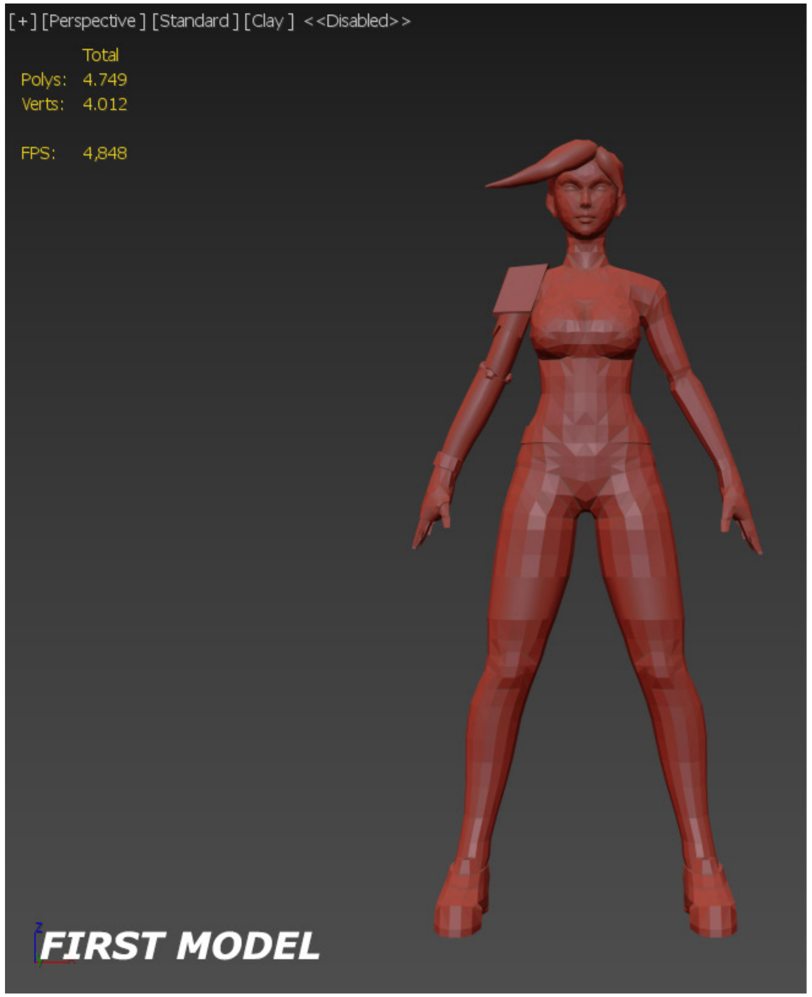
The next modeling reference [Fig.50] was based on Katarina's character of *League of Legends* [21].

Figure 50: Reference to model Erin.



The model has some changes during the time. In the next figure can be seen the first and the last model [Fig.51-52]. The main change was the robotic arm.

Figure 51: One of the first Erin's model and the final model, without textures.



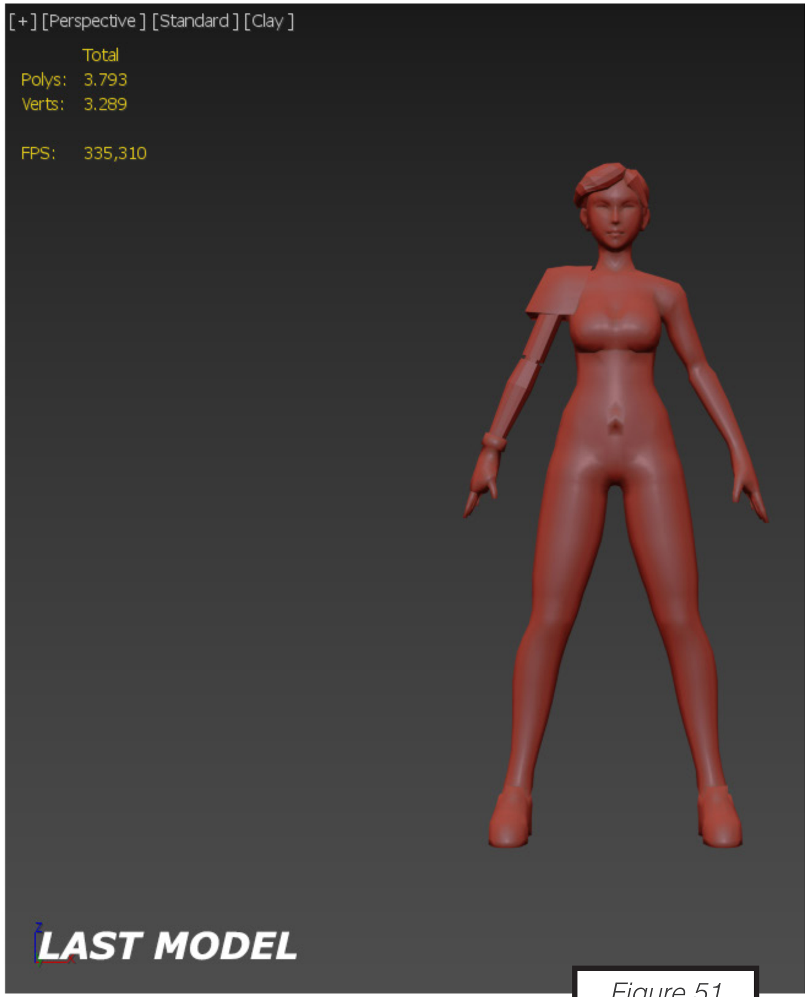


Figure 51

MODELING

Figure 52: Final Erin's model with textures. At the right side its unwrap.

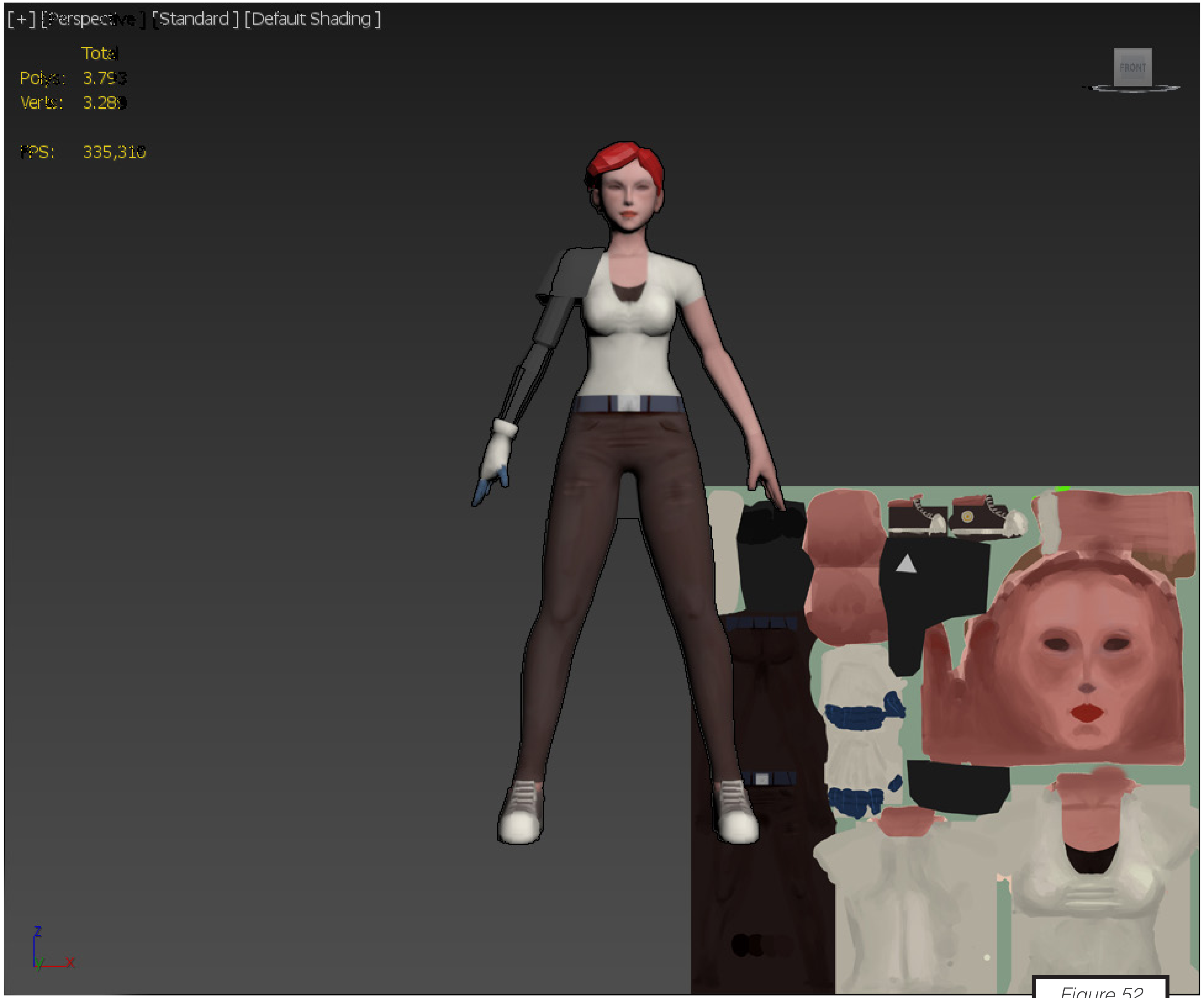


Figure 52

Robot

The robot didn't generate all the problems as Erin generated. It was simpler and direct. First of all is the reference image to model. And then, the model. The texture is a simple plane colour. The rest of colours are lights implemented with Unity3D [Fig.53-54].

Figure 53: Reference to model the robot.

Figure 54: Final model, without textures.

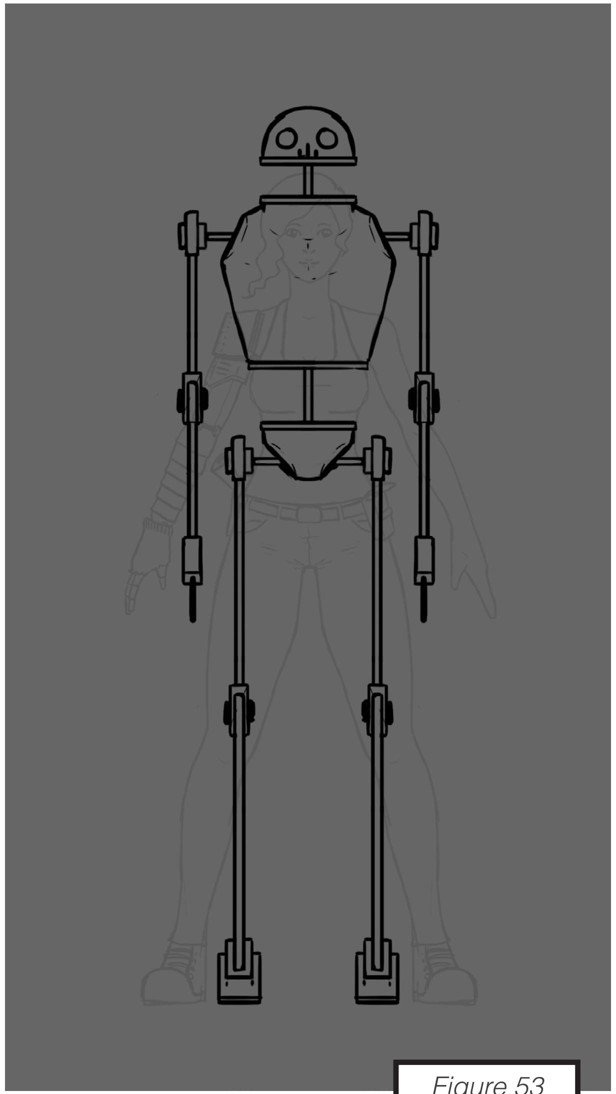


Figure 53

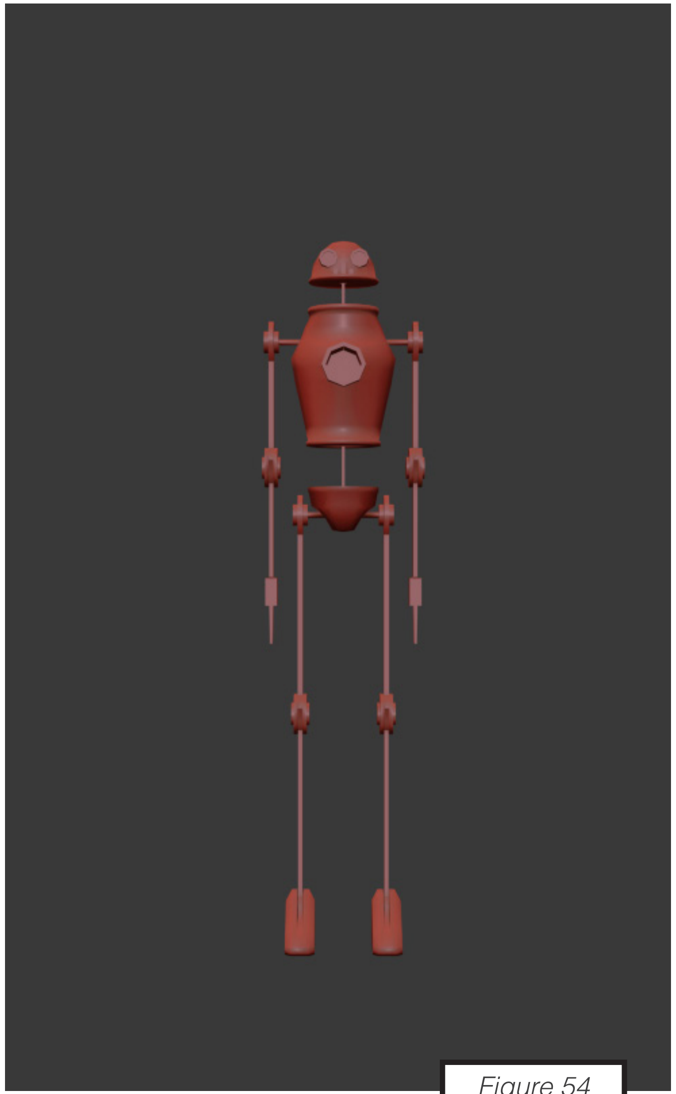


Figure 54

Weapons

The weapons were easy to do too. In the next figures can be seen the reference and the models [Fig.55-56].

Figure 55: Reference to model the weapons.

Figure 56: Final models, without textures.

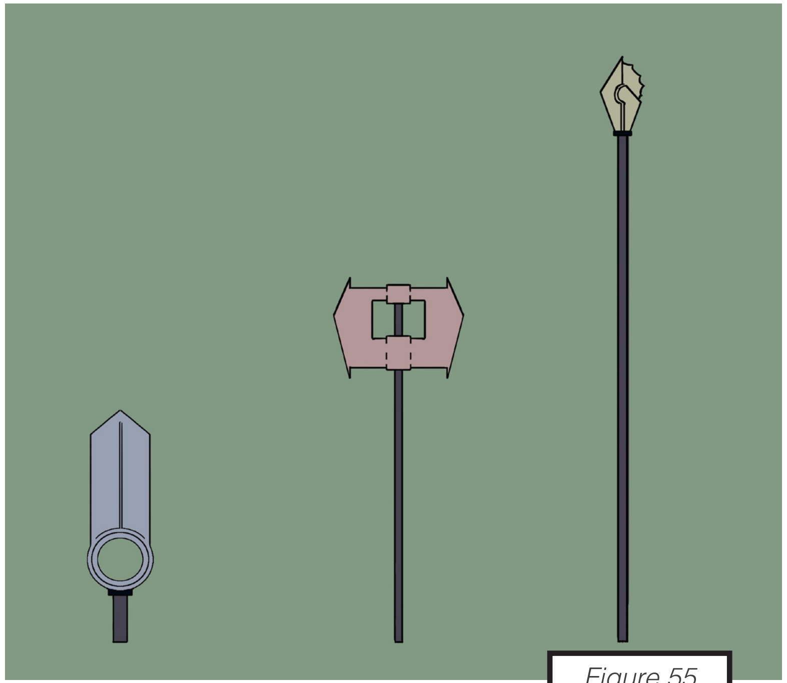


Figure 55

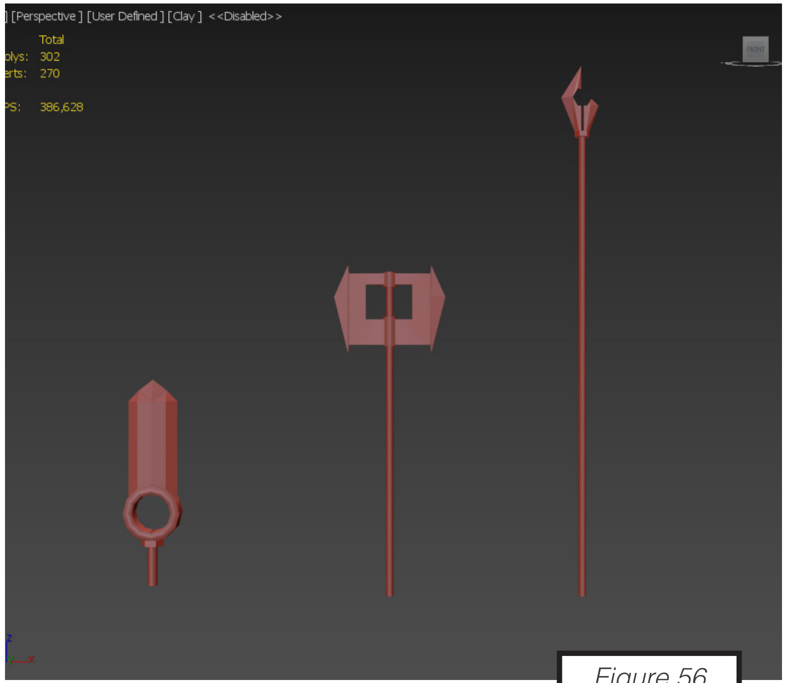


Figure 56

ERIN'S ANIMATIONS

Erin is rigged using the bones option of 3D Max [Fig.57].

Many Erin animations have conceptualized in 2D too [Fig.58].

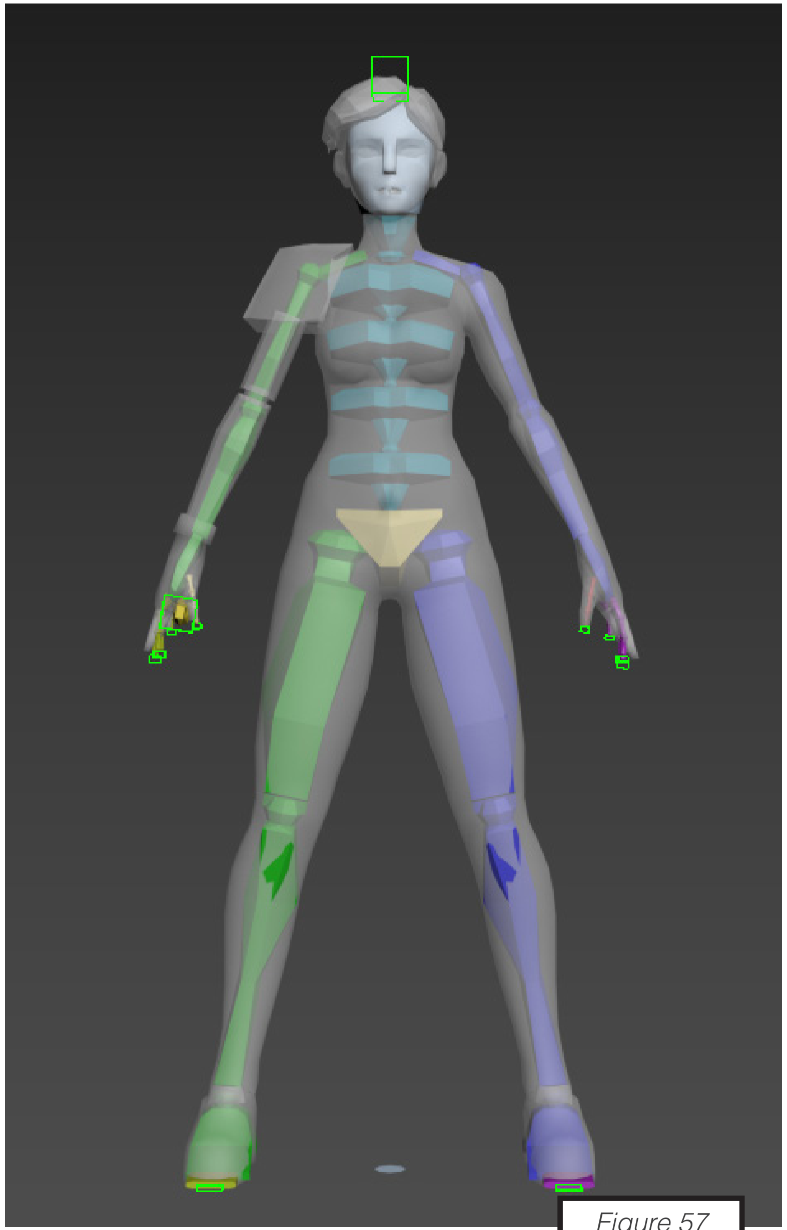


Figure 57

Figure 57: Erin's model with her bones visibles.

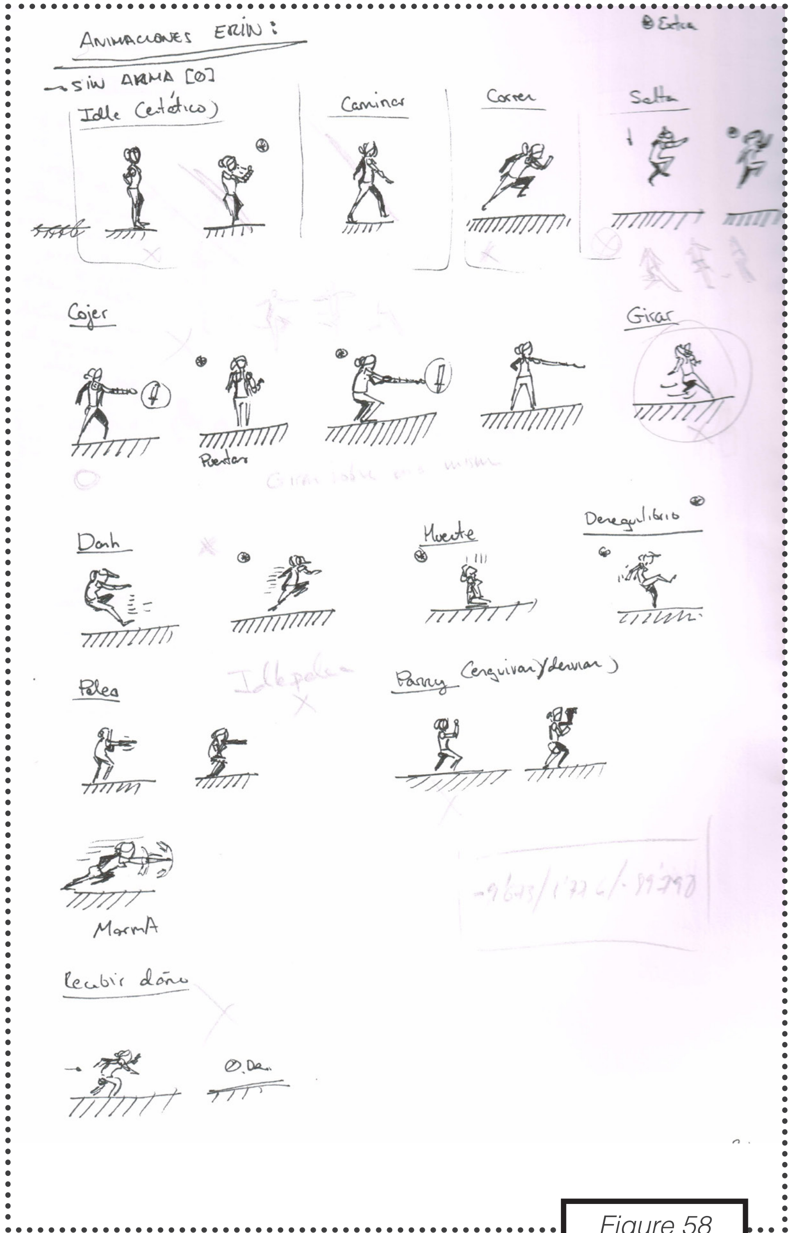


Figure 58

Figure 58: Concepts of animations.

The animations evolved from paper to the game. In the next figure can be seen the evolution [Fig.59].

Figure 59: Evolution of an Erin combo. From paper to computer.

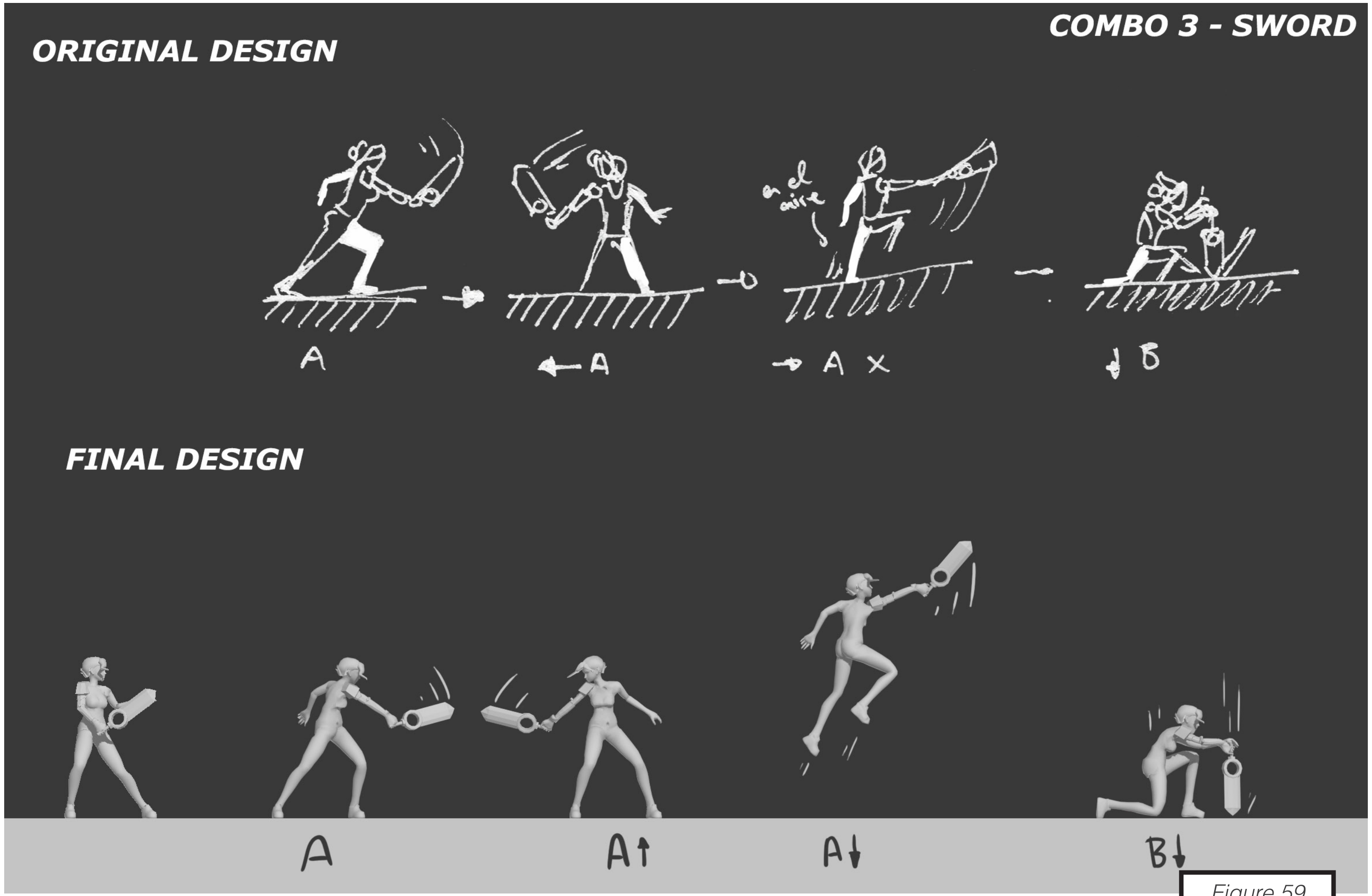


Figure 59

ERIN'S ANIMATIONS

ROBOT'S ANIMATIONS

The robot is rigged using the bones option of 3D Max [Fig.60].

The robot animations were born directly of the combos [Fig.61-62].

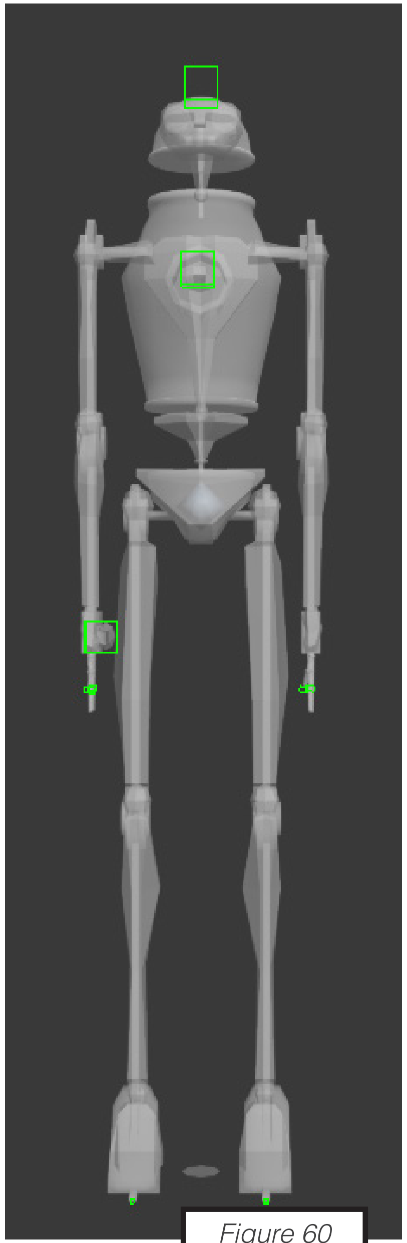


Figure 60: Robot's model with its bones visible.

Figure 60

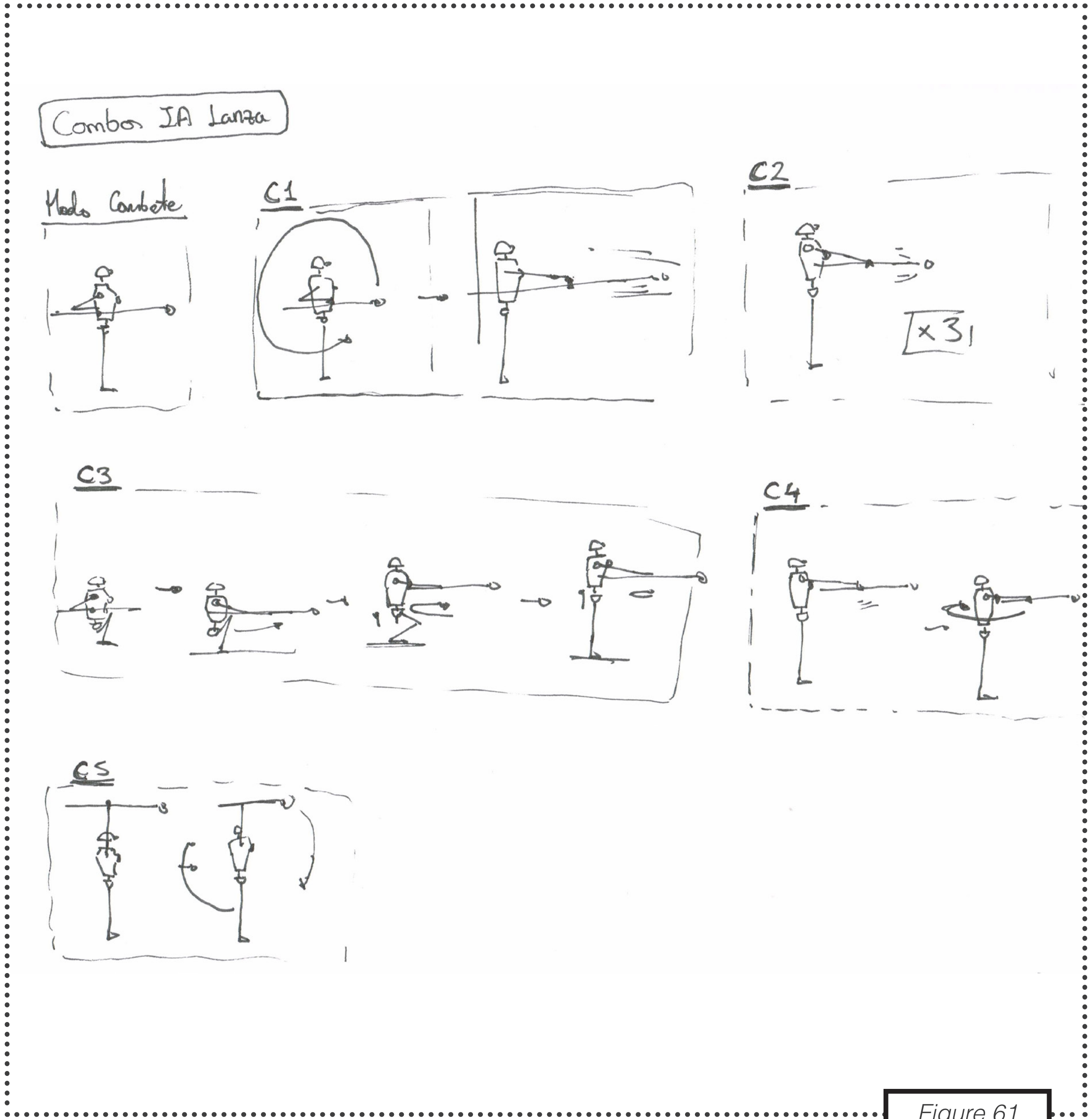


Figure 61: First concepts of the spear combos to the AI.

Figure 61

Figure 62: Evolution of a Robot combo. From paper to computer.

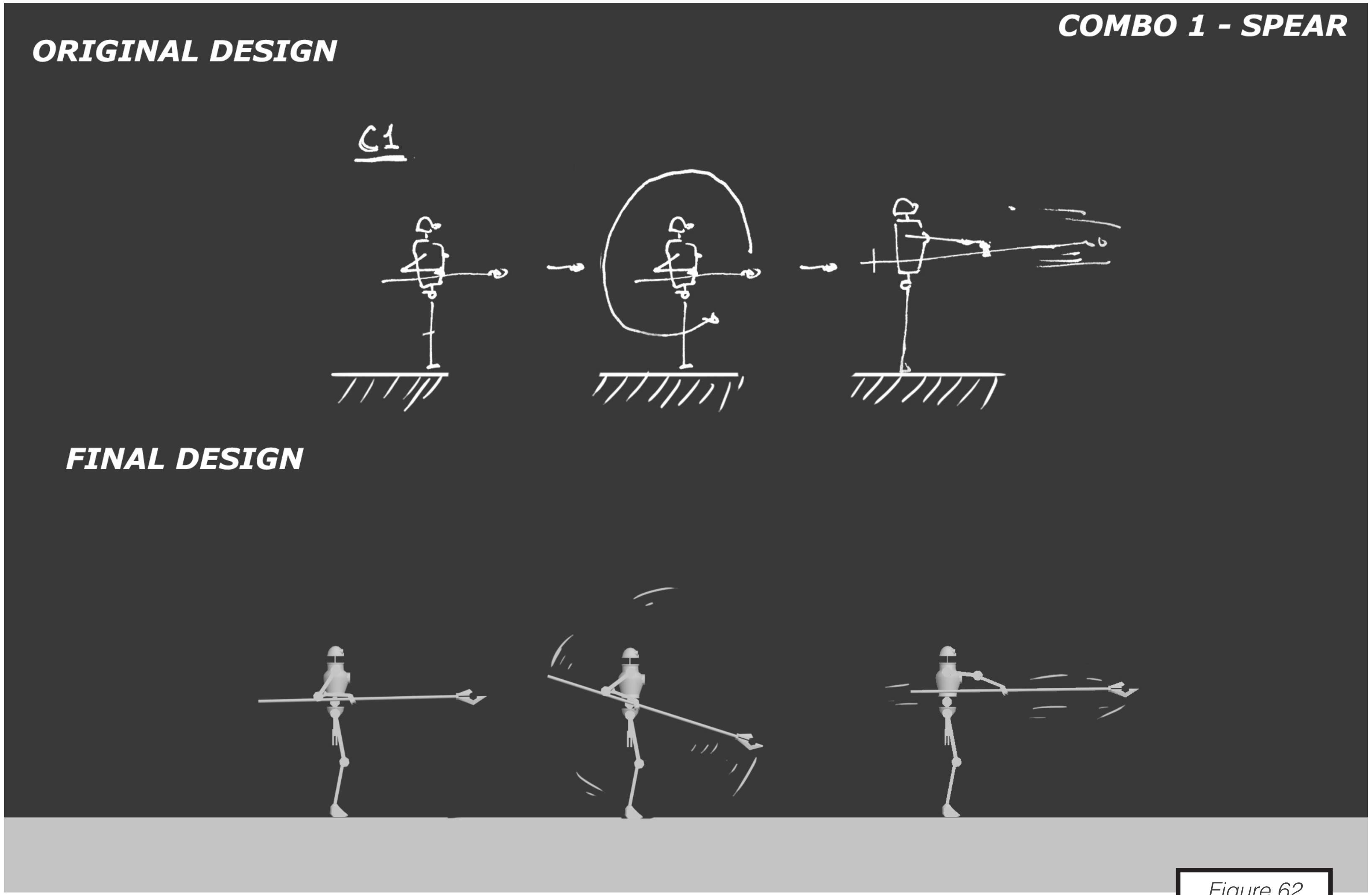


Figure 62

ABSTRACT

This chapter gathers the game results and the conclusions. It can be seen screens-hots from the final game and a compari-son of the expected objectives result and the final result on it.

5 RESULTS

RESULTS

As a result we have a game with a simple playability but with several possibilities. That force the player to adapt himself continuously to the obtained weapons, changing the gameplay [Fig.63].

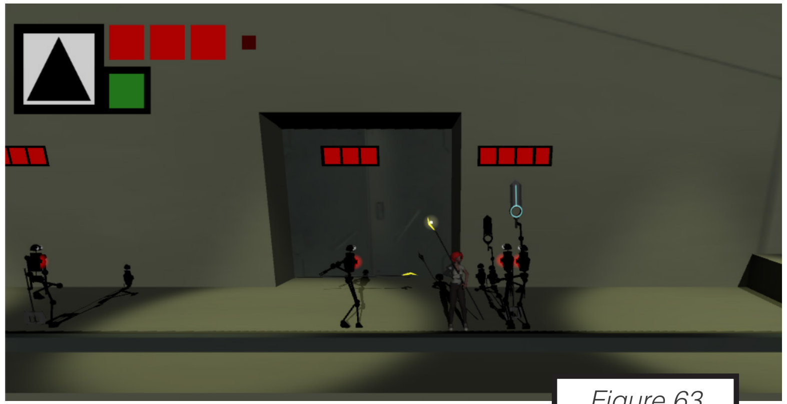


Figure 63

Figure 63: Screenshot of the game. Erin surrounded by enemies.

The game is composed by a simple Menu scene in which can be seen the controls and go into the arena scene [Fig.64]. In the arena the player fights versus enemies waves to be defeated. This waves are increasingly more difficult to be passed, until arriving to the tenth wave. Then the waves repeat the number of enemies, but change their weapons.

The game have four weapons: punch, sword, axe and spear. Every weapon has their own combos, life, animations and combat style [Fig.65]. This gives the player a lot of gameplay options. As future plans, if it is decided to continue with the project, it will be introduced a new weapon and more combos for all the weapons. This hasn't been done because lack of time.

The kinesthesia of the game is an important part that has been done too. The controls have been distributed in a way that the player has the actions organized in zones. There are differents particles that do easier the knowledge about what is happening on the screen, too. Also, the game responds with sounds and vibrations (if the controller is connected) to some important events (especially related with the damage perform).

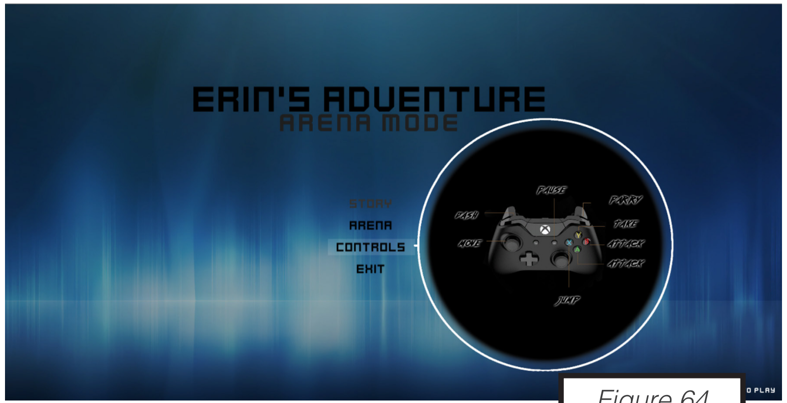


Figure 64

Figure 64: Screenshot of the game. Main menu and controls.

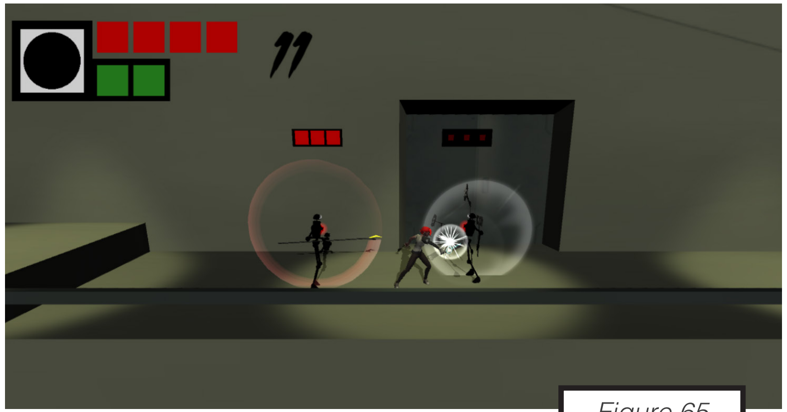


Figure 65

Figure 65: Screenshot of the game. Erin killing a robot.

The project has a lot of animations that help to give dynamism to the game. Finally, the animations haven't been so fluids as it was planned in the very beginning. This is because the Unity3D transactions haven't been correctly managed. In addition, about the combos part, has been compulsory to cut the final combo step animation to give the player enough time to continue easily the combo. This cuts the animations and the fluidity but it benefits the gameplay

The artificial intelligence (AI) is very solid in spite of its simplicity [Fig.66]. The AI was done following three determined guidelines that had been accomplished. It had to:

- Make predictable actions but with unexpected results. It is achieved with a simple behaviours. The unexpected part is achieved telling to the AI to do a random combo when it is going to attack. The player will know that the robot will attack, but he/she will not know with which combo.

- Answer what the player does. It is achieved by the distance system (if the player is far away, the AI will not move) and through the damage reception. If the AI receives damage, it will go back and its actual actions are delayed.

Figure 66: AI Action diagram.

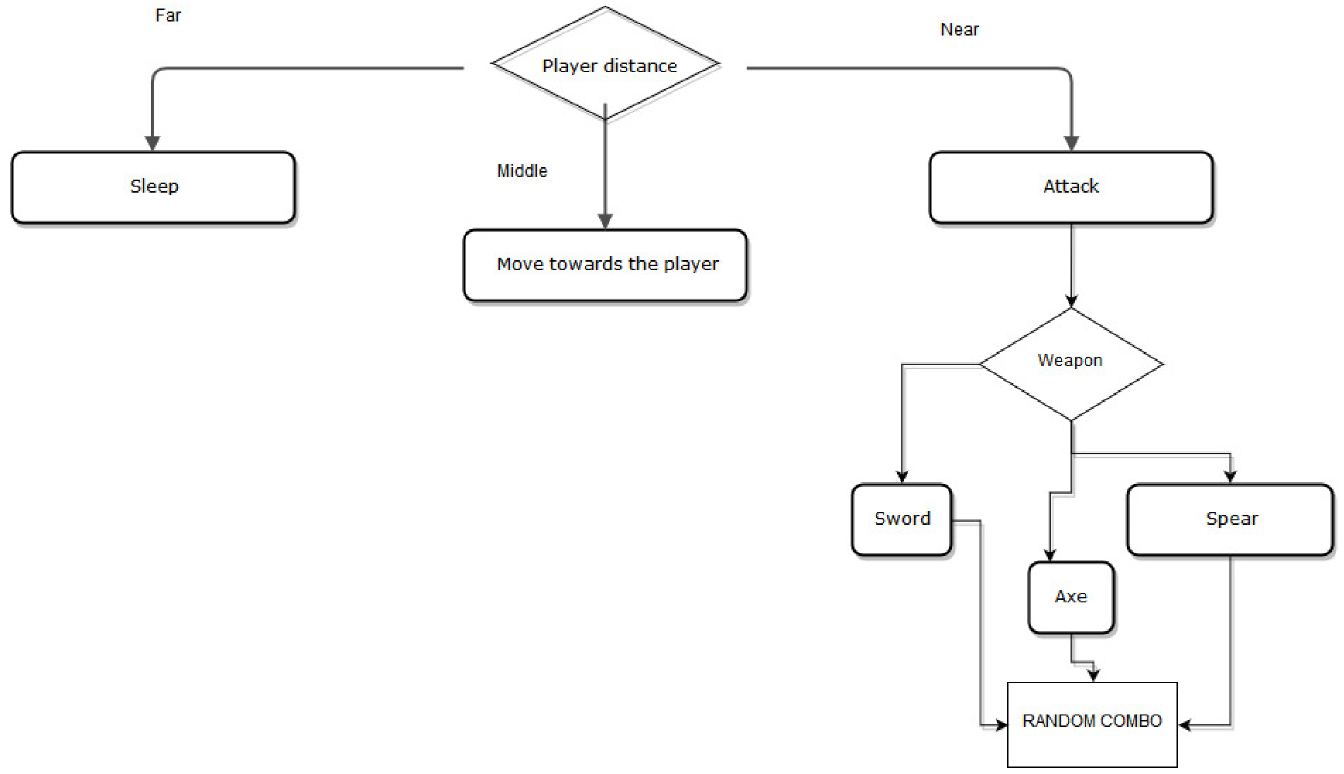


Figure 66

- Use the same system of the player. The game simulated that it is like that, but really it isn't. The enemies use a weapons system and combo system like the player. It gives the expected sensation. But really, the AI weapons can't be broken and its combo system works in a different way.

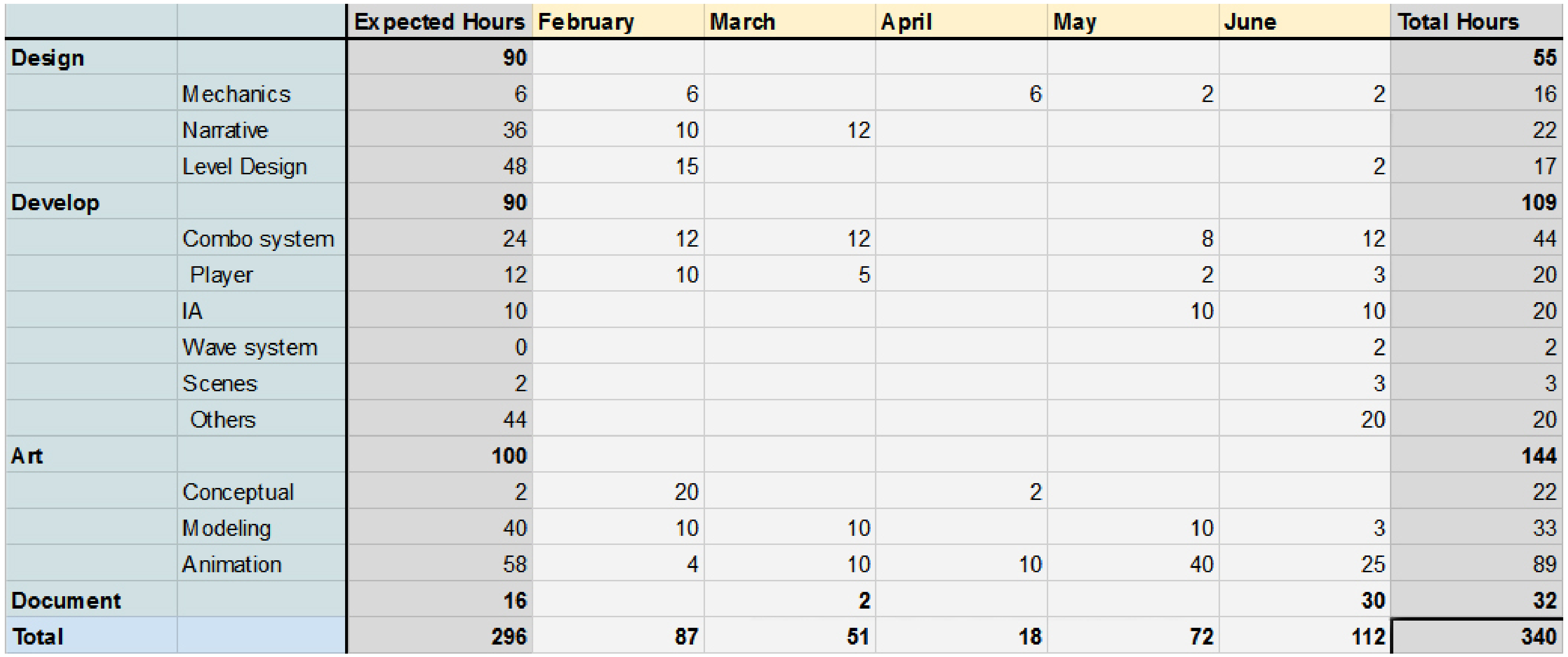
RESULTS

The project has ended up being a solid game with a funny arena mode. In it the player can challenge himself (or herself) versus a big number of robots in a fair fight.

If you want to try the game or see some gameplay you can get it from the following link:

LINK TO THE GAME

PROJECT DEVIATIONS



		Expected Hours	February	March	April	May	June	Total Hours
Design		**90**						**55**
	Mechanics	6	6		6	2	2	16
	Narrative	36	10	12				22
	Level Design	48	15				2	17
Develop		**90**						**109**
	Combo system	24	12	12		8	12	44
	Player	12	10	5		2	3	20
	IA	10				10	10	20
	Wave system	0					2	2
	Scenes	2					3	3
	Others	44					20	20
Art		**100**						**144**
	Conceptual	2	20		2			22
	Modeling	40	10	10		10	3	33
	Animation	58	4	10	10	40	25	89
Document		**16**		**2**			**30**	**32**
Total		**296**	**87**	**51**	**18**	**72**	**112**	**340**

Figure 67

Figure 67: An excel table with the dedicated hours and the expected hours

CONCLUSIONS

Objectives had been completed but not as well as it was imagined:

- Playable demo: It has been done and it requires to the player some effort to overcome.

- Achieve fluency in combat and animations: The combat is fluid and it has a lot of animations. But it hasn't been as it was planned. While the player is fighting he/she can't move. It rest dynamism to the gameplay. Furthermore there are always specific combos that works better than others. It does that some of the combos hardly ever used.

- Easy and comfortable to play: It has been also aimed, thanks to the game's kinesthesia and the controls position.

During the project I have learnt more things that I thought I was going to learn:

- I have learned to model and animate characters correctly. Furthermore I learned to export this to Unity3D. With all that, I have created my own workflow in the animation.

- I have learned to do particles systems in Unity3D. I haven't done this before and I have realized that it is a thing that I like a lot.

- I have learned to plan projects in a more realistic way. This has been achieved after my first fail in the planification.

- I have learned to create extensible game mechanics. With this I want to say that actually it will be quite simple to amplify the actual mechanics, weapons and combos of the project.

The production of this project has taken me to the classic curve of illusion [Fig.67]:

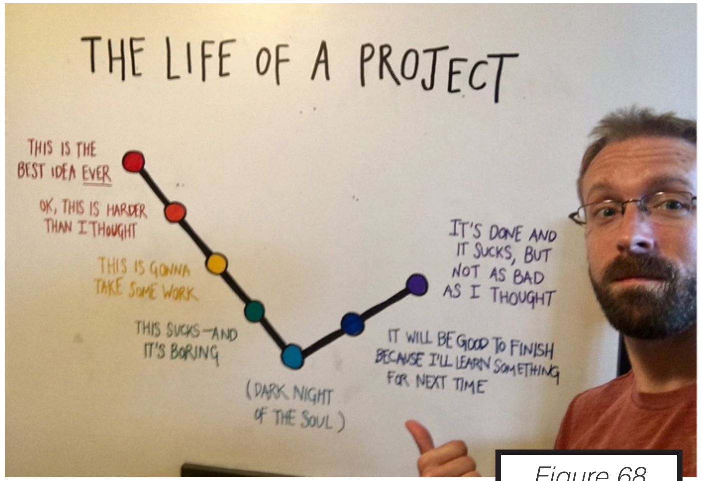

THE LIFE OF A PROJECT
THIS IS THE BEST IDEA EVER
OK, THIS IS HARDER THAN I THOUGHT
THIS IS GONNA TAKE SOME WORK
THIS SUCKS — AND IT'S BORING
(DARK NIGHT OF THE SOUL)
IT WILL BE GOOD TO FINISH BECAUSE I'LL LEARN SOMETHING FOR NEXT TIME
IT'S DONE AND IT SUCKS, BUT NOT AS BAD AS I THOUGHT
Figure 68

Figure 68: The Life of a project by Jonah Lobe.

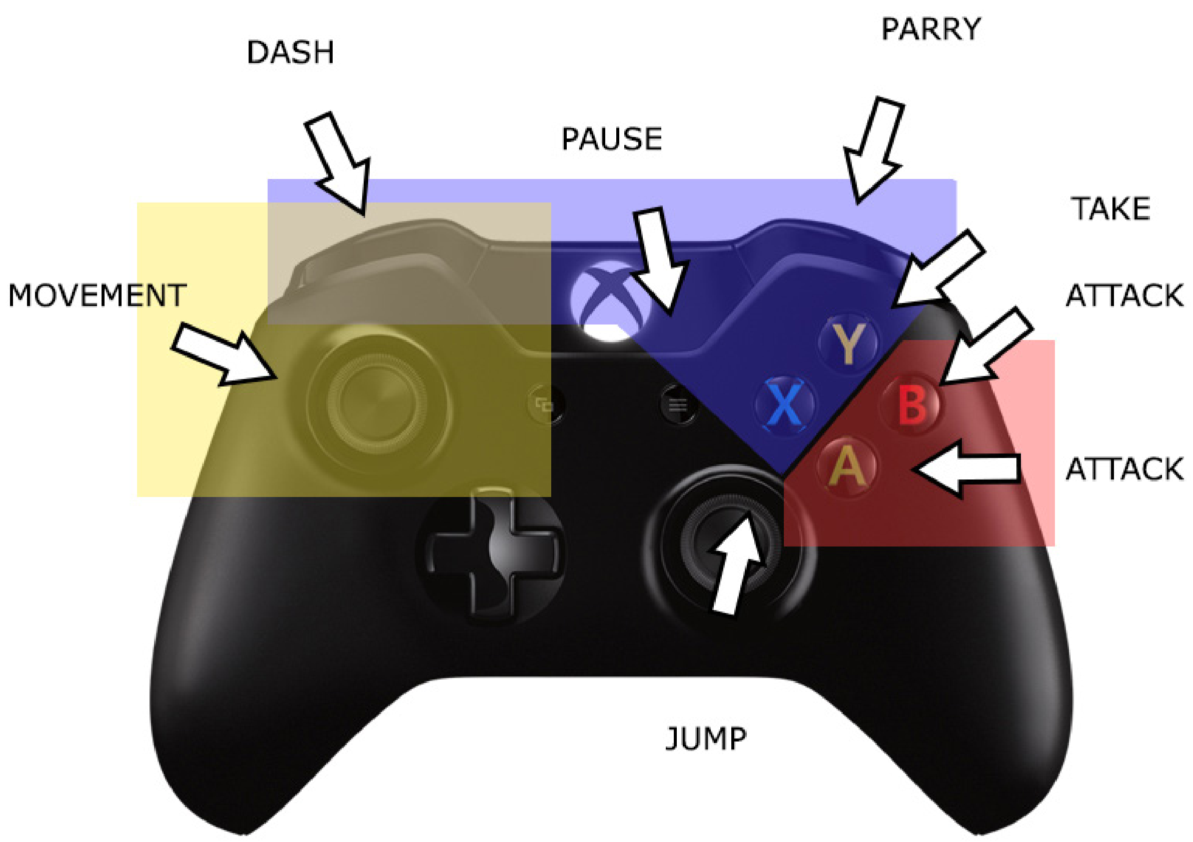
But, now that it is finished, I'm proud with the results. In spite of the stress, the lost of hours of sleep and the headaches, all have been compensated with all the learned things and the self- realisation that entails to do this kind of project. Nowadays, I can still see a lot of lucks and things to be improved in the game (mentioned majority in this memory). But this nonconformity tells me that I have capacities to continue improving, capacities to go forward.

Finally, give thanks to the people who have helped me to do this project, specially to my family (highlighting my brother in the memory layout and my mother with english).

BIG FIGURES

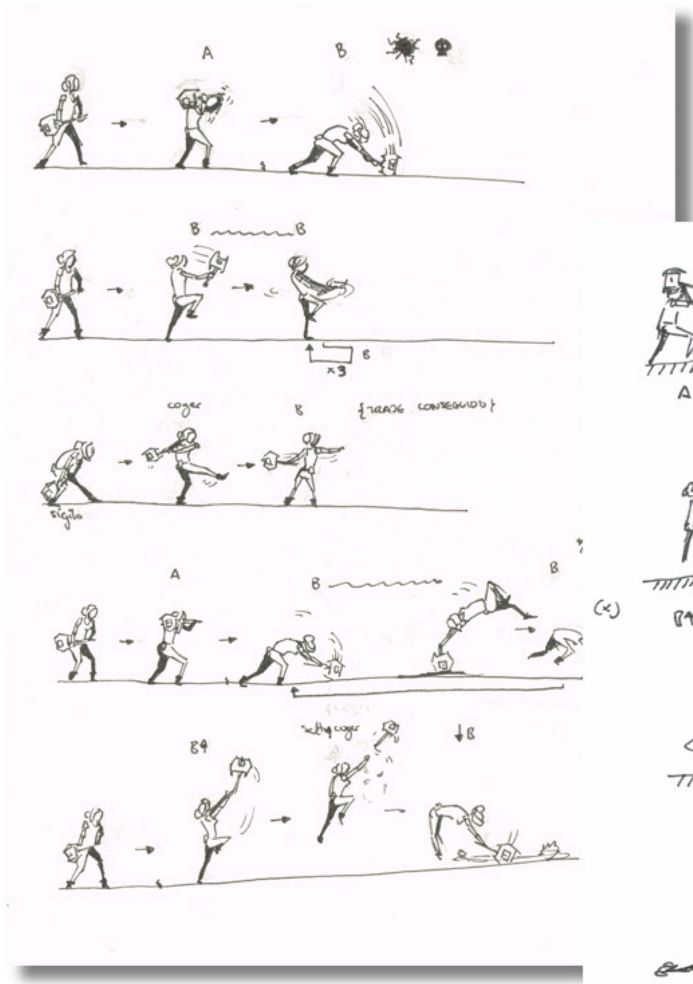
In the next pages can be better seen some figures that have been showed smaller in previous pages. All that figures have in their descriptions the sentence "*It can be seen better in the section "Big Figures"*".

Figure 06: Controls of the game over a Xbox Controller.

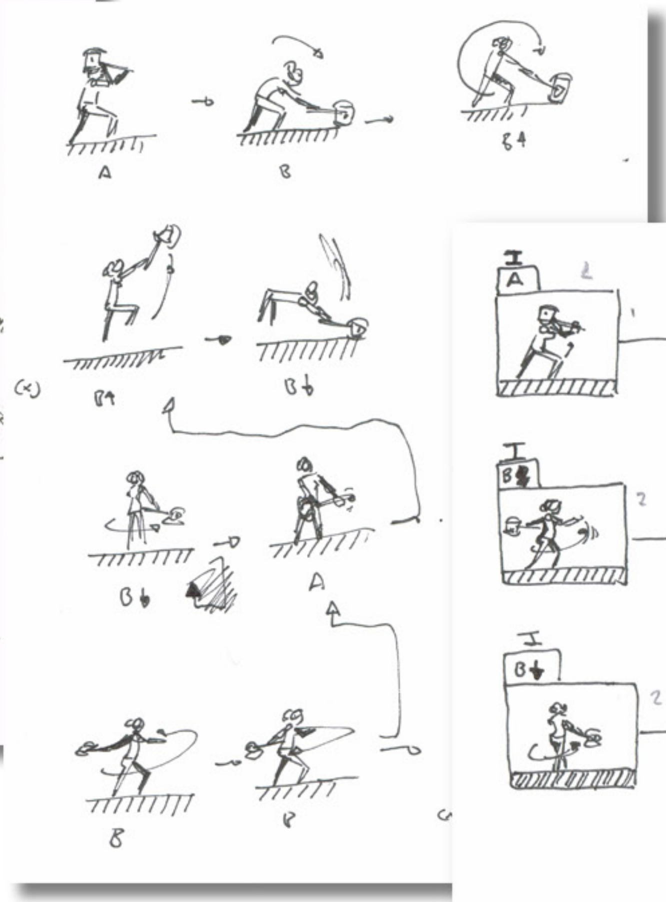


🟨	Player’s movements
🟦	Player’s actions
🟥	Player’s attack

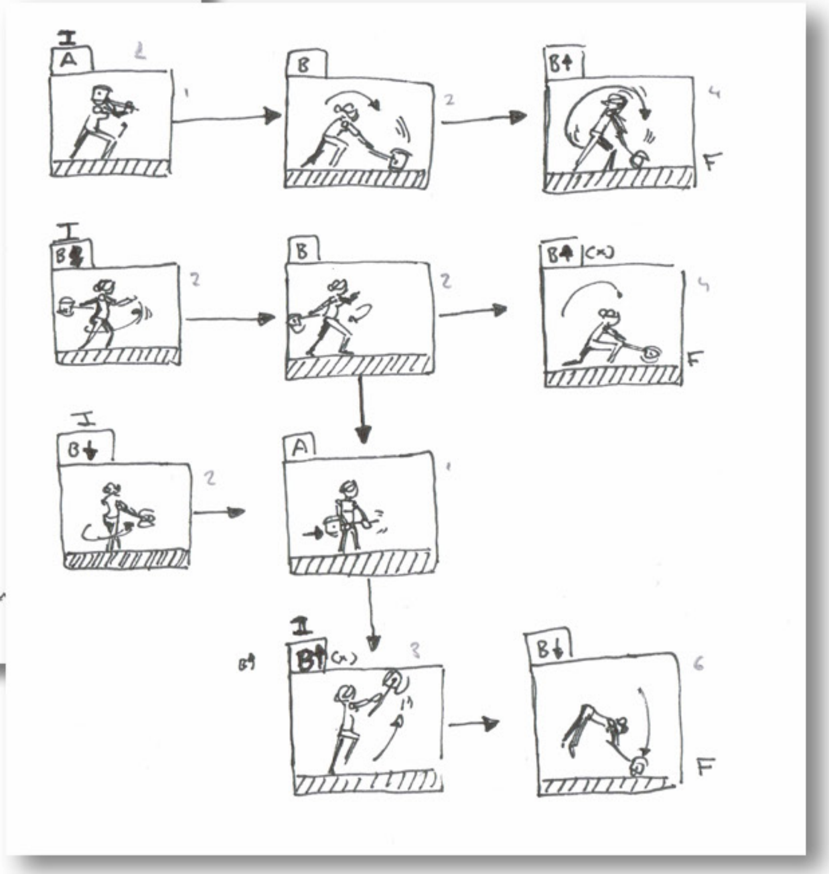
Figure 21: Evolution of the axe combos.



Primera version

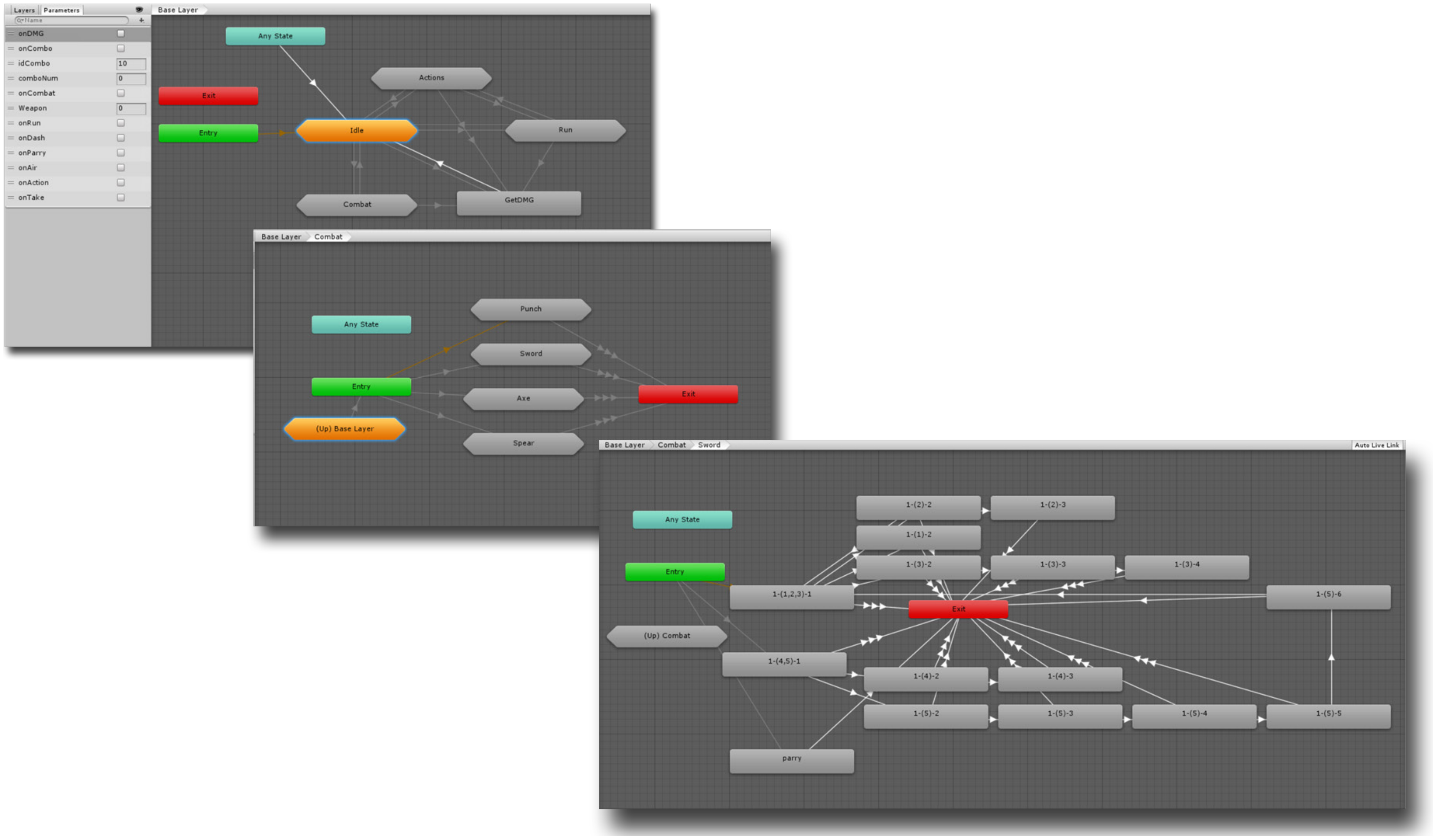


Segunda version



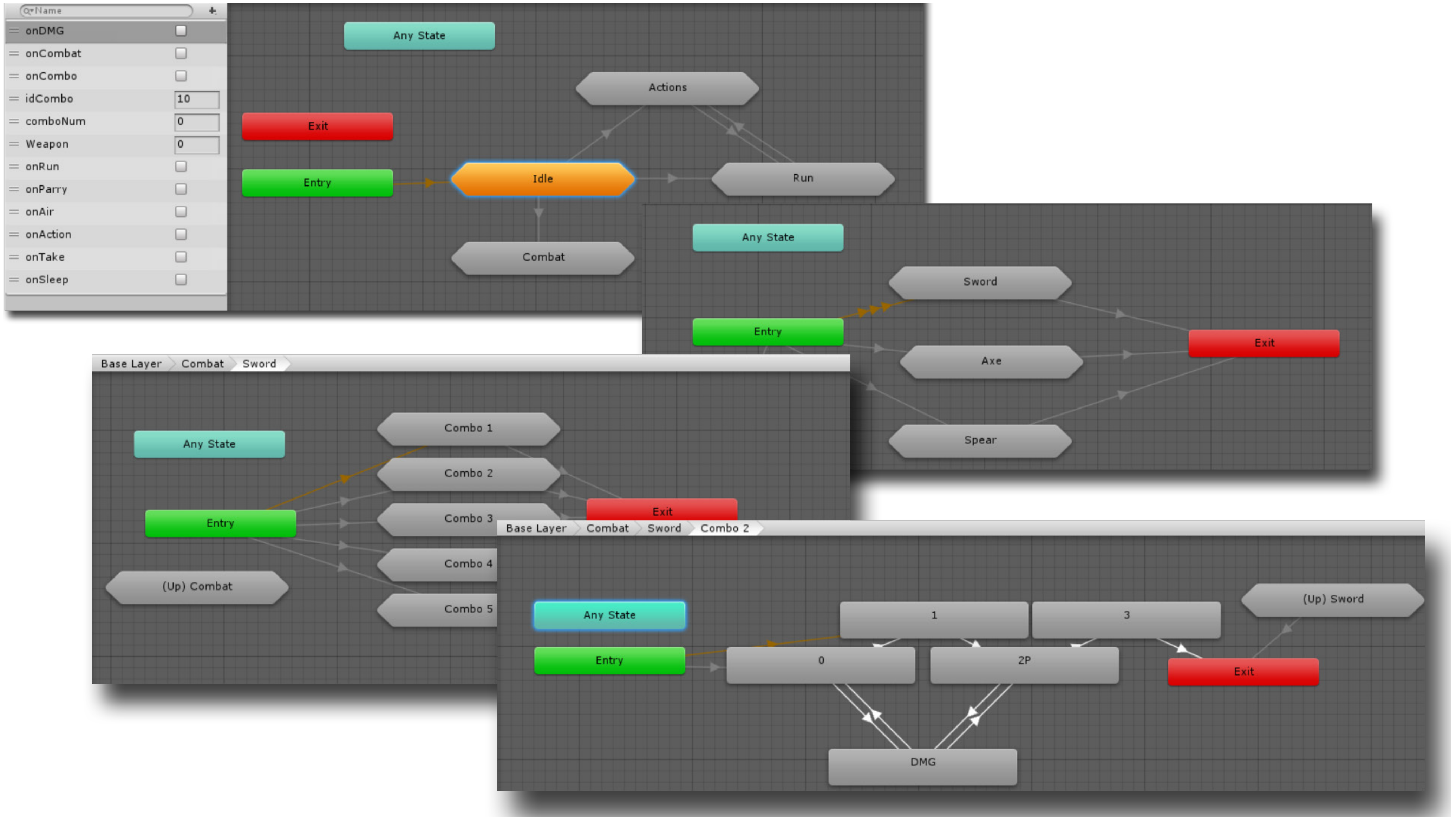
Version final

Figure 24: Differents states and substates on the Animator of the main character.



onDMG	☐
onCombo	☐
idCombo	10
comboNum	0
onCombat	☐
Weapon	0
onRun	☐
onDash	☐
onParry	☐
onAir	☐
onAction	☐
onTake	☐

Figure 26: Differents states and substates on the Animator of the enemies.



onDMG	☐
onCombat	☐
onCombo	☐
idCombo	10
comboNum	0
Weapon	0
onRun	☐
onParry	☐
onAir	☐
onAction	☐
onTake	☐
onSleep	☐

Figure 49: Evolution of the designs of the sword and spear combos. (I)

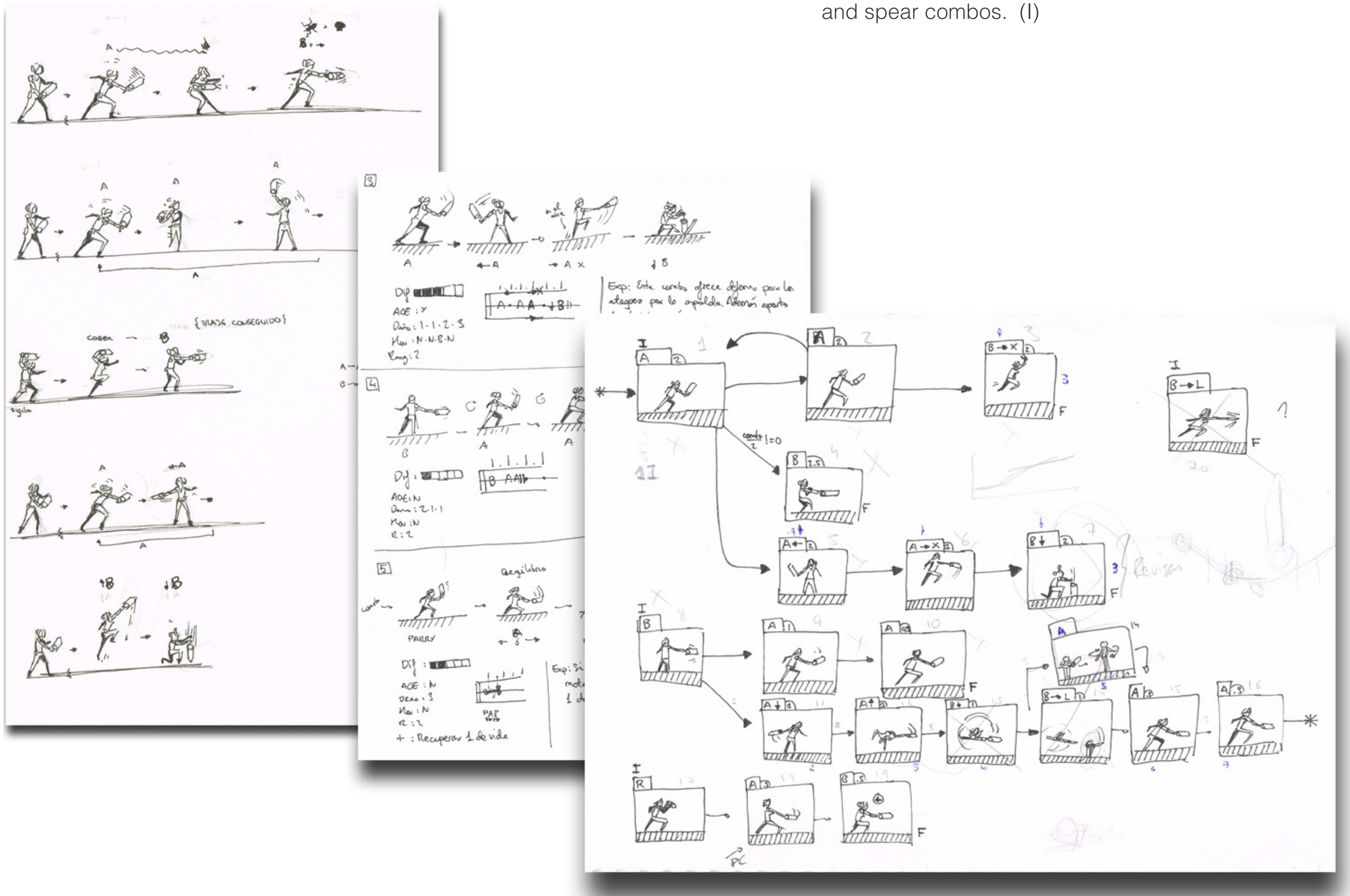
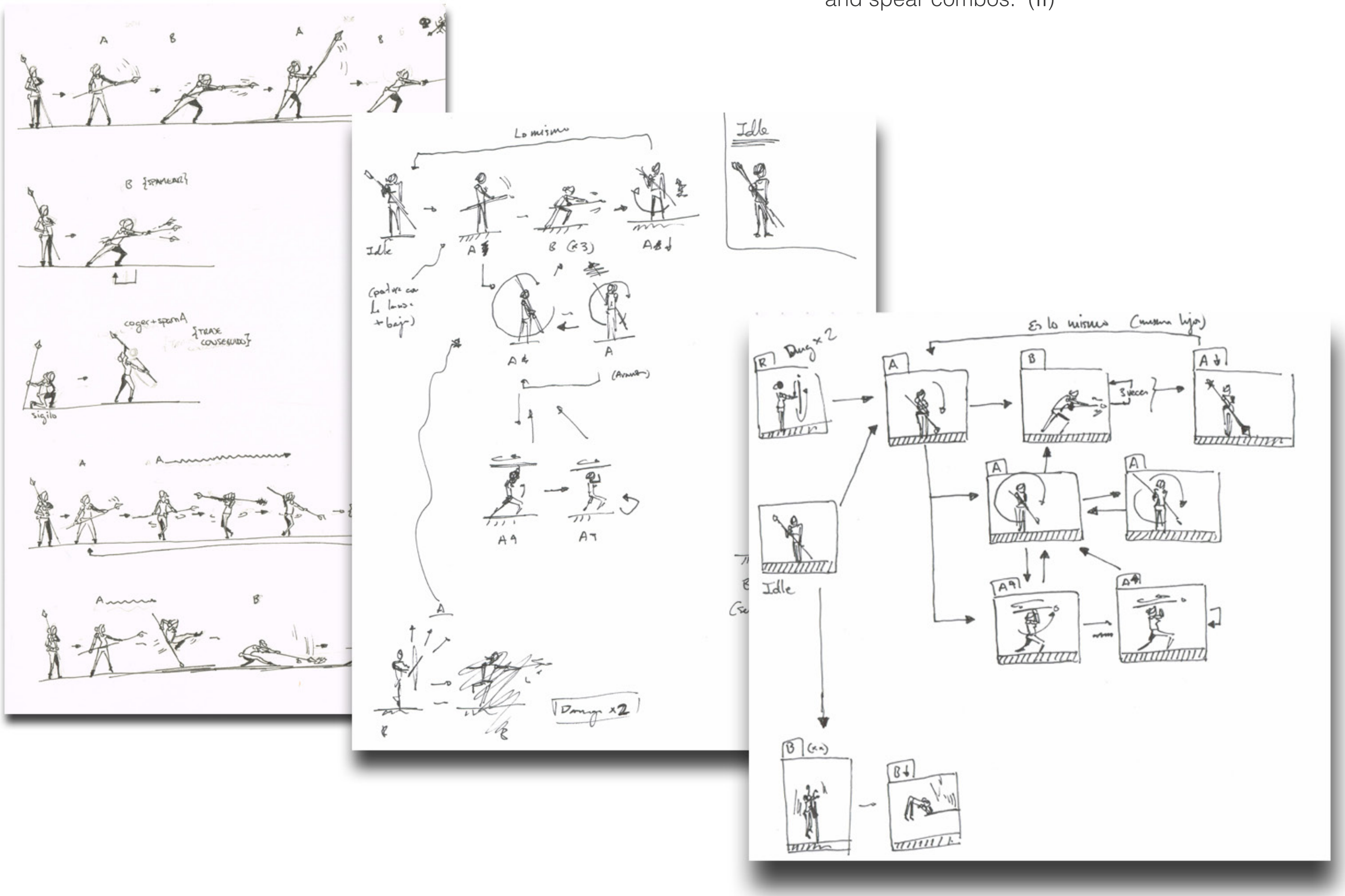


Figure 49: Evolution of the designs of the sword and spear combos. (II)



REFERENCES

[1]: Skull Girls. http://skullgirls.com/es/

[2]: Injustice 2. https://www.injustice.com/

[3]: Street Fighter. http://streetfighter.com/?lang=es

[4]: Hollow Knight. http://hollowknight.com/

[5]: The Legend of Zelda: Breath of The Wild. http://www.zelda.com/breath-of-the-wild/

[6]: This war of mine. http://www.11bitstudios.com/

[7]: Assassin's Creed Chronicles: China. https://www.ubisoft.com/es-es/game/assassins-creed-chronicles

[8]: Hyper Light Drifter. http://www.heart-machine.com/

[9]: NewGrounds. http://www.newgrounds.com/audio/

[10]: freeSFX. http://www.freesfx.co.uk/

[11]: freesounds. https://freesound.org/

[12]: Animator controller. https://docs.unity3d.com/Manual/Animator.html

[13]: Brackeys. https://www.youtube.com/watch?v=-Vrld13ypX_I

[14]: Sirhaian'Arts. https://www.youtube.com/watch?v=0YlBinFC9So&t=889s

[15]: imn nam. https://www.youtube.com/watch?v=wFgS5pzG1Qs&index=1&list=PLdLWcjzv7wtHVR8NFwrRdWFSNfLHDhsTA

[16]: The Legend of Zelda: The Wind Waker. http://zelda.com/windwaker/

[17]: Limbo. http://playdead.com/

[18]: The Witcher 3: Wild Hunt. http://thewitcher.com/en/witcher3

[19]: Star Wars. http://www.starwars.com/

[20]: ART 88/44. http://art88-46.subcultura.es/

[21]: League of Legends. http://euw.leagueoflegends.com/es