

Rapport de projet : Hunter x Hunted

Sacha HIBON, Matthieu CAMART, Arthur BARBIER
et Ronan PEDRON

Promo 2025: Projet S2



FIGURE 1 – Logo SMAR



FIGURE 2 – Logo Hunter X Hunted

Table des matières

1	Introduction	4
1.1	Comment notre groupe s'est-il formé ?	4
1.2	Origine du Nom et du Logo	4
1.3	Qui sommes-nous ?	4
1.3.1	Sacha HIBON	4
1.3.2	Matthieu CAMART	5
1.3.3	Arthur BARBIER	5
1.3.4	Ronan PEDRON	5
1.4	Origine et nature de notre projet	6
2	Répartition des tâches	7
3	GamePlay	8
3.1	Chat et la souris	8
3.2	Déplacement	8
3.3	"Attraper"	8
4	Réseau	9
4.1	PUN	9
4.2	Lobby/Hall	9
4.3	Room/Salle	9
4.4	Personnage	9
4.5	Intelligence artificielle	10
4.6	Autre	10
5	Menu	11
5.1	Création ou connexion à un compte	11
5.2	Menu principal	11
5.3	Options	12
6	Modélisation 3D	13
6.1	Nos modèles	13
6.1.1	Les armes	13
6.1.2	Les personnages	14
6.2	La carte	16
6.3	Optimisation	17
6.4	La salle d'attente : le bar	18
7	Salle d'attente	20



8 Animation	21
8.1 L'animation des personnages	21
8.1.1 Première soutenance : la découverte et la prise en main	21
8.1.2 Deuxième Soutenance : le renouveau	22
8.1.3 Résultat finale	26
8.2 L'animation des menus	27
9 Intelligences Artificielles	29
9.1 "Bot Rectiligne"	29
9.1.1 "Maintenance"	29
9.2 Mémoire/Sauvegarde	31
9.3 Fuyard	32
9.3.1 Path Finding	33
9.4 Suiveur	34
9.5 Machine Learning	34
9.5.1 Réseau neurones	34
9.5.2 Entrainement	35
9.5.3 Sauter	36
9.5.4 Déetecter	36
9.5.5 Amélioration	37
9.6 Ajustement	37
9.7 Conclusion	37
10 Base de donnée	38
10.1 Implémentation dans le jeu	38
11 Son	39
12 Site web	39
12.1 Le menu	39
12.2 Accueil	39
12.3 Nos membres	40
12.4 Notre jeu	41
12.4.1 Download	41
12.4.2 Notre IA	41
12.4.3 Nos modèles	41
12.5 L'hébergement	41
13 Problèmes rencontrés et solutions	42
13.1 Collision	42
13.2 Taille de la carte	42
14 Page d'outils	43



15 Conclusion	46
15.1 Ce que le projet nous a apporté	46
15.1.1 Matthieu CAMART	46
15.1.2 Arthur BARBIER	46
15.1.3 Ronan PEDRON	47
15.1.4 Sacha HIBON	47
15.2 Un petit mot pour la fin	47
16 Annexes	48
16.1 Documentation	48
16.2 Autre	48

1 Introduction

Ce rapport de projet a pour but de présenter tout le travail accompli durant l'année ainsi que montrer son rendu final. Nous nous présenterons, nous exposerons nos peines, nos joies, les compétences acquises et l'évolution du nom de notre projet.

1.1 Comment notre groupe s'est-il formé ?

Depuis le début de l'année, nous avons fait connaissance tous les quatre et nous avons rapidement sympathisé. Nous nous sommes régulièrement aider lors des travaux pratiques de programmation et dans beaucoup d'autres matières. Nous nous complétons. Certains ont des difficultés en anglais alors que d'autres excellent dans la matière. Certains avaient des bases en programmation avant la S1 alors que d'autres étaient seulement à leurs premiers pas. Et enfin, une personne est experte dans l'art de la recherche d'information et de la gestion de projets. La formation de ce groupe s'est donc faite naturellement.

1.2 Origine du Nom et du Logo

Pour le nom de notre groupe et de notre projet, nous avons beaucoup parlé car nous voulions un nom qui nous représenterait. C'est pour cela que notre nom de groupe n'est d'autre que l'initiale du prénom de chacun de ses membres (Sacha, Matthieu, Arthur et Ronan). Pour le coup, le nom du groupe SMAR nous a mis tout de suite d'accord car il est simple et efficace. C'est pour le nom du jeu que nous avons beaucoup hésité. Nous avions choisi "SMAR Game 1" et il a fini par évoluer en "Hunter X Hunted". Enfin pour le logo, même si l'inspiration peut sembler évidente, nous sommes partis du logo de la SNCF et nous avons remplacé les lettres par celles de notre groupe !

1.3 Qui sommes-nous ?

1.3.1 Sacha HIBON

J'ai toujours aimé tout ce qui touchait à la création, la conception. Des «Kapla» à l'application “SolidWorks” en passant par le jeu “Minecraft”. J'ai fini par m'intéresser à des robots programmables via scratch. Composé de dix servomoteurs, je pouvais le remodeler à l'infini afin d'entreprendre différents desseins comme un robot danseur, un robot shooter ou encore un robot évoluant dans un labyrinthe. Enfin, depuis ma terminale, l'informatique fait partie de mon quotidien, c'est une source inépuisable de développement. J'ai pu implémenter sur python un Pacman, un puissance 4, ma propre bibliothèque de lettres sur tkinter. En définitive, j'aime reproduire les jeux vidéo en partant d'une feuille blanche et apporter ma touche personnelle.



1.3.2 Matthieu CAMART

Depuis tout petit, je suis un passionné de tous les types de jeux : de construction, de logique mais aussi de jeux vidéo. J'ai découvert le véritable univers de l'informatique à mes 10 ans, lorsque j'ai vécu au Japon. J'ai tout de suite appris à utiliser des logiciels tels que SketchUp et Scratch me permettant ainsi de faire plus de choses qu'avec des legos et des kaplas. J'ai commencé à créer des petits jeux et des maisons mais je voulais aller plus loin. J'ai découvert la robotique dans un atelier proposé au lycée français de Tokyo. L'objectif était de construire des robots de "combat" en lego tout en essayant de créer un algorithme imbattable. J'y ai pris goût et je me suis perdu dans cet univers. Puis je me suis retrouvé aux Philippines. Pendant ces trois années, je me suis mis seul au HTML tout en continuant les jeux vidéo, ma vraie passion. J'ai eu la chance de passer une journée de découverte dans le studio Ubisoft Philippines et c'est à ce moment là que je me suis rendu compte que c'était vraiment ce que je voulais faire. Pour accomplir mes rêves, Epita est l'école idéale. J'essaie de ne pas perdre pied malgré le confinement ! J'apprends tellement de choses avec des gens que j'apprécie et j'adore travailler avec eux. Dès que je peux faire plus, je le fais avec plaisir et curiosité. Voilà pourquoi j'ai choisi de travailler avec ces 3 personnes.

1.3.3 Arthur BARBIER

Depuis mon plus jeune âge, j'ai toujours touché aux jeux vidéo, ou même aux ordinateurs, avec mon père qui est ingénieur. Dès lors de mes 11 ans, j'ai eu mon premier pc sur lequel j'ai pu commencer à découvrir les vrais jeux multijoueurs, mais aussi où j'ai pu découvrir ce qu'est l'informatique. Ainsi, j'ai commencé à aimer regarder, m'y intéresser et j'ai tout de suite su ce que je voulais faire. Ainsi, lorsque je suis rentré en Terminale S, j'ai pris option ISN pour vraiment apprendre à coder. J'ai donc fait plusieurs jeux en python, un puissance 4, mais aussi un jeu du mélange mot (il faut retrouver un mot avec ses lettres mélangées). J'ai aussi eu la chance de découvrir ce qu'est l'HTML et le CSS et j'ai pu faire un site. Toujours à côté de tout cela, j'ai toujours joué aux jeux vidéo par passion, je touche à énormément de types de jeux vidéos, notamment des jeux en multijoueur, que ce soit de la coopération ou du 5 contre 5 car ça me permet de me détendre. Lorsque j'ai su que je rentrais à l'Epita, j'ai pu apprendre beaucoup plus sérieusement le monde de la programmation, et j'ai tout de suite apprécié. J'ai bien aimé le Ocaml et encore plus le C#.

1.3.4 Ronan PEDRON

Pendant le confinement j'ai ressorti un vieil ordinateur portable que j'avais. J'ai commencé à toucher un peu à tout, et j'ai de plus en plus apprécié le monde des pc, par conséquent j'ai abandonné ma Xbox qui me suivait depuis plusieurs années déjà. Après le "bac", j'ai monté mon propre pc, mon propre setup qui ne cesse de s'améliorer et dont je suis très fier et très chanceux d'avoir. Pendant le confinement et ainsi que



pendant une partie des grandes vacances, j'ai appris énormément sur comment je pouvais l'utiliser pour apprendre des choses autre que jouer à des jeux vidéo. J'ai commencé à toucher un peu à tout, à démonter mon pc portable, et regarder tout ce que je pouvais faire avec lui à mon échelle (maintenant je démonte plus mon pc pour le plaisir juste pour le nettoyer). Puis EPITA est rentré dans ma vie et j'ai découvert le monde incroyable qu'est la programmation qui m'était alors inconnu. J'ai appris tellement de choses en si peu de temps, je passe de quelqu'un qui ne savait même pas ce qu'était une condition à une personne qui est dans une équipe qui a fait un jeu. Apprendre avec des personnes qui sont passionnés tels que les ACDC et apprendre de mes camarades est la meilleure chose que je pouvais espérer en rentrant dans le supérieur.

1.4 Origine et nature de notre projet

Tout d'abord, nous étions tous d'accord pour développer ensemble un jeu vidéo. En effet, ce format nous a laissé une infinité de projets abordables même à notre niveau de S2. Nous sommes aussi quatre fans de ce genre de distraction/sport, ce qui a pour conséquence de nous stimuler davantage pour ce type de projet. Pour commencer, nous avons regardé les projets de nos aînées épitéens pour avoir une idée du niveau, ainsi que de la discipline à adopter pour cette épreuve. Un projet a attiré notre attention, le concept est simple, un joueur ayant le rôle de chasser tous les autres participants. Ils pourront se cacher parmi des personnages non joueurs qui auront eux aussi une forme humanoïde (durant cette présentation, différents termes seront utilisés pour désigner les personnages non joueurs tel que, PNJ, bot, intelligence artificielle et IA). Rien que la base de ce jeu est extrêmement riche en possibilités. Il fallait créer un terrain avec beaucoup de modèles 3D, façonnez des personnages à l'aspect humain et implémenter de simples IA qui pouvait évoluer via du « machine learning » par exemple. Ainsi, ce jeu se pratique en multijoueur et est aussi appelé « guess who ». Nous n'avons donc pas inventé le concept de notre projet mais, cela nous laissera plus de temps sur la partie intéressante et technique, passer d'une idée incomplète à un projet abouti.



2 Répartition des tâches

Nous avons essayé d'équilibrer au maximum les tâches en prenant en compte les points forts de chacun pour optimiser notre productivité. Cette répartition n'a que très légèrement changé depuis l'écriture de notre cahier des charges.

Ce n'est pas parce que nous sommes responsables d'une partie que nous ne nous sommes pas impliqués dans les autres. Nous y avons consacré la majorité de notre temps mais nous avons aussi décidé de travailler en paire pour rendre le jeu le plus qualitatif possible.

TABLE 1 – Répartition des tâches

Membre Tâche \	Sacha	Matthieu	Ronan	Arthur
Graphisme		+	+	
Réseau	+			+
Site web			+	
Intelligence Artificielle	+	+		
Son & Musique		+	+	
Gameplay	+			+
Animation		+	+	
Environnement du joueur		+		+
Manuel d'utilisation			+	+
Mémoire/Stockage	+		+	

3 GamePlay

La partie "gameplay" regroupe toute la structure de notre jeu : les rôles, les objectifs et les déplacements des joueurs, les interactions possibles entre chaque partie de l'environnement et comment une partie se termine.

3.1 Chat et la souris

Comme supposé plus tôt, notre jeu s'inspire "du chat et de la souris" et du "cache-cache" où un joueur nommé "Chasseur" traquera tous les autres participants de la partie nommés "Chassés", ils se dissimuleront grâce à des PNJs qui auront le même design, un aspect humain. Ce concept a permis deux choses, d'une part, les joueurs devront sans cesse adapter leur comportement avec les intelligences artificielles et d'un autre côté, les coder a été un grand défi pour nous, leur programme devait simuler les mouvements complexes et imprévisibles d'une personne.

3.2 Déplacement

Ainsi, la priorité a été d'implémenter tous les déplacements relatifs aux joueurs. Avancer, reculer, se mouvoir sur les côtés, sprinter, sauter et se tourner étaient des actions incontournable pour notre jeu. Par ailleurs, nous avons développé des mouvements supplémentaires tel que s'accroupir et s'assoir, ceux-ci ne sont pas nécessaires mais ajoute des possibilités aux joueurs. D'autre part, coder les déplacements en premier lieu nous permettait de circuler librement, nous pouvions observer l'environnement sous tous ces angles et ainsi améliorer la productivité de l'équipe chargée de la conception des modèles 3D.

3.3 "Attraper"

Le "chasseur" devra traquer les "chassées" mais, comment indiquera-t-il qu'il les a reconnu, trouvé, attrapé ? Nous avons choisi de lui donner des armes blanches et à feu. Il attaquaera les personnages qu'il soupçonnera, s'il s'avère être un fin détective, le chassé "trouvé" perdra des points de vie, dans le cas contraire, s'étant trompé de cible, le chasseur recevra un malus en perdant lui-même des points de vie. Lorsqu'un personnage meurt (ses points de vie sont inférieurs ou égaux à zéro), il disparaîtra, si c'est un joueur, celui-ci réapparaîtra en tant que spectateur. Un spectateur est une entité suivant la vision de tous les joueurs encore vivant. Une partie se termine quand l'un des deux camps a perdu tous ses soldats. Le chasseur gagnera lorsque tous les chassés auront disparu et vice-versa.

4 Réseau

Cette section regroupe tout ce qui rend possible le jeu à plusieurs. La première étape était de rejoindre différents utilisateurs dans la même partie.

Ensuite, nous nous sommes acharnés à minimiser cette partie en favorisant l'initiative locale. Cependant, certaines actions devaient être obligatoirement coordonnées entre les ordinateurs des différents participants, tel que l'apparition des personnages, leur rôle et d'autres attributs relatifs au jeu en lui-même.

Pour ce genre de partage d'informations, nous avons usé d'un système de "Hash". Ce procédé consiste à envoyer un message possédant une étiquette (une chaîne de caractères) pour le distinguer des autres messages, d'une 'target' pour savoir quel participant est ciblé par ce communiqué et enfin, d'une valeur tel qu'un entier ou une chaîne de caractères, c'est le message en lui-même.

4.1 PUN

Tout d'abord, pour se connecter à une même partie, nous avons décidé de choisir la librairie "PUN" (Photon Unity Network), qui nous permet d'héberger notre partie multijoueur sur les serveurs de Photon. Nous avons usé de la version gratuite qui permet de gérer jusqu'à 20 joueurs simultanément. Par ailleurs, lors du lancement du jeu, l'utilisateur se connecte automatiquement aux serveurs de Photon.

4.2 Lobby/Hall

Lorsqu'un joueur rejoint les serveurs, il a plusieurs possibilités, soit de trouver une salle ("room") déjà existante, créée par un autre joueur, soit d'en créer une, qui sera visible par tous les autres utilisateurs présents sur le serveur. Si elles s'accumulent, elles seront nommées et listées par ordre d'apparition. La personne qui aura créé la salle aura pour titre le "MasterClient".

4.3 Room/Salle

Lorsque le joueur a finalement rejoint une salle, il a la possibilité de voir tous ceux déjà présents dans celle-ci, avec le pseudo qu'ils auront mis au préalable. Seul le joueur ayant créé la salle aura la possibilité de lancer la partie. D'autre part, cette "salle" se déroule dans un bar que nous avons modélisé et qui fera office de salle d'attente. Attention, ce n'est pas la scène où les joueurs s'amuseront au chat et à la souris.

4.4 Personnage

Après s'être tous connecté dans la même partie, il reste à synchroniser le mouvement des personnages et des PNJs. Pour ce faire, il suffit d'utiliser des fonctionnalités



prédéfinies de photon qui est "Photon View" et qui confère à chaque joueur son propre point de vue. Ainsi, chaque participant contrôle son propre personnage et partage ses déplacements aux autres grâce à "Photon transform View". Par ailleurs, les animations sont synchronisés grâce à la fonctionnalité "Photon Animator View".

4.5 Intelligence artificielle

Concernant la gestion des IAs, assigner leur contrôle était le premier objectif. Lors de la première soutenance, nous avions décidé que seul le "MasterClient" maîtrisait les PNJs. Seulement, leur grand nombre provoquait un ralentissement de l'ordinateur en question. La meilleure solution était que tous les participants aient un nombre équivalent de bots à gérer. Cela répartissait la charge de traitement et a grandement fluidifié la qualité de jeu.

4.6 Autre

Enfin, beaucoup d'attributs relatifs au jeu en lui-même devait être partagés, cette sous-section énumère tous les ajustements nécessaires à une partie.

Tout d'abord, il fut impératif de dissocier l'apparition de chaque humanoïde pour éviter toute collision qui aurait pour conséquence de propulser un joueur à 150 mètres de la carte. Ainsi, chaque ordinateur ne pouvait choisir l'emplacement de son avatar sans le communiquer aux autres. L'alternative la plus simple était que seul le créateur de la partie (le MasterClient) décide de la position de tous les joueurs.

Ensuite, les rôle : comme nous souhaitons établir le nombre exact de « Chasseur » et de « Chassé », le MasterClient devait organiser les attributions au lieu qu'elles soient faites aléatoirement au risque de bouleverser l'équilibre du jeu.

Enfin, chaque ordinateur doit être capable d'actualiser la santé de tous les participants mais aussi des bots. En effet, il peut s'avérer, par exemple, qu'un chasseur fasse perdre des points de vie à un PNJ, celui qui contrôle ce bot devra en être informé pour pouvoir le faire disparaître lors de sa mort. Pareillement concernant les chasseurs.



5 Menu

Depuis la première soutenance, notre menu a beaucoup évolué sur différents aspects tels que le design ou encore les fonctionnalités.

5.1 Crédation ou connexion à un compte

Lors du lancement du jeu, l'utilisateur doit commencer par créer un compte stocké sur la base de donnée de Firebase si c'est sa première fois, sinon il peut juste se connecter avec son identifiant et son mot de passe afin d'arriver sur le menu principal.

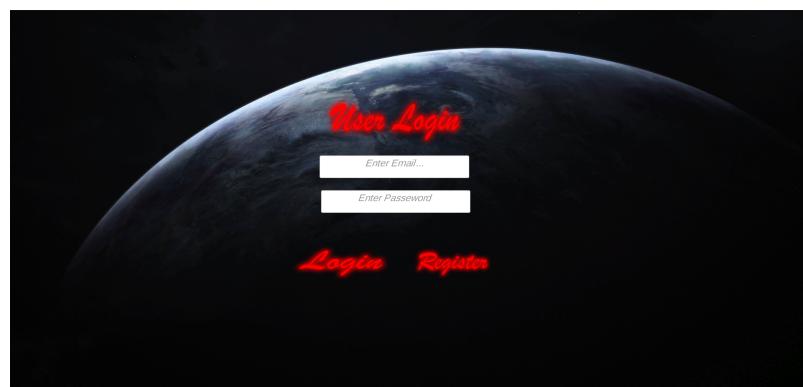


FIGURE 3 – Connexion à Firebase

5.2 Menu principal

De même, lors de sa première connexion au jeu, l'utilisateur doit rentrer un pseudo, il pourra à tout moment le changer dans le menu principal et celui-ci sera enregistré automatiquement pour éviter aux joueurs de devoir le rentrer à chaque lancement du jeu. Par la suite, l'utilisateur a plusieurs choix, soit il clique sur le bouton jouer où il peut rejoindre une salle déjà constituée ou en créer une, soit changer ses paramètres de jeu en cliquant sur le bouton "Options". Une salle permet donc de regrouper des joueurs afin qu'ils jouent dans la même partie en multijoueur.

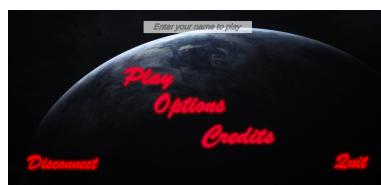


FIGURE 4 – Menu Principal



FIGURE 5 – Créeer ou rejoindre une salle

5.3 Options

L'utilisateur peut donc changer ses paramètres, tels que les graphismes, la résolution, le plein écran ou encore le volume du jeu.

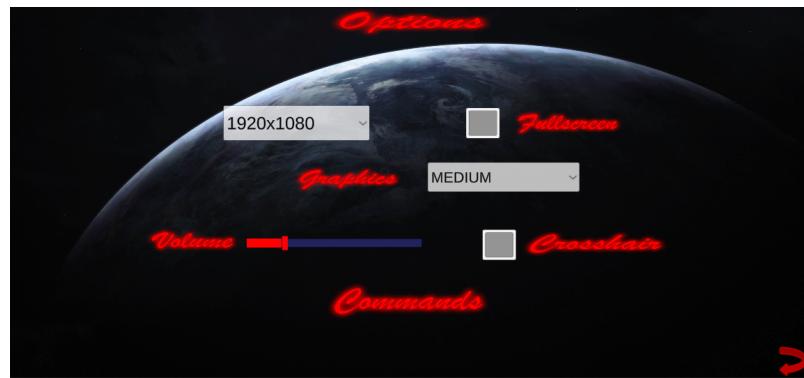


FIGURE 6 – Menu des options

Par ailleurs, il peut changer ses touches en appuyant sur le bouton "Commands". Par la suite, il peut juste appuyer sur la touche qu'il souhaite modifier et en presser une autre pour la remplacer.

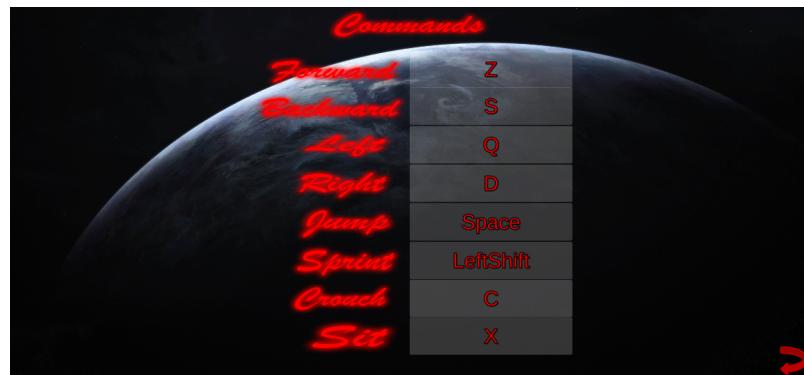


FIGURE 7 – Menu pour changer ses touches

6 Modélisation 3D

Afin de mener à bien la modélisation de tous mes modèles de notre jeu nous avons utilisé Blender. Blender est un logiciel de modélisation qui nous a permis non seulement de créer des modèles mais, également d'assembler la carte. Nous avons décidé de reproduire le campus d'EPITA à Villejuif, ceci n'a pas été mince à faire et a été une de nos principales préoccupation durant toute la durée de ce projet, nous sommes fière du résultat.

La création de tous les modèles qui constituent la carte s'agit d'un travail extrêmement long et minutieux. Nous avons fait d'énormes progrès sur Blender depuis nos débuts, non seulement dans la maîtrise de cet outil mais également dans la création de modèles.

6.1 Nos modèles

6.1.1 Les armes

Pour les armes, nous avons suivi des photos prises sur Internet. Ci-dessous vous retrouvez nos armes sous la forme suivante : la photo de gauche est notre résultat 3D et la photo de droite est notre inspiration.



FIGURE 8 – AK-47



FIGURE 9 – Clef à Molette

6.1.2 Les personnages

Vous pouvez également trouver l'évolution des différents personnages conçu tout au long du jeu.

Tout d'abord le tout premier qui n'a jamais réellement intégré le jeu :

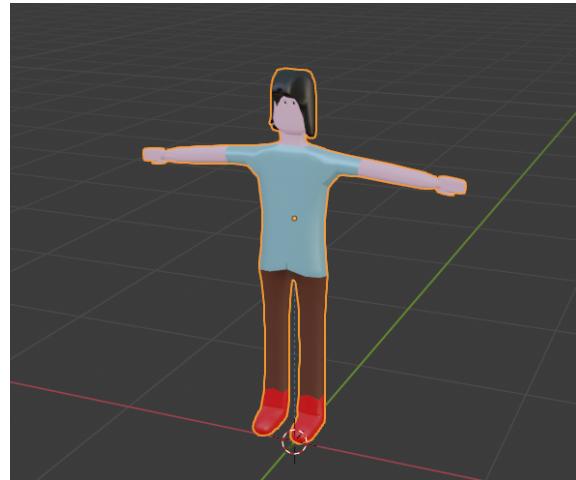


FIGURE 10 – Premier Personnage

Ensuite, nous avons eu les deuxièmes personnages qui nous ont suivi jusqu'à la fin de la deuxième soutenance :



FIGURE 11 – Le chasseur et le chassé

Pour la soutenance finale, nous avons décidé de recréer les personnages à partir de zéro afin qu'ils soient plus agréable à la vue :



FIGURE 12 – Chasseur



FIGURE 13 – Chassé n°1



FIGURE 14 – Chassé n°2



FIGURE 15 – Chassé n°3

6.2 La carte

La carte n'a pas eu de modifications significative entre la deuxième et troisième soutenance, nous avons rempli toutes les salles de classes de façon très légère afin de ne pas surcharger ces lieux. Cela permet aussi aux utilisateurs restreint en terme de puissances ou de stockage d'y jouer. A travers les photos suivantes, vous verrez l'évolution de notre carte au cours des soutenances. Cependant, entre la deuxième et la soutenance finale, il est exclusivement exposé le changement de l'intérieur d'EPITA.

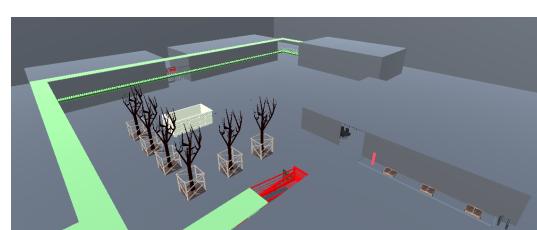
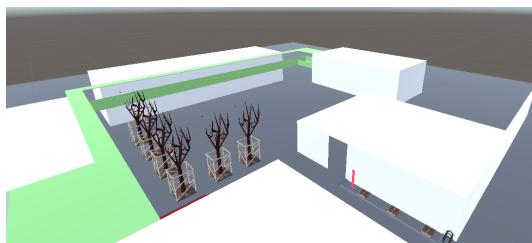
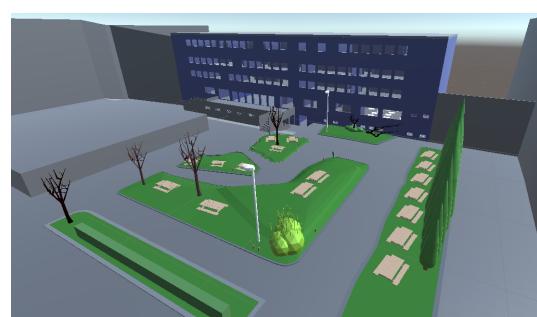
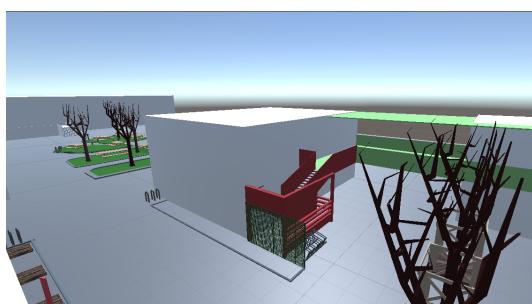
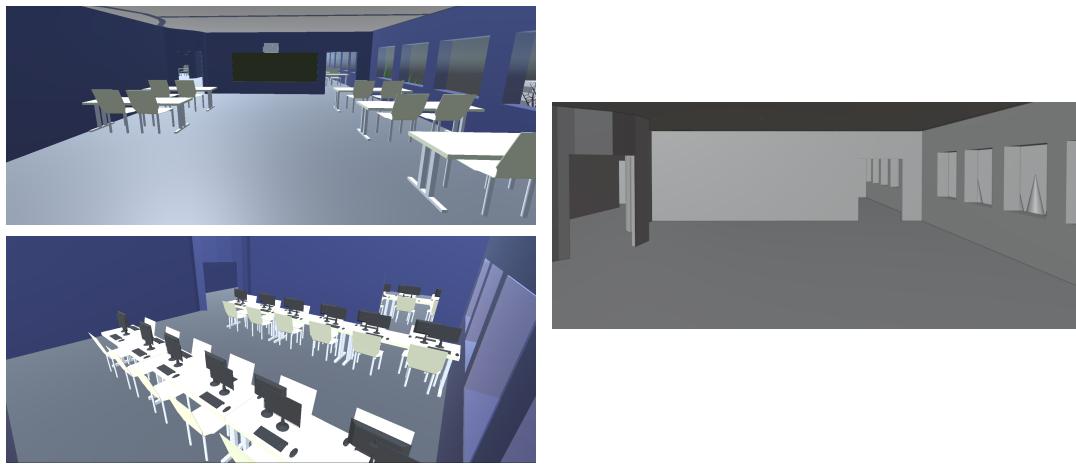


FIGURE 16 – première soutenance

FIGURE 17 – deuxième soutenance

**Actuel****2ème soutenance**

Au départ, nous souhaitions pouvoir déplacer le mobilier de l'école, après quelques essais nous nous sommes vite rendu compte que cela devenait injouable. En effet, placé dans un environnement fermé, cela pouvait bloquer l'accès de certaines classes.

6.3 Optimisation

Nous sommes arrivés sur un problème auquel nous ne pensions pas avoir à faire face, celle de l'optimisation. En effet, créer ses propres modèles n'est pas forcément la chose la plus optimisée que l'on puisse avoir en terme de point sur la carte. Plus il y a de point, plus l'objet est détaillé mais plus l'objet est lourd. Nous devons donc faire attention au nombre de points que nous avons sur la carte. Nous apprenons également que faire un arrondi avec 10 rotations de face ou 42 ne change pas beaucoup quant à l'apparence de notre modèle. Il faut donc privilégier les modèles qui sont esthétiques mais qui ont également un nombre de points raisonnable. Par exemple, ces deux modèles sont quasiment identique mais une des chaises possède moins de la moitié des points de l'autre.

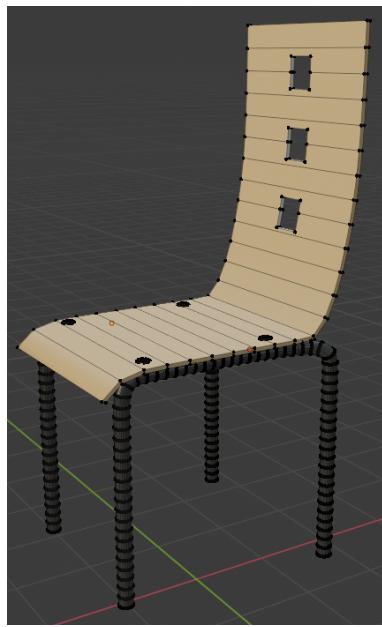


FIGURE 18 – Chaise non optimisé

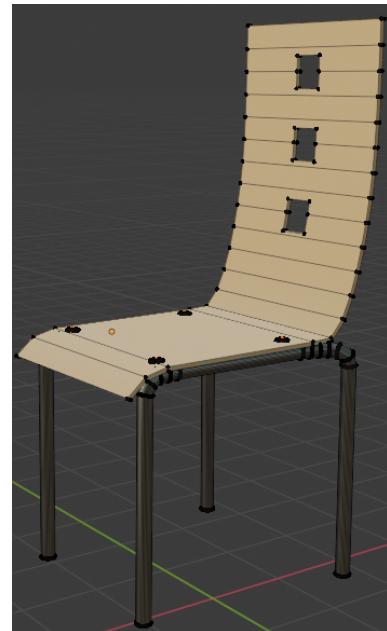
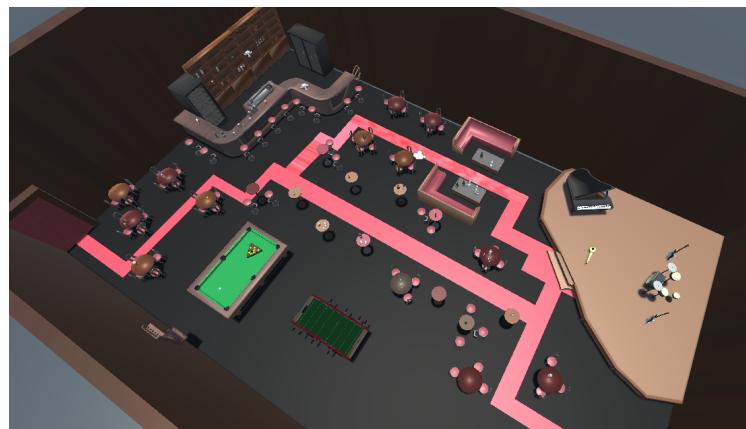
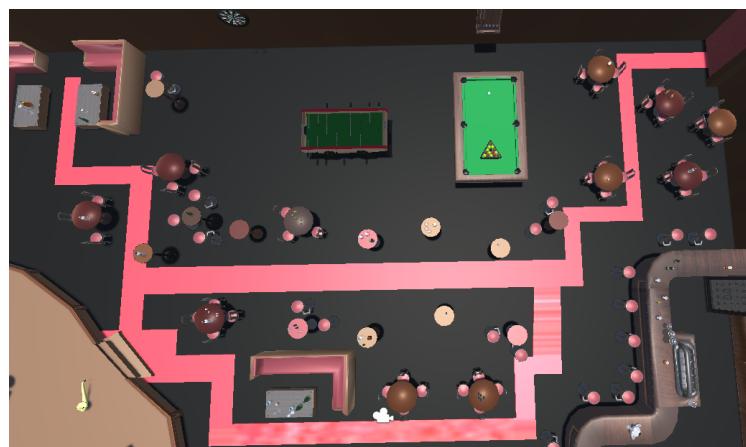


FIGURE 19 – Chaise optimisé

6.4 La salle d'attente : le bar

Comme détaillé plus tôt, il nous fallait une zone où le joueur puisse attendre le début de la partie et réglé certains aspect du jeu. Nous avons donc conçu un bar qui fait office de salle d'attente. Chaque élément de ce dernier fut créé de notre main.





7 Salle d'attente

Lorsqu'un joueur crée ou rejoint une partie, il apparaît dans une salle d'attente. Ainsi, cela évitera aux joueurs de rester devant un écran vide pendant de longues minutes. Ils pourront tout de même se déplacer avec leur personnage.



FIGURE 20 – Notre salle d'attente

Par ailleurs, seul le Master Client pourra lancer la partie. Il pourra aussi modifier les paramètres de celle-ci. Pour cela, il est nécessaire d'appuyer sur la touche Echap, le Master Client sera le seul à voir les boutons "Game Settings" et "Start Game".



FIGURE 21 – Le menu Echap de la salle d'attente

Ainsi, le créateur de la partie peut modifier le nombre de chasseur, le temps qu'ils auront pour trouver les chassés et la section de la carte où se déroulera la partie. Une plus grande liberté de jeu est donnée aux utilisateurs.



FIGURE 22 – Changer les paramètres de la partie



FIGURE 23 – Changer la carte

8 Animation

Comme pour tout dans notre jeu, l'animation a beaucoup évolué au fil des soutenances. C'est pour cela que nous allons vous expliquer durant cette partie comment nous avons travaillé au cours de tout notre projet.

8.1 L'animation des personnages

L'animation des personnages a subi un changement radical entre la première et la deuxième soutenance.

8.1.1 Première soutenance : la découverte et la prise en main

Nous avons commencé par découvrir le monde de l'animation sur Blender et Unity. Nous avons donc créé une armature adaptée pour tous les personnages afin que leurs animations soient identiques.

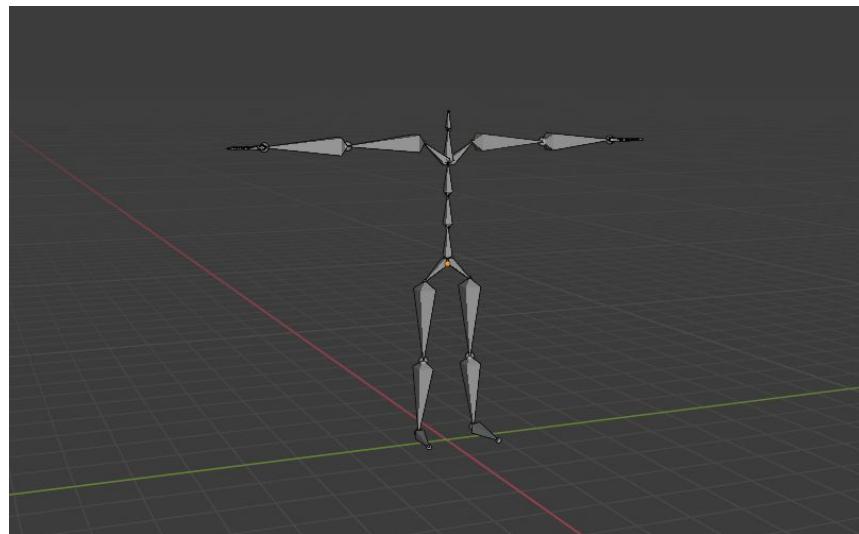


FIGURE 24 – Première Armature

Ensuite, nous avons dû animer les personnages. Comme nous ne connaissons pas



bien Blender et son système d'animation, nous avions fait au plus simple. Avec les animations de bases : marcher, courir, sauter, se mouvoir dans une direction (gauche ou droite), reculer et enfin s'accroupir. Toutes ces animations se trouvaient sur une seule lignée temporelle, une seule "timeline". Celle-ci était peu lisible (cf. l'image ci-dessous) car elle n'était qu'un amas de points. Pour s'y repérer, nous devions noter à part les indices de temps de début et de fin de chaque animation.

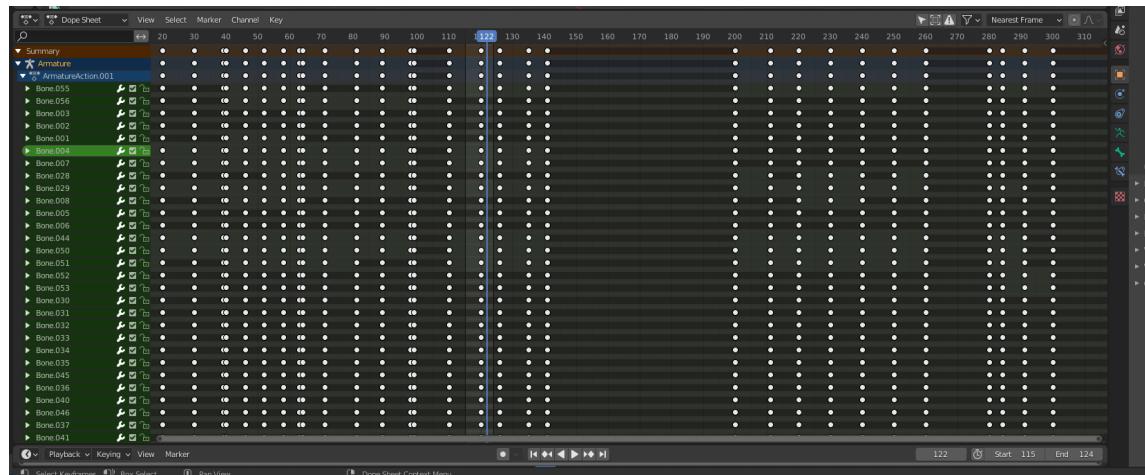


FIGURE 25 – La Timeline des animations

Pour la première soutenance, nous avions donc des animations fonctionnelles qui nous ont permis une certaine progression sur l'utilisation de Blender et Unity.

8.1.2 Deuxième Soutenance : le renouveau

Comme nous devions faire les animations spécifiques aux chasseurs (hunters) et que les animations que nous avions faites jusque-là ne nous plaisaient plus, nous avons donc décidé de repartir de zéro. Recréer les animations avec plus de connaissances sur Blender nous a permis de travailler plus vite et mieux.

Tout d'abord, nous avons façonné une nouvelle armature, plus adaptée à nos besoins, qui permet de travailler sur nos animations plus vite et d'avoir de meilleurs résultats. Cette armature était plus complexe et plus longue à concevoir mais elle nous a fait économiser énormément de temps lors de la conception des animations.

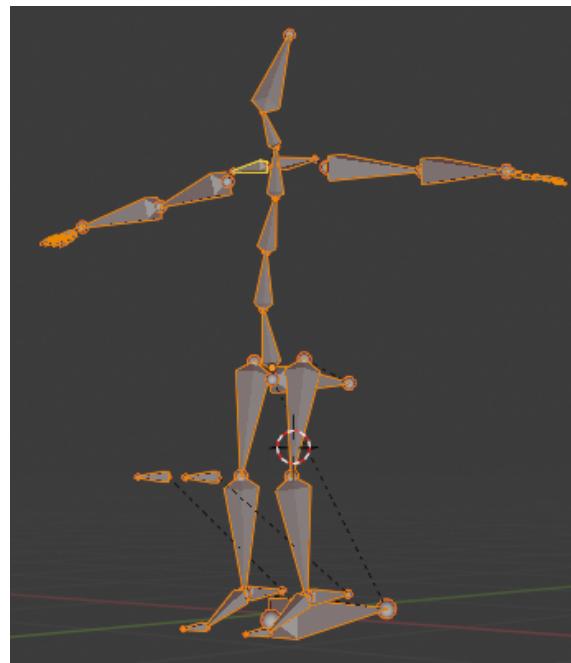


FIGURE 26 – Nouvelle Armature

Ensuite, comme pour la première soutenance, nous avons dû animer. Cette fois-ci nous l'avons fait de manière intelligente et ordonnée. Chaque animation a eu le droit à son dossier nommé. Nous n'avions plus à retenir les indices de temps.

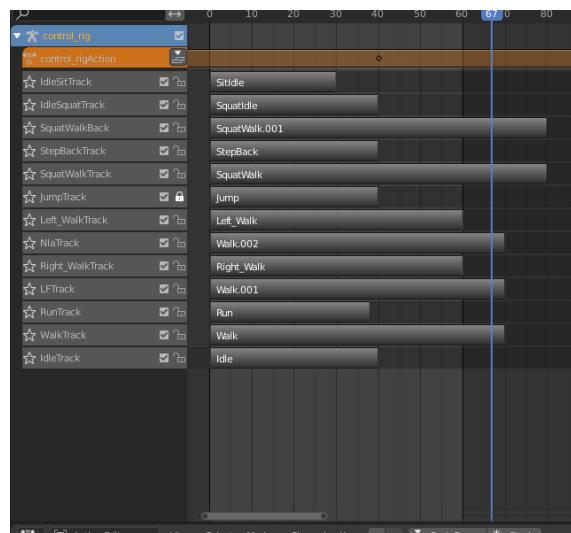


FIGURE 27 – Nouvel Arrangement

Ce nouvel arrangement nous a aussi permis de régler un problème que nous avions lors de l'importation sur Unity. En effet, le fait d'avoir toutes les animations sur une

seule timeline nous faisait perdre du temps à chaque importation sur Unity. En effet, nous devions les séparer manuellement en indiquant chaque indice de temps de début et de fin d'animation. Avec le nouvel arrangement, tout cela se faisait automatiquement.

La majorité des animations étaient communes aux chasseurs et aux chassés : marcher dans 6 direction (tout droit, à reculons, à droite, à gauche, en diagonale droite et en diagonale gauche), courir, sauter, s'accroupir et se déplacer tout en étant accroupit en avant et en arrière.

Le chassé (hunted) a la possibilité de s'asseoir. Les chasseurs ont la possibilité de frapper avec la clef à molette, ainsi que de viser et tirer avec l'arme à feu.

Pour les chasseur, l'armature est légèrement différente car elle permet d'accueillir les armes comme ci-dessous.



FIGURE 28 – Amarture du chassé et chassé avec la clef à molette

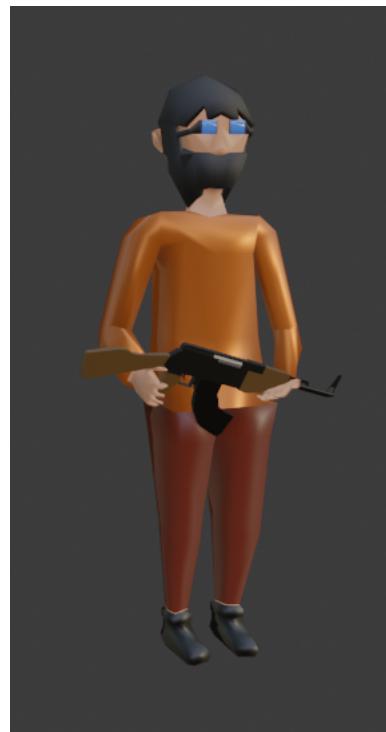
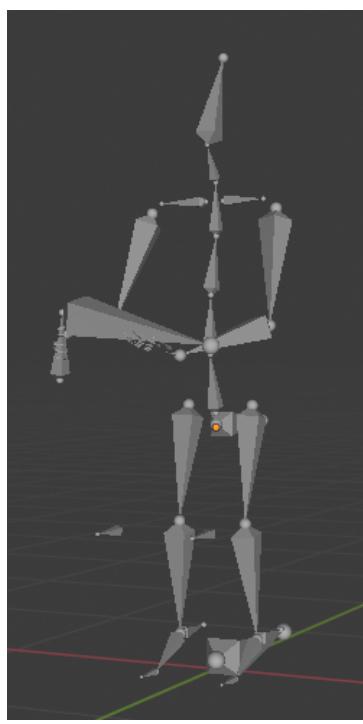


FIGURE 29 – Amarture du chassé et chassé avec la AK-47

Enfin, pour que toutes ces animations ne paraissent pas saccadées, nous avons dû utilisé l'Animator d'Unity ainsi qu'une cascade de booléen. Le schéma parle de lui-même (cf. ci-dessous), il est complexe mais reste simple. Chaque flèche est une transition possible d'une animation vers une autre.

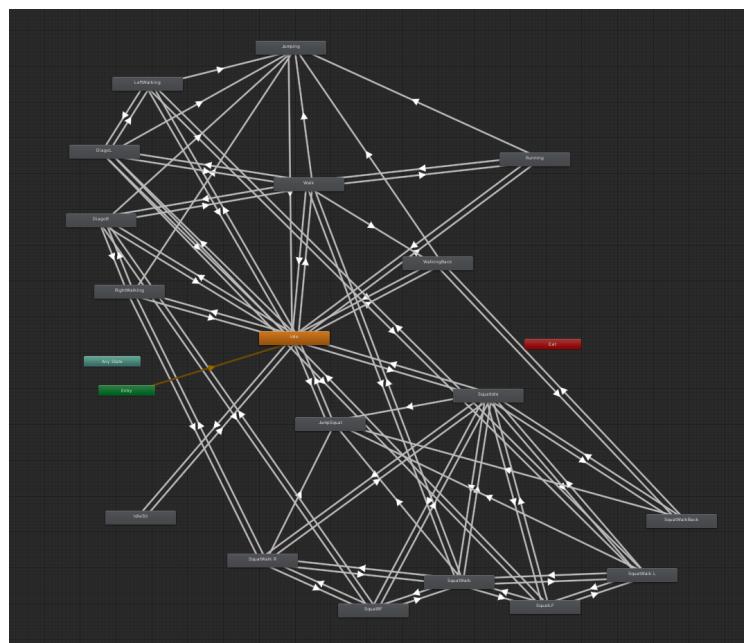


FIGURE 30 – Transition par l'animateur

8.1.3 Résultat finale

Le rendu sur nos personnages est devenu différent et beaucoup plus agréable à voir. Dans le même principe, nous sommes passé de l'image de gauche à l'image de droite.



FIGURE 31 – Ancienne carrière



FIGURE 32 – Nouvelle carrière

8.2 L'animation des menus

Pour rendre notre menu plus vivant nous avons décidé de mettre des animations d'entrée et d'interactions sur certains boutons.

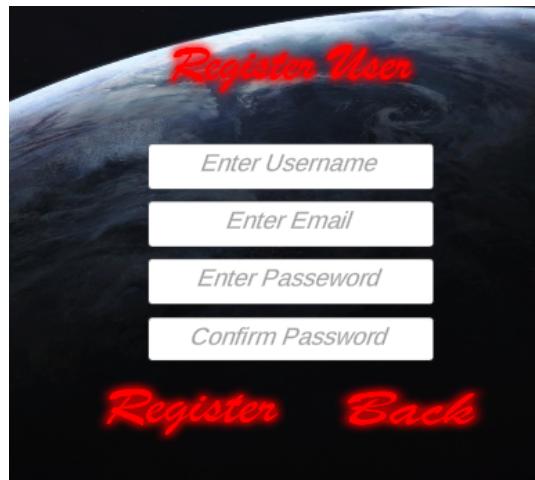


FIGURE 33 – Normal

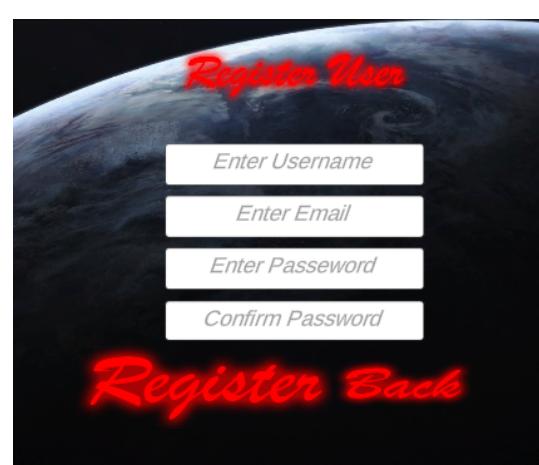


FIGURE 34 – Highlight

Lorsqu'on met la souris sur un bouton, celui ci s'avancera en premier plan. Vous pouvez remarquer que la taille du bouton "Register" diffère entre la figure 33 et 34.

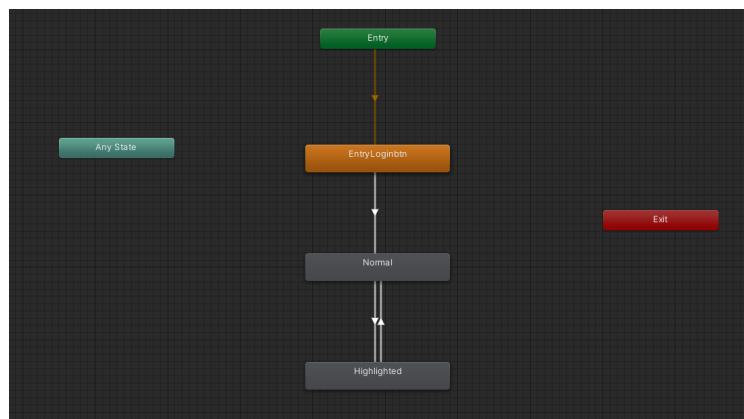


FIGURE 35 – Animator d'un bouton

Cet animator (réseau d'instructions) correspond à celui des boutons, qui ont une entrée (ils translatent), une position de base (quand on n'y touche pas), et une transition vers le "Highlight" quand nous passons notre souris dessus, comme dit précédemment.

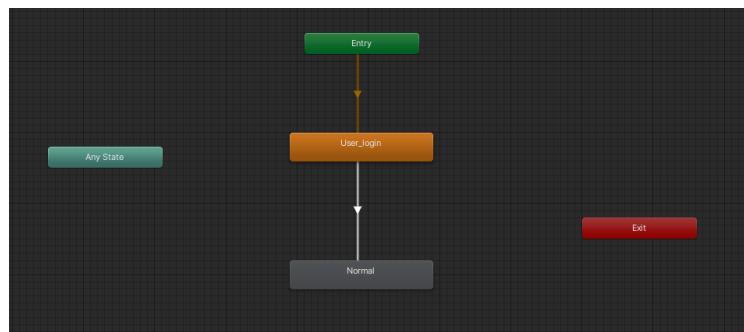


FIGURE 36 – Transition par l'animat

Cet animator correspond à celui des titres, qui est plus simple que l'animat des boutons étant donné qu'il n'y a pas de mode "highlight". Il n'y a qu'une animation d'entrée et une fois celle-ci faite celui-ci reste à sa place.

9 Intelligences Artificielles

Cette partie traitera des intelligences artificielles qui seront aussi appelés IA, bot, PNJs et personnages-non-joueurs. C'est l'une des parties incontournables de notre projet. Celle qui permet aux « chassés » de rester discret, autrement dit de subsister, mais aussi celle qui propose une infinité d'implémentation à nous, les programmeurs.

9.1 "Bot Rectiligne"

Tout d'abord, nous devions nous assurer que des personnages-non-joueurs se baladent à chaque recoin de notre environnement. S'ils ne le pouvaient pas, un "chassé" n'aurait pu se rendre à certains endroits au risque de se faire démasquer. Ainsi, des points ont été placés au quatre coins du campus et l'objectif de notre première intelligence artificielle était de naviguer entre ces coordonnées. Le "bot rectiligne" a ainsi obtenu un comportement basique. Lorsqu'il se trouve à un point, il en choisit aléatoirement un autre, s'oriente vers celui-ci, s'y déplace en ligne droite et recommence le processus. Nous considérons les déplacements de ce bot comme supervisés étant donné qu'il ne se dirigera jamais à un lieu que nous n'avions pas prévu. De plus, cette implémentation possède un autre avantage, les lieux de passages, autrement dit les espaces où les chassés auront le plus de chance de croiser les chasseurs, seront davantage parcourus par ces mêmes bots étant donné que davantage de points y seront placés.

En revanche, ce PNJ se déplace en ligne droite, nous devions nous assurer qu'il ne tente pas de voyager entre deux positions où leur traversée était obstruée. Par conséquent, nous devions vérifier l'existence d'obstacle infranchissable tel que les murs, les barrières, les fausses en les distinguant des escaliers, des buttes et des passages sans aucune entrave. Une vérification sommaire via des fonctionnalités prédéfinies de Unity ne permettaient pas de prendre en compte la vitesse et la force de saut des bots ainsi que la forme de nos modèles. Nous devions les considérer pour maximiser leur déplacement. Ce qui nous amène à la "maintenance".

9.1.1 "Maintenance"

L'objectif de cette sous-partie était de détecter si chaque point était voisin d'un autre. C'est à dire si un PNJ pouvait naviguer entre ces deux coordonnées sans tomber, rester bloqué ou se précipiter dans un mur. En conséquence, notre algorithme devait simuler le déplacement des vrais joueurs donc user de la même vitesse que ceux-ci, ce programme devait être lancé hors d'une partie.

Ainsi, nous est venu l'idée d'une « maintenance », le programme qui déterminerait les déplacements possibles des bots avant même que l'on diffuse notre jeu. Cela implique que les résultats soient stockés dans des fichiers et qu'un code puisse les lire pour en extraire les informations pertinentes, c'est le début de la mémoire de notre intelligence artificielle.



La maintenance, en pratique, est constituée de trois étapes. On choisit les zones de notre carte que l'on veut sonder. Chaque point présent dans la zone enverra une capsule qui aura, la masse, les dimensions et la vitesse d'un bot. Une seule capsule à la fois sera expédiée pour ne pas provoquer de potentielles collisions faussant les recherches. Enfin, lorsque chaque point a envoyé des capsules vers tous ces voisins, les résultats sont compactés et rangés dans un dossier regroupant la sauvegarde de toutes les zones du terrain. La manœuvre de cette maintenance peut paraître longue mais, sonder tout le campus (près de 400 points) dure seulement 6 minutes et atteint 95% de réussite dès la première tentative.

La figure 37 tente de montrer ce processus, les points sont affichés en bleu, les "capsules" ont l'aspect humain et les liaisons déjà trouvées sont désignées par les lignes rouges. D'autre part, vous pouvez observer une vidéo d'une "maintenance" sur notre site dans la section "Notre jeu" et "Notre IA".

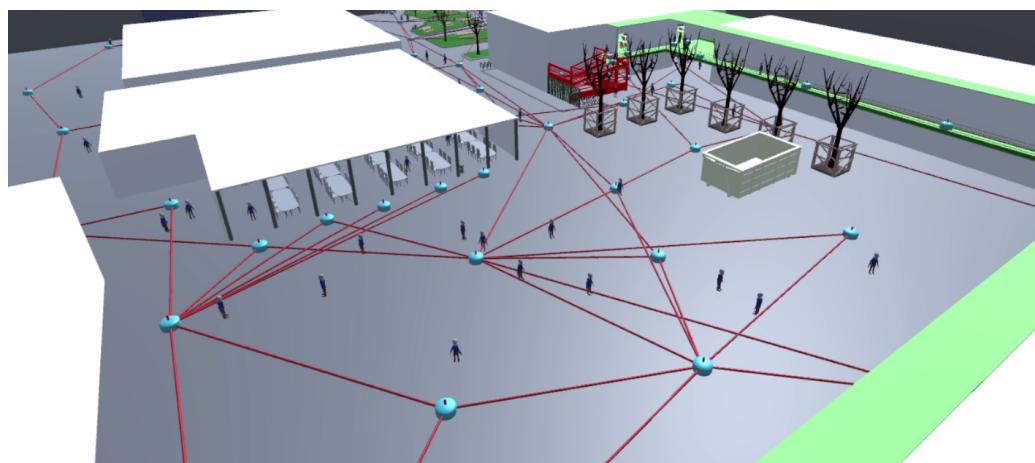


FIGURE 37 – Capture d'écran maintenance

La figure 38 représente le réseau obtenu grâce à la maintenance, chaque ligne rouge atteste qu'un bot pourra traverser en ligne droite d'une extrémité à une autre sans être bloqué.

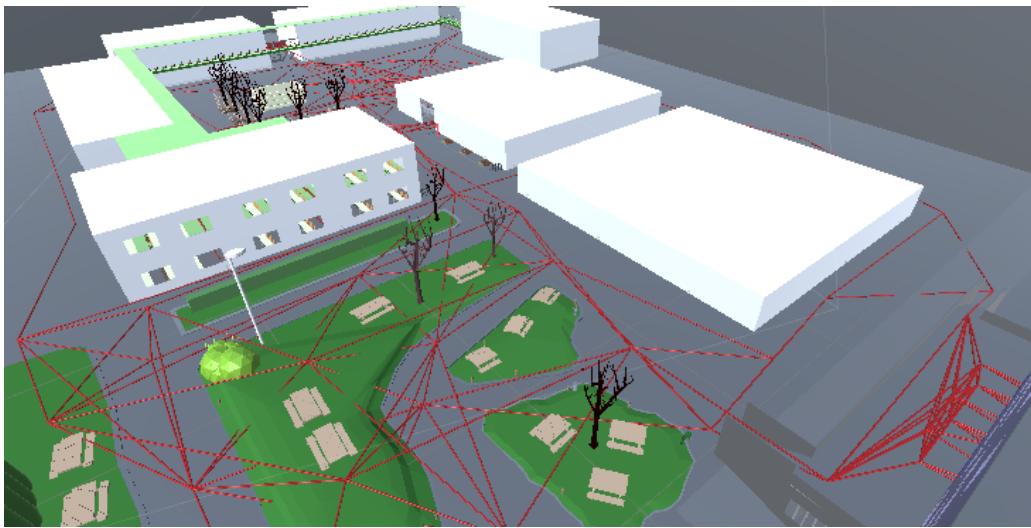


FIGURE 38 – Réseau de point

9.2 Mémoire/Sauvegarde

```

CrossPoint (0) : ,1,3,7,10,113,114
CrossPoint (1) : ,0,2,4,5,7,8,42,110
CrossPoint (2) : ,1,3,4,5,8,40,42,114
CrossPoint (3) : ,0,2,4,7,42,113,114,115
CrossPoint (4) : ,1,2,3,5,42
CrossPoint (5) : ,1,2,4,40,42
CrossPoint (6) : ,9,38,40,41,110
CrossPoint (7) : ,0,3,10,107,110
CrossPoint (8) : ,0,1,2,4,40,41,110
CrossPoint (9) : ,6,38,40,41,109
CrossPoint (10) : ,0,1,7,11,13,101,107,109
CrossPoint (12) : ,14,17,104
CrossPoint (13) : ,10,104
CrossPoint (14) : ,12,15
CrossPoint (15) : ,14,16,111
CrossPoint (38) : ,6,9,37,40,41,109

```

FIGURE 39 – Fichier de sauvegarde

La figure 39 représente la sauvegarde précédemment évoquée, elle se décompose en deux parties. Sur chaque ligne est inscrit le nom du point. Le nom n'est pas complètement libre, il doit contenir à son bout, un nombre entouré de parenthèses. Deux points ne pouvant avoir le même numéro, ils sont tous directement référencés par leur nombre. Il est donc très facile de les ranger dans une liste lors de la lecture de

tous les fichiers. Ensuite, est enregistré tous les voisins du point en question, ceux-ci sont désignés par leur nombre et séparés par une virgule.

9.3 Fuyard

Passons à la description de notre deuxième IA, le « Fuyard ». Celui-ci a directement été inspiré des joueurs pouvant perdre leur sang-froid à l'approche d'un chasseur. Ainsi, ce bot simule la vue d'un chassé afin de détecter un assaillant rentrant dans son « champs de vision ». Dans ces circonstances, le fuyard se déplacera vers un autre PNJs le plus loin possible de son assaillant en prenant le chemin le plus rapide. Il utilisera donc l'ensemble des "points" comme un graphique relié entre eux en fonction des obstacles qui les séparent. Cet algorithme est très peu coûteux puisqu'il réutilise la maintenance.

La figure 40 représente le graphique créé à partir des points et des informations engendrées par la maintenance. Le départ étant le cube marron et la destination le cube jaune. Chaque intersection désigne un point. Vous pouvez retrouver le processus complet sur notre site web.

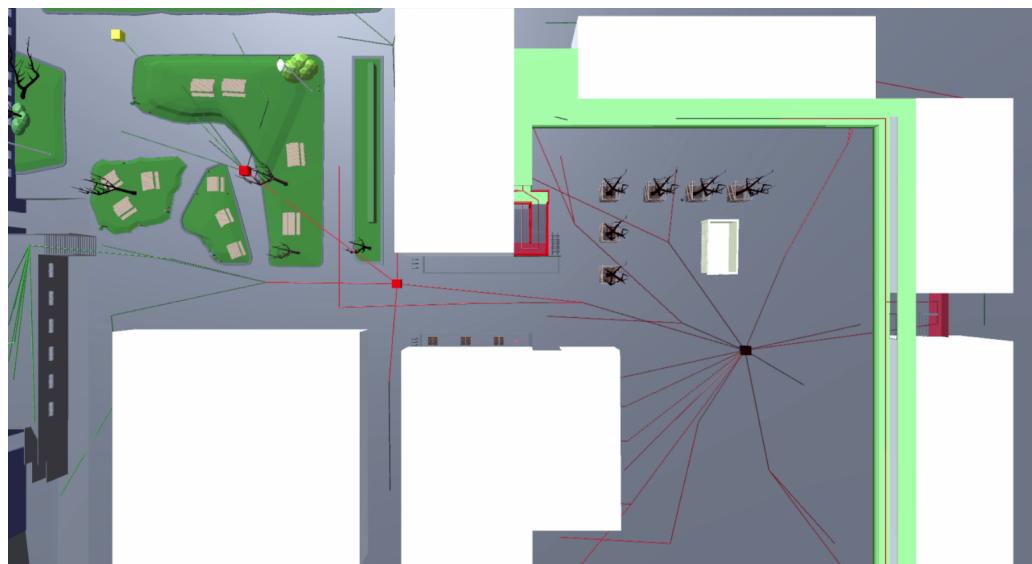


FIGURE 40 – Graphique

Cependant, ce choix d'implémentation présente un inconvénient. Nous ne plaçons pas des points à chaque coin, à chaque intersection, à chaque coordonnée de la carte. Ainsi, le chemin obtenu par ce graphique ne sera pas le plus rapide même s'il est le plus court à partir du réseau de points. Ce qui nous amène au « path finding ».

9.3.1 Path Finding

Le path finding est un algorithme censé trouvé le chemin le plus court entre deux coordonnées quelconque d'une zone. Premièrement, nous avons fait le choix de le coder sur deux dimensions étant donné la nature des déplacements des personnages. Ensuite, il devait être efficace, c'est à dire très peu coûteux en temps puisqu'un fuyard doit presque instantanément récupérer le résultat de la recherche. De plus, il est impossible de faire une sorte de « maintenance » pour sonder la zone avant la partie puisque cela reviendrait à trouver les meilleurs chemins entre ABSOLUMENT toutes les coordonnées de la carte.

Au final, l'algorithme prend une configuration similaire au parcours largeur d'un arbre général ou binaire. Tout d'abord, on part du point d'origine, le programme vérifie si sa position est valide, c'est-à-dire qu'elle n'est pas située sur un obstacle. Si ce n'est pas le cas, il s'arrête là, dans le cas contraire, il enfile quatre coordonnées (node) à une distance 'd' de lui et recommence ce travail avec le premier élément de la file. Deux résultats sont possibles, si la file se retrouve vide sans que la destination n'ait été atteinte, il existe manifestement aucun chemin entre les deux points rentrés en argument.

Dans le cas d'une recherche positive, le chemin sera construit en reliant les coordonnées qui se sont chaînées au fur et à mesure. En effet, chaque node retient celui qui l'a créé (enfilé). Cette implémentation fonctionne seulement parce que chaque node se trouve à équidistance de ces voisins. Ensuite, le chemin est raffiné en prenant un minimum de point à suivre. Cet algorithme est particulièrement intéressant dans un dédale désorientant. Vous pouvez d'ailleurs observer son utilisation sur notre site web grâce à un labyrinthe que le responsable d'édition 3D a pris soin de modéliser.

D'autre part, ce choix d'implémentation présente des défauts. Premièrement, un chemin sur deux étages différents ne saurait être trouvé étant donné que le programme opère sur deux dimensions (trois dimensions seraient beaucoup trop coûteux en temps). De plus, cet algorithme possède une complexité supérieur à celui utilisant le réseau de points, il prendra davantage de temps lors de sa recherche. Ainsi, ces deux programmes ayant des avantages et des inconvénients différents, ils sont donc utilisées par les "fuyards" à des situations précises.

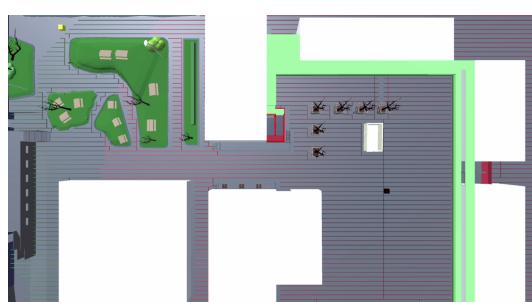


FIGURE 41 – Dispersion des nodes

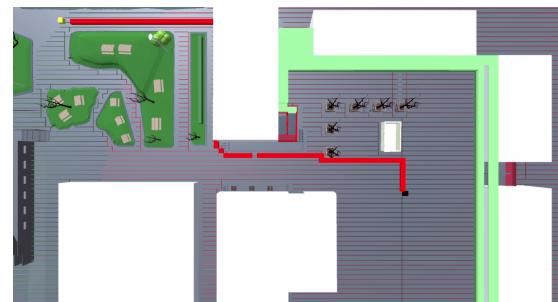


FIGURE 42 – Crédit du chemin



FIGURE 43 – Raffinement du parcours

9.4 Suiveur

Le « suiveur » est le dernier bot développé lors de ce projet. Cette intelligence artificielle a pour dessein de déboussoler le "chasseur". Son objectif principal est de maintenir en permanence une certaine distance avec le "chasseur". Ainsi, il s'en rapprochera lorsqu'il sera trop loin, s'en éloignera lorsqu'il sera trop proche et cessera de se déplacer dans le dernier cas. Grâce à cette attitude très particulière, les chassés pourront tenter de suivre le chasseur aux côtés des suiveurs sans être soupçonnés.

9.5 Machine Learning

Cette partie traite des avancées de la deuxième à la troisième soutenance. L'objectif n'a pas été d'implémenter de nouvelles natures de déplacements mais d'ajouter de nouvelles fonctionnalités aux intelligences artificielles déjà existantes. Comme par exemple améliorer leurs capteurs ou leur apprendre à sauter. Mais pour cet exercice, impossible de déterminer nous-même le moment précis lorsqu'un bot doit enclencher son saut, cela regroupe trop de paramètres à gérer. Il y a la distance entre l'obstacle et le bot, la hauteur de l'obstacle, la force de saut et la vitesse du bot, la forme et la taille du collider du bot (c'est ce qui crée toutes les collisions avec son environnement). En définitive, interpréter toutes ces informations nous étaient impossible. Nous avons donc opté pour une implémentation non supervisée qui comporte des réseaux neurones.

9.5.1 Réseau neurones

Un réseau par couche (layer) semblait le plus abordable, en voici un exemple.

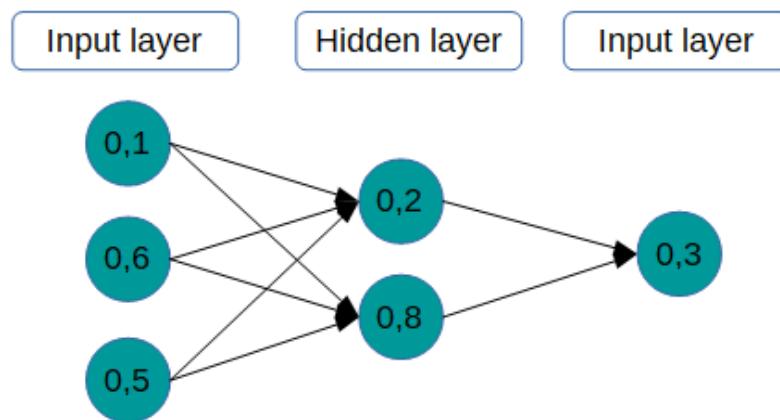


FIGURE 44 – Exemple réseau neurones

Input layer : représente la première couche, c'est ici que nous fournissons nos paramètres d'entrée.

Hidden(s) layer(s) : représente la couche cachée, elle permet de simuler des fonctions plus complexes. Cependant, nous n'en avons pas usé durant ce projet.

Output layer : représente le résultat attendu, ce sera, par exemple, si le bot doit sauter ou non.

Ce type de réseau, ne l'ayant pas inventé, nous n'y attarderons pas davantage mais vous pouvez trouver des précisions concernant ce réseau dans la vidéo de "3Blue1Brown" nommée "But what is a neural network ?" (<https://www.youtube.com/watch?v=aircAr>)

9.5.2 Entrainement

Cependant, ce "cerveau" devait s'améliorer pour finir par activer ces neurones au moment opportun. Nous avons décidé d'user d'un apprentissage par renforcement s'inspirant de la théorie de Charles Darwin : la théorie de l'évolution. Celui que vous allez faire est une sous-catégorie de cette même sous-catégorie, les algorithmes génétiques.

Ce que dit Wikipedia sur ce principe : En biologie, la sélection naturelle est l'un des mécanismes d'évolution des espèces qui expliquent le succès reproductif entre des individus d'une même espèce et le succès des gênes présents dans une population. Elle est ainsi « un avantage ou un désavantage reproductif, procuré par la présence ou l'absence de variations génétiques propices ou défavorables, face à un environnement », le système évolutif de la nature étant un immense jeu d'essais et d'erreurs. C'est un des aspects majeurs de la biodiversité, sur la planète, comme au sein des écosystèmes et des populations.

Ainsi, des espèces avec des bons gênes survivent et vont créer une descendance

avec de meilleurs gênes qui survivront encore mieux et pourront donc à leur tour engendrer plus d'enfants avec de meilleurs gênes.

9.5.3 Sauter

Pour apprendre à sauter, nous avons créé un réseau neurones possédant, en entrée, les paramètres précédemment exposés et en sortie, l'information si un saut doit être enclenché. La zone où les intelligences artificielles devait survivre ressemble à un saut d'obstacle avec des haies de hauteur différentes au cours duquel l'entité doit modifier sa vitesse. Ces "entités" ont les mêmes caractéristiques qu'un bot (saut, vitesse et collider), avance seulement en ligne droite et "meurent" lorsqu'ils rentrent en collision avec un obstacle trop haut (vers les genoux). Les réseaux neurones des entités parcourant le plus de distance sont sauvegardés et pourront être réutilisés par d'autre programme.

La figure 45 représente le terrain d'entraînement précédemment décrit avec les "entités" à l'aspect humain. Par ailleurs, vous pouvez observer un entraînement de saut sur notre site dans la section "Notre jeu" et "Notre IA".

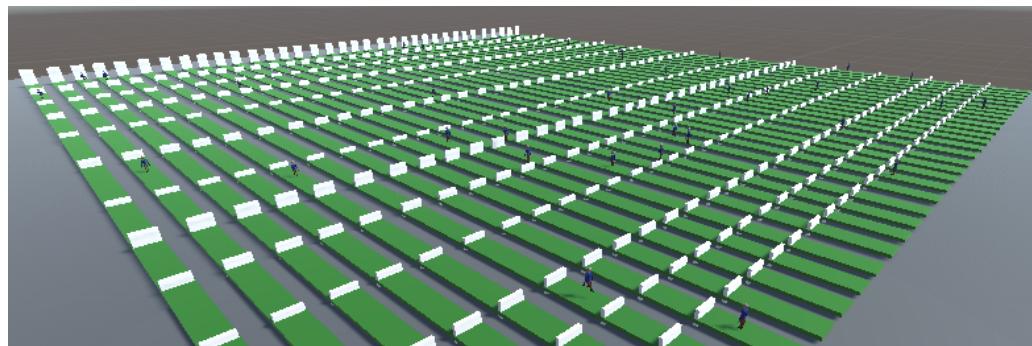


FIGURE 45 – Parcours saut

9.5.4 Déetecter

Le deuxième réseau neurones a été créé pour détecter si un obstacle était infranchissable ou abordable. Celui-ci est beaucoup plus simple, il prend une information en entrée, la hauteur de l'obstacle et une information en sortie.

Ce réseau ne pouvait être créé sans le précédent. En effet, comment savoir s'il est impossible de traverser une haie si nous ne savons pas sauter.

Ensuite, son terrain d'entraînement est similaire au saut de haies, les obstacles sont des murs qui diffèrent dans leur hauteur, cette fois-ci, certains seront volontairement trop imposants pour s'assurer que l'entité ne puisse les traverser en sautant.

Ainsi, l'entité qui possédera le réseau de neurones sautera à chaque obstacle et traversera un mur en supprimant temporairement la collision de celui-ci. L'entité "meurt" lorsqu'elle n'a pas supposé ce qui allait arriver, par exemple lorsque le réseau

prédit que l'entité pourra passer tandis qu'elle rencontre une entrave plus haut que ses genoux. D'autre part, un réseau suivant un schéma similaire a été créé pour détecter les escaliers et les dunes.

Une vidéo décrivant ce processus est sur notre site dans la section "Notre jeu" et "Notre IA". Lorsque l'entité suppose qu'il va rencontrer un obstacle, celui-ci se met en position assise, dans le cas contraire, il se met en position debout.

9.5.5 Amélioration

Après avoir créé, "entraîné" et sauvegardé ces réseaux neurones, nous pouvions enfin les intégrer à nos personnages-non-joueurs.

Premièrement, les "bots rectilignes" iront de point en point en sautant par dessus les obstacles trop haut. Pareillement pour les "fuyards" qui usent aussi du réseau de points. Enfin, les "suiveurs" ne profitant pas des informations obtenues par la "maintenance", ceux-ci ne savaient distinguer les murs des escaliers. Ainsi, ils ne se déplaçaient seulement sur un étage. Ces réseaux neurones ont donc pallier ce problème.

9.6 Ajustement

Chaque PNJ à un script précis à suivre mais, des imprévus sont toujours possibles. Ainsi, tous les bots vont réajuster leur direction toutes les certaines périodes. D'autre part, lorsqu'une IA reste bloqué un certain temps pour une raison inconnue, il se stoppera, se redirigera ou pivotera. Ces ajustements permettent de garder des PNJs opérationnels tout au long de la partie même si une situation imprévu les font dévier de leur droit chemin.

9.7 Conclusion

Trois bots aux attitudes différentes ont été implémentés. Cela empêche les joueurs de s'habituer aux déplacements caractéristiques qu'un PNJ pourrait posséder. Ils se feront, nous l'espérons, en permanence surprendre par leurs différences.

10 Base de donnée

Pour le projet nous avons décidé de rajouter un système de compte grâce à une base de données, Firebase.



Pour jouer, chaque joueur devra avoir un compte. Notre système de connexion est simple, il suffit d'une adresse mail et d'un mot de passe de plus de 6 caractères. Par exemple, nous pouvons voir sur l'image ci-dessous qu'il y a actuellement 4 comptes dans notre jeu.

Recherchez par adresse e-mail, numéro de téléphone ou ID utilisateur					Ajouter un utilisateur		
Identifiant	Fournisseurs	Date de création	Dernière connexion	UID utilisateur ↑			
arthur.barbier@epita.fr		28 avr. 2021		200Mdx7dfnRgo7HNPKObQMSyZ...			
sacha.hibon@epita.fr		28 avr. 2021		3gQ50RjD3SQSqJzVbgGQp24U4k...			
ronan.pedron@epita.fr		28 avr. 2021		6gBrtqOlcCSbH1p6C25y6Nrv1B13			
matthieu.camart@epita.fr		28 avr. 2021		Ilz7yTTcGWUyNcfkYJEPYKNbYc92			

Lignes par page : 50 ▾ 1 – 4 of 4 < >

FIGURE 46 – Comptes associés à notre jeu

10.1 Implémentation dans le jeu

Grâce à un menu login (pour se connecter) et register (pour se créer un compte) directement implémentés sur notre jeu, les utilisateurs peuvent se connecter et seront directement ajoutés à notre base de donnée.



11 Son

Notre premier son est une musique jazz que Ronan a joué avec des confrères avant la crise sanitaire. Cependant, nous espérions que la situation s'améliorerait pour en enregistrer de nouvelles. Ceci n'étant pas possible, nous avons décidé d'enregistrer des musiques que Ronan a joué avec un saxophone alto tout seul. D'autre part, le joueur peut modifier le volume de la musique dans ses options du menu pour un confort optimal.

12 Site web

Notre site internet a été créé à partir de langages web tels que l'html, le css ainsi que du javascript. Même si l'utilisation de framework tel que du Wordpress aurait pu améliorer la qualité de notre site, nous sommes très fiers du rendu final. Vous pouvez retrouver notre site web grâce à l'URL suivant : <https://smar.cf>

12.1 Le menu

Nous avons structuré notre site web en trois parties distinctes, l'accueil, nos membres et notre jeu. Dans ces sections il y a des sous-parties. "Nos membres" comporte 4 sous-sections correspondant aux quatre membres de notre projet "Ronan Pédrone", "Arthur Barbier", "Sacha Hibon", "Matthieu Camart". Pour terminer, "Notre jeu" a trois sous-sections, "Download", "Notre IA", "Nos modèles".

Accueil	Nos membres	Notre Jeu
	Ronan Pédrone	
	Arthur Barbier	
	Matthieu Camart	
	Sacha Hibon	

Accueil	Nos membres	Notre Jeu
		Download
		Notre IA
		Nos modèles

12.2 Accueil

Notre accueil permet de présenter brièvement notre équipe via une section "Qui sommes-nous ?".





D'autre part, nous avons implémenté un carrousel qui introduit tous nos membres grâce à une petite citation ainsi que tous les rôles qu'il a endossé durant ce projet.



Tous nos rapports de soutenance sont également disponibles en téléchargement.



Notre Instagram ainsi que nos mails peuvent être retrouvés à la fin de chaque page de notre site.

12.3 Nos membres

La section "Nos membres" de notre site permet d'en savoir un peu plus sur notre groupe. Une description de nos différentes visions de l'informatique ainsi qu'une section intitulée "Ce que le projet m'apportera" y sont affichées.



12.4 Notre jeu

12.4.1 Download

Dans cette partie du site, vous pouvez retrouver en téléchargement les différentes versions de notre jeu avec leur date de sortie. Un patch note leur sont associés pour savoir ce que nous avons amélioré, les erreurs que nous avons réglées ou les fonctionnalités que nous avons ajoutées.

12.4.2 Notre IA

L'équipe y met des explications, des vidéos et des images traitant de nos IAs ce qui n'est pas forcément possible sur les rapports écrits faute de place et de format.

12.4.3 Nos modèles

Nous y mettons tout ce qui est relatif aux modèles que nous faisons sur blender, ainsi que les bande-annonces que nous faisons pour notre jeu.

12.5 L'hébergement

Notre site internet est hébergé sur le serveur de Ronan.



13 Problèmes rencontrés et solutions

13.1 Collision

Comme pour tout jeu, gérer les collisions n'est pas une partie facile. Dans notre cas, si une personne s'entête à foncer dans un coin de notre carte, il peut arriver qu'il translate vers le haut où même qu'il le traverse (si les murs ne sont pas épais). Nous n'avons pas réussi à déceler l'origine de ce problème mais nous l'avons contourné et exploité.

Nous avons doté chaque personnage d'un cœur situé au milieu de l'abdomen, si un objet autre que le corps du joueur rentre en collision avec cet organe, celui-ci perdra des points de vie. Cela permet deux choses, les petits malins pourront profiter de raccourcis mais jamais en abuser.

13.2 Taille de la carte

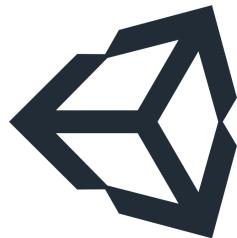
Comme évoqué plus tôt, notre environnement est le campus de Villejuif d'EPITA. Ayant fait ce choix, adapter la surface en fonction des besoins de notre jeu n'était pas une possibilité. Ce qui avait pour conséquence de déséquilibrer une partie. Les chassés avaient trop de cachettes et le chasseur très peu de chance de gagner.

Nous avons décidé de séparer la carte en sous section. Les joueurs auront la possibilité de choisir laquelle utiliser dans la salle d'attente, le bar.



14 Page d'outils

- Unity est un moteur de jeu multi-plateforme. C'est le logiciel qui nous a permis de créer Hunter x Hunted. Il regroupe toutes les ressources que nous avons utilisées (scripts, images, textures, models 3D...) afin de les mettre en relation et créer un jeu.



- Jetbrain Rider est un IDE (environnement de développement intégré). Il a été utilisé pour écrire tous nos scripts en C# pour Unity.



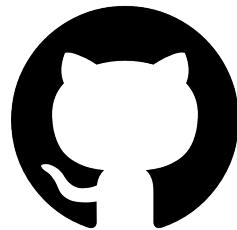
- Visual Studio Code est un IDE. Il a été utilisé pour écrire notre site Web.



- Git est un logiciel de gestion de versions. Il nous permet de travailler à plusieurs en même temps sur le projet en facilitant le partage de code. Il permet également de créer automatiquement des sauvegardes du projet.



- GitHub est un site web qui nous permet d'héberger en ligne les changements effectués avec Git.



- WinRAR est un logiciel de compression de données.



- Blender est un logiciel de modélisation, d'animation et de rendu en 3D.



- Inno Setup est un logiciel permettant de créer des installateurs pour Windows. Il nous a servi à créer l'installateur de notre jeu.





- Overleaf est un logiciel de traitement de texte basé sur le web. Ce logiciel nous a permis d'éditer à plusieurs en même temps des documents écrits en LATEX. Il nous a servi pour la rédaction des rapports.



- Première Pro est un logiciel de montage vidéo. Celui-ci nous a permis d'éditer toutes les vidéos que nous avons pu enregistrer pour le bien de notre projet.



- Photoshop est un logiciel de retouche, de traitement et de dessin assisté par ordinateur. Ce logiciel nous a aidé pour tous les aspects design de notre projet.



- Procreate est une application d'édition graphique et de peinture numérique, nous l'avons utilisé pour dessiner le logo de notre jeu.





- Firebase est un ensemble de services d'hébergement pour n'importe quel type d'application. Ce service nous a été utile pour la mise en place de nos comptes.



15 Conclusion

15.1 Ce que le projet nous a apporté

15.1.1 Matthieu CAMART

J'ai pris beaucoup de plaisir à faire ce projet avec ce groupe. Il m'a permis de me développer sur des domaines que je connaissais peu tel que la modélisation 3D et l'animation. J'ai tout de même beaucoup appris en code au travers des autres et de ce que j'ai fait. J'ai bien aimé le travail de groupe qui permet un mélange de plein d'idées différentes permettant de trouver des solutions qu'on ne trouverait pas seul. J'ai découvert le Monster dans les zones de rush qui a engendré beaucoup de fous rire. J'ai appris aussi qu'en plus de ne pas en boire il fallait ne pas en abuser. En bref, ce projet était vraiment une expérience agréable et j'espère que mes prochains projets vont m'être aussi bénéfique que celui-ci.

15.1.2 Arthur BARBIER

J'ai apprécié travailler sur ce projet pendant ces nombreux mois, il m'a permis de découvrir comment créer un jeu vidéo de A à Z. Par la même occasion, j'ai découvert la plateforme de développement qu'est Unity ce qui m'a permis de m'améliorer dans le langage C#. Enfin, cela m'a permis d'apprendre à travailler en groupe pendant une longue période et ainsi obtenir de l'aide lorsque je bloquais et vice-versa.



15.1.3 Ronan PEDRON

Ce projet m'a permis d'apprendre réellement ce qu'est le travail en groupe, de devoir savoir où en est tout le monde afin de pouvoir coordonner le travail de tout le monde. J'ai également appris qu'en étant "responsable" d'un aspect d'un projet qu'il était important de devoir savoir répondre à toutes les questions de ses camarades et qu'il y avait une grande différence entre comprendre soit même et comprendre afin de pouvoir l'expliquer. J'ai également appris beaucoup de chose quant à de nouvelles technologies que je connaissais pas tel que les bases de données ou les langages web. J'ai apprécié ce projet et cela ne présage que du bon quand aux prochains projets à venir.

15.1.4 Sacha HIBON

Ce travail était pour moi, passionnant. Le thème laissé libre nous a permis de choisir un projet qui nous correspondait et plaisait réellement. De plus, il était étalé sur une période assez longue où de nombreuses idées pouvaient encore émergées. Enfin, j'avais seulement entrepris des projets seuls, j'ai appris que le travail en équipe pouvait être formateur et gratifiant.

15.2 Un petit mot pour la fin

Pour conclure sur ce projet, nous sommes heureux d'avoir finalement pu concrétiser ce que nous avions en tête depuis novembre 2020. Bien que la modélisation 3D nous ait demandé beaucoup de temps, cela nous a permis de donner notre touche personnelle sur ce projet, que nous avons pu créer de A à Z. Nous sommes très satisfaits de tout le travail accompli et du rendu final, qui dépasse nos espérances du début de l'année.

16 Annexes

16.1 Documentation

- Unity :

<https://docs.unity3d.com/Manual/index.html>

- Photon :

<https://doc.photonengine.com/en-us/pun/current/getting-started/pun-intro>

- Latex :

<https://fr.overleaf.com/learn>

16.2 Autre

- Chaine Youtube de Brackeys :

<https://www.youtube.com/user/Brackeys>

- Chaine Youtube de Dapper Dinos :

<https://www.youtube.com/channel/UCjCpZyil4D8TBb5nVTMMaUw>

- Chaine Youtube de Rugbug Redfern :

<https://www.youtube.com/channel/UClawNSOpRYRqi6Twk3GUG-A>

- Chaine Youtube de Unity Pour les Nuls :

<https://www.youtube.com/channel/UCuU8cONIgZ182KheI1s6HqQ>