Rapport de projet

MS 402

Juin 2023

Erwan LE GRAND - Alexis LATOURNERIE
Thomas GRAVELINE-MERCIER - Nathan CHAMPAGNE

Table des matières

1 Introduction

1.1 L'équipe et le projet

A2NT est le nom de l'équipe de développement du projet dont voici les membres :
— Erwan LE GRAND (chef de groupe)
— Alexis LATOURNERIE
— Thomas GRAVELINE-MERCIER
— Nathan CHAMPAGNE

Nous avons choisi comme nom de groupe **A2NT**, correspondant aux initiales des pseudonymes Discord de chacun des membres du groupe.

Nous avons choisi de réaliser pour notre projet du deuxième semestre un jeu vidéo en 3D nommé *MS 402*. Pour ce faire, nous utiliserons le moteur de jeu Unity et le langage de programmation C# sur l'environnement de développement JetBrains Rider.

MS 402 est un jeu d'infiltration qui se joue à deux joueurs, en ligne et en coopération. Les deux joueurs n'ont pas les mêmes habiletés et doivent s'infiltrer de zone en zone sans se faire repérer afin de découvrir la vérité sur l'histoire de *MS 402*.

Figure 1 – Logo du groupe A2NT

2 Origine et nature du projet

2.1 Origine

Le style du projet a été décidé après plusieurs discussions, celui qui a retenu notre attention est le jeu d'infiltration. C'est le type qui nous permet d'exploiter au mieux nos idées et nos envies pour la création d'un jeu vidéo. Ainsi, notre inspiration provient de différents jeux d'infiltrations tels que *Hitman* ou *Watch Dogs*.

Notre histoire a ensuite été imaginée grâce à l'échange de nos idées. Le tout s'est révélé beaucoup plus clair lorsque nous avons trouvé ce que nos joueurs doivent aller récupérer pour terminer la partie, il s'agit d'un manuscrit dont le nom éponyme est celui de notre jeu.

Ce manuscrit a été imaginé en prenant pour base un manuscrit existant réellement, le *MS 408*. Dit aussi le Manuscrit de Voynich, il s'agit d'un livre datant approximativement de 1409-1438, qui contient des textes indéchiffrables, des plantes inconnues ou encore des schémas incompréhensibles. La quête de nos personnages s'est précisée et nous avons pu fusionner nos nouvelles idées liées au *MS 408* avec le scénario que nous avions initialement.

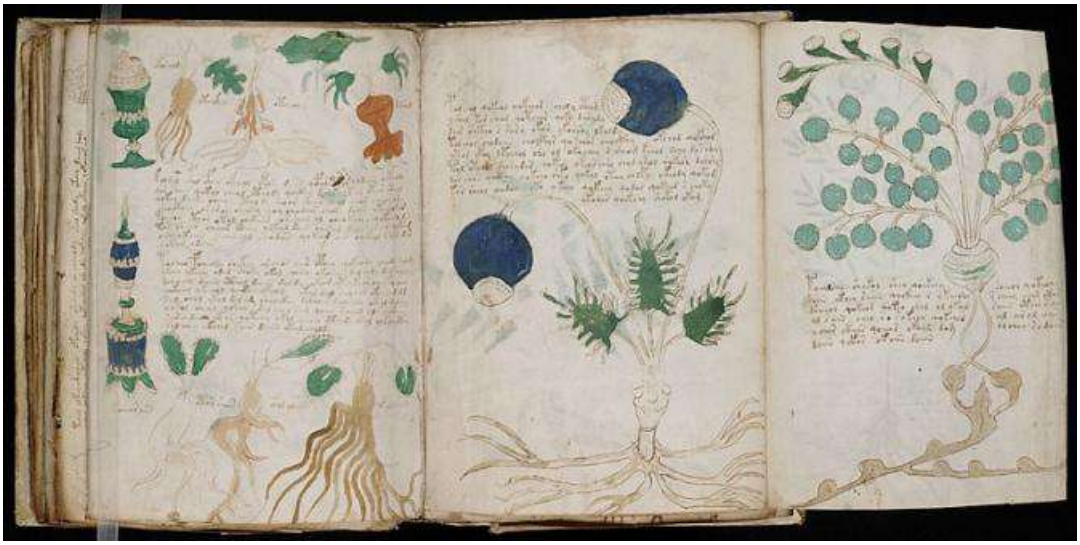

Figure 2 – Manuscrit de Voynich

2.2 Résumé de l'histoire

En 2035, deux amis journalistes, Tom Voy et Jack Nich, sont chargés d'enquêter sur un ancien manuscrit indéchiffrable appelé *MS 402*. Après deux années de recherches, ils apprennent que le gouvernement a saisi le manuscrit et leur reportage est annulé. Tom devient obsédé par le manuscrit et continue à chercher des réponses. Près de trois ans plus tard, Jack reçoit une lettre de Tom lui disant qu'il a découvert quelque chose d'important sur le *MS 402* et qu'il est poursuivi. C'est probablement leur dernier contact, car Tom est en danger et ne sait pas s'il va survivre. Jack et la sœur de Tom partent à la recherche de son ami et souhaitent éclaircir le mystère autour du manuscrit *MS 402*.

3 Répartitions des tâches

3.1 Multijoueur et réseau

MS 402 se joue en coopération en ligne. Il faut donc relier en réseau les deux joueurs afin que toutes les actions se déroulent en même temps pour les joueurs. Nous conseillerons aux joueurs d'utiliser des logiciels de communication vocale comme Discord ou TeamSpeak si nécessaire.

3.2 Scénario du jeu

Le scénario du jeu consiste à écrire une histoire cohérente pour le déroulé du jeu, mais aussi de déterminer tous les éléments de jeu tels que les missions, les objectifs à réaliser, les habilités spécifiques des deux joueurs, leurs outils, etc.

3.3 Création de la map

Cela consiste à trouver les assets et modéliser en trois dimensions la carte de notre jeu vidéo. Si besoin, des petits modèles 3D pourront être créés par nous grâce au logiciel Blender.

3.4 Site Web

Réaliser un site qui contient une page d'accueil permettant d'accéder à la présentation du projet, des liens dirigeant vers les sites des outils utilisés, nos différents rapports et documents, le téléchargement du projet et d'une version lite de celui-ci.

3.5 Menus et HUD

Création du menu principal à l'arrivée dans le jeu avec la possibilité de créer une partie et d'en rejoindre une. Création d'un menu pause accessible en cours de partie et permettant de quitter cette dernière et permettant également d'accéder aux paramètres ainsi qu'à l'histoire du jeu. Affichage de l'inventaire, affichage et déclenchement des dialogues, quêtes et avertissements en jeu.

3.6 Mécaniques du jeu

Cette partie consiste à mettre en œuvre les actions comme les mouvements et la gestion des caméras, mais aussi les différents éléments du scénario, développer les interactions entre la carte et les entités (joueurs, ennemis).

3.7 Intelligence artificielle

L'intelligence artificielle sera implémentée dans le comportement des ennemis. Ceux-ci détecteront les joueurs selon certaines conditions (comme la présence du joueur dans son champ de vision) et se déplaceront automatiquement en faisant des rondes. Ils seront aussi capables d'attaquer les joueurs afin d'essayer de les mettre hors d'état de nuire (dans le cas où un joueur est neutralisé, les deux joueurs ont perdu et sont téléportés au dernier point de sauvegarde).

3.8 Animations

Faire les animations des joueurs et des objets tels que l'ouverture des portes ainsi que les interactions possibles entre les éléments. Les animations doivent aussi être reliées au système de multijoueur.

3.9 Tableau de répartition des tâches

Voici le planning de la répartition des tâches par personne avec deux personnes par tâches : le responsable (Resp.) et son suppléant (Sup.).

	Erwan	**Thomas**	**Alexis**	**Nathan**
Création de la map		**Resp.**	**Sup.**	
Multijoueur et réseau			**Resp.**	**Sup.**
Scénario du jeu	**Sup.**	**Resp.**		
Mécaniques du jeu			**Resp.**	**Sup.**
Intelligence Artificielle	**Resp.**		**Sup.**	
Menus et HUD	**Resp.**	**Sup.**		
Site Web	**Sup.**			**Resp.**
Animations		**Sup.**		**Resp.**

4 Réalisation des tâches

4.1 Tableau d'avancement

Tâches	1ère soutenance	2ème soutenance	Soutenance finale
Multijoueur et réseau	100% / 100%	100% / 100%	100% / 100%
Scénario du jeu	90% / 85%	95% / 95%	100% / 100%
Création de la map	60% / 50%	80% / 85%	100% / 100%
Site Web	40% / 60%	60% / 90%	100% / 100%
Menus et HUD	30% / 40%	80% / 80%	100% / 100%
Mécaniques du jeu	20% / 25%	60% / 80%	100% / 100%
Intelligence Artificielle	15% / 15%	80% / 60%	100% / 100%
Animations	15% / 30%	60% / 75%	100% / 100%

4.2 Multijoueur et réseau

4.2.1 Confection du multijoueur

Tout d'abord, nous avons recherché les différentes solutions existantes permettant d'implémenter un multijoueur dans le moteur de jeu Unity. Notre choix s'est arrêté sur Mirror. Cette librairie correspond à notre projet, car celle-ci fonctionne en Peer-to-Peer (les données transitent uniquement entre les deux joueurs, elles ne passent pas par un serveur global). Ce qui est idéal dans notre situation puisque *MS 402* est un jeu de coopération, avec seulement deux joueurs qui ont décidé de jouer ensemble. Contrairement à d'autres jeux-vidéos, nous n'avons pas besoin de relier des joueurs qui ne se connaissent pas dans la création des parties.

Ensuite, nous avons mis en place la base du système de multijoueur Mirror. Dans la configuration la plus classique, l'un des deux joueurs est le "host". C'est-à-dire que c'est sur son ordinateur que sera hébergé le serveur en même temps qu'il sera

client puisqu'il jouera un personnage. Le deuxième joueur est lui uniquement un client et se connecte au premier en renseignant l'adresse IP de celui-ci ou alors en se connectant à un des serveurs, automatiquement détecté sur le réseau local. En outre, nous avons ajouté les modèles 3D des personnages et les points d'apparitions afin que les joueurs puissent jouer leur personnage. Cela nécessitait de désactiver certains éléments comme les caméras ou les fonctions permettant de contrôler les joueurs sur les éléments n'appartenant pas au joueur, afin que chaque joueur soit bien dissocié.

Nous avons aussi créé des fonctions publiques afin de pouvoir relier le menu principal aux fonctionnalités du multijoueur. Nous avons aussi modifié le code de Mirror pour y ajouter nos propres fonctionnalités comme le choix des personnages par les joueurs ou le blocage des joueurs voulant rejoindre le serveur alors que celui-ci est déjà complet ou en cours de partie. Pour cela, il a fallu utiliser le système de message de Mirror afin de créer des liaisons entre les deux joueurs. Le joueur 1 "host" a directement accès aux informations du serveur. En effet, il se trouve sur la même instance du jeu, contrairement au joueur 2 "client" où les nouveaux messages permettent de récupérer les personnages disponibles : notre jeu comporte deux personnages : Jack et Ellie. Chacun des deux joueurs choisit un personnage parmi ceux disponibles (ils ne peuvent pas avoir tous les deux le même personnage).

Enfin, nous avons ajouté un "authentificateur" afin que seuls les joueurs de notre jeu puissent se connecter au serveur. Sans celui-ci, n'importe quel jeu utilisant Mirror pourrait se connecter au serveur, ce qui peut causer de gros problèmes de sécurité.

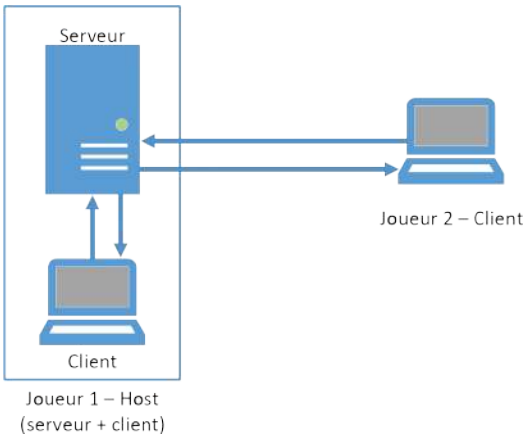

Figure 3 – Schéma de connexion classique entre deux joueurs

4.2.2 Développement continu du multijoueur

En outre, nous avons synchronisé les différents éléments qui en ont besoin à chaque fois que nous implémentions de nouvelles fonctionnalités. Dans ceux-ci, on retrouve notamment les sons à jouer sur les deux joueurs en même temps, chaque objet contenant des animations et/ou des déplacements, le lancement de la cinématique, de la partie, les particules... Il nous a aussi fallu créer un système de sauvegarde de tous les éléments qui peuvent changer au cours d'une partie afin de pouvoir restaurer des points de sauvegardes placés à différents endroits pour que les joueurs ne refassent pas le jeu depuis le début à chaque mort. Il est aussi nécessaire de répliquer certaines choses côté client quand celui-ci se connecte, par exemple, nous envoyons au nouveau joueur toutes les informations sur l'état des portes.

La compréhension de Mirror n'était pas évidente au début, car il y a très peu de documentation en ligne sur des sujets poussés d'utilisation de cette librairie, mais à force de régler les problèmes, on comprend mieux comment celle-ci fonctionne et celle-ci laisse une réelle liberté d'utilisation et de modification afin d'implémenter toutes les fonctionnalités voulues.

4.3 Scénario du jeu

4.3.1 Début de la partie avec cinématique

Le scénario commence avec une cinématique. Cette cinématique montre Jack et Ellie arriver en bateau jusqu'à un petit ponton à l'entrée du village de *Canaan*. Une fois arrivé, la partie peut commencer.

Figure 4 – Entrée de la ville

4.3.2 Reconstitution du message codé

Nous avons tout d'abord déterminé précisément où se situent les objets avec lesquels nous voulons interagir. Ces objets ont tous un lien logique avec l'histoire que nous avons imaginé. Les indices proviennent du frère d'Ellie, l'un des protagonistes de notre histoire.

Le premier objectif est de regrouper, à l'aide d'un travail d'équipe, des morceaux de papier qui permettent de recréer un message. Le travail d'équipe est nécessaire, car pour l'un des quatre papiers, Jack doit soulever un rocher tandis qu'Ellie récupérera le papier coincé dessous.

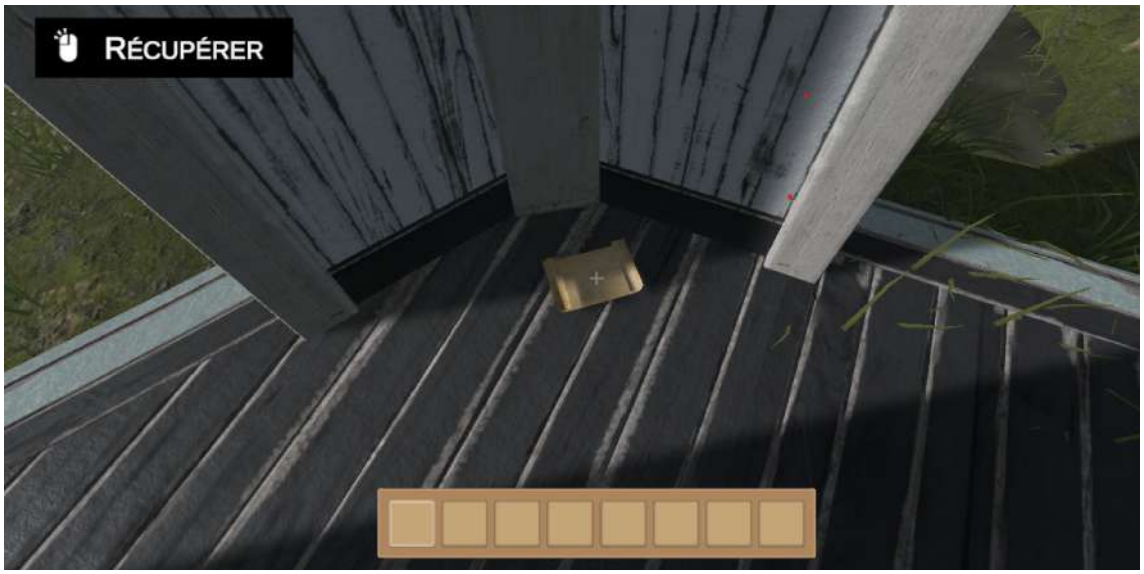

FIGURE 5 – Papier à récupérer

Ce message, qui n'est déchiffrable que lorsque Ellie possède les quatre papiers, indique aux joueurs vers quel bâtiment ils doivent se rendre ensuite. Les dialogues présents nous indiquent que Ellie est la seule à pouvoir déchiffrer le message, car c'est un code qu'elle avait inventé avec Tom lorsqu'ils étaient enfants.

4.3.3 Hôtel de ville, serre et Labyrinthe

Arrivés devant l'hôtel de ville, ils doivent trouver une clé cachée sous un banc à côté de l'entrée pour s'introduire dans l'immeuble. Cela leur permettra de se retrouver devant deux nouveaux indices. Le premier est le schéma du labyrinthe présent sur la carte, avec des croix rouges indiquant l'emplacement d'un potentiel objet à récupérer. Le second est une note avec marquée "Note : Faire l'inventaire des outils de la serre".

Ce second indice amène les joueurs à se rendre dans la serre qui est située derrière le bâtiment dans lequel ils se trouvent. Dans la serre, une bêche est présente au sol, elle nous permet de nous rendre dans le labyrinthe et d'y casser un tonneau qui contient une seconde clé. La bêche se cassera à ce moment pour ne plus avoir la possibilité de l'utiliser par la suite. Pour réussir à trouver le tonneau, les joueurs doivent mettre à profit leur coopération pour que le premier joueur, resté devant le schéma, guide le second qui s'aventure dans le labyrinthe.

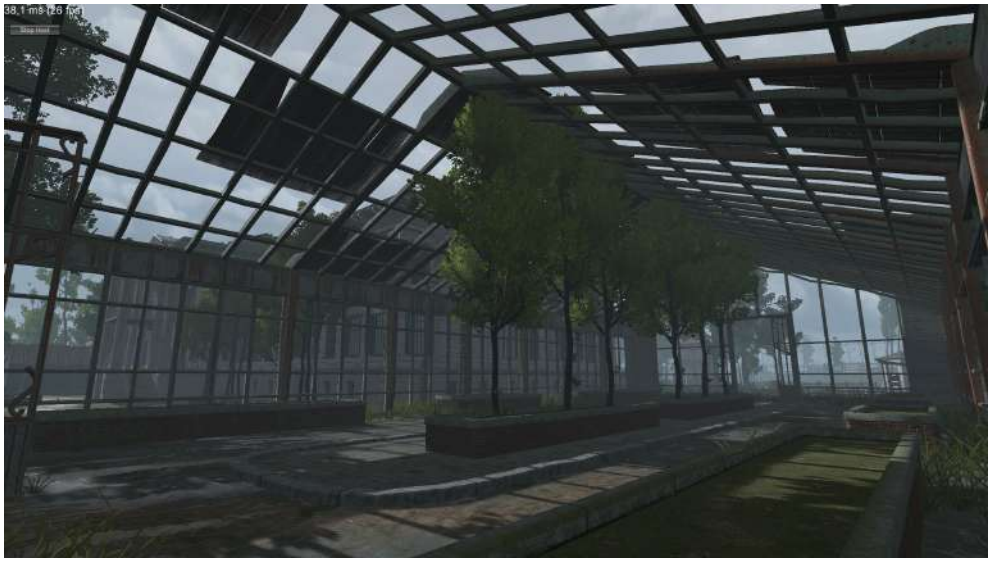

FIGURE 6 – La serre

4.3.4 Manoir

La clé trouvée permet d'ouvrir de nouveaux portails qui nous donnent accès à une seconde partie de la carte.

FIGURE 7 – Portail à ouvrir

Il faut ensuite trouver un bâtiment quelque peu éloigné du reste de la ville et s'y infiltrer à l'aide d'une fenêtre cassée.
Il est possible d'accéder depuis le début de la partie à un endroit sur lequel on verra deux gardes qui nous diront de repartir de là où l'on vient. Cette partie sert uniquement à attirer le joueur et lui proposer une meilleure immersion.

C'est dans cette maison que le premier garde fait son apparition. C'est donc à ce moment que le jeu devient non seulement un jeu de coopération, mais aussi d'infiltration.

Pour permettre une difficulté progressive, ce garde est peu agressif. Le joueur pourra donc les éviter aisément, mais il est désormais averti que des ennemis peuvent être présents. Ce garde effectue des rondes dans la maison, car il y a un passage secret au sol.

Figure 8 – Première IA rencontrée

4.3.5 Grotte et Domaine de Kelley

Lorsque les joueurs sont dans le passage secret, ils se retrouvent dans un tunnel. Ce tunnel comporte deux voies restreintes. L'une est bloquée par des lianes, la seconde contient un couteau déposé sur une boite, couteau qui permet de découper les lianes.

FIGURE 9 – La grotte sous la maison

Lorsque les lianes sont découpées, les joueurs ont accès à une échelle où se trouve une trappe en haut, trappe qu'ils ne peuvent ouvrir que s'ils interagissent en même temps dessus. Cette trappe les amène sur une île du nom de Domaine de Kelley, séparée du reste de la ville, mais visible depuis le début de la partie, l'île qui était protégée par les gardes qui nous parlent. Ils doivent encore une fois s'introduire dans la grande maison devant eux dans laquelle se trouve de nombreux gardes. Ils ont la possibilité de les tuer avec le couteau récupéré auparavant et d'ainsi ramasser leurs armes et munitions. Cette maison comporte un passage secret qui nécessite la coopération des joueurs, car il se trouve en dessous d'une horloge qui ne peut être poussée que par les deux joueurs.

FIGURE 10 – Horloge à pousser

4.3.6 Partie finale : le bunker

Après cela, ils tombent dans un bunker qui est la partie finale du jeu. Ils doivent aller jusqu'au MS 402 et le récupérer. Pour cela, ils doivent traverser une salle qui comporte quelques gardes, mais aussi un second couteau si les joueurs le trouvent. Après avoir réussi à traverser cette pièce, ils iront dans la dernière salle.

FIGURE 11 – Bunker Salle Principale

Cette salle comporte le plus de gardes vu depuis le début du jeu, les joueurs doivent donc mettre à profit leur intelligence pour réussir à éliminer tous les ennemis. Le MS 402 est situé au centre de la pièce et lorsque les joueurs touchent le manuscrit, le jeu s'arrête.

4.4 Création de la map

4.4.1 Ambiance de la carte

La création de la map s'est tout d'abord faite à l'aide d'un asset, qui est un ensemble de modèles 3D, nous ayant permis de choisir un environnement préfabriqué. Nous avons ainsi beaucoup remodelé la map initiale et avons rajouté différents assets qui permettent de la compléter pour qu'elle s'accorde au mieux avec nos idées. Il s'agissait d'une ville type post-apocalyptique, ce qui se rapprochait fortement de la ville abandonnée que nous voulions. De la végétation a été rajoutée pour rendre le tout moins lugubre.

FIGURE 12 – Vue aérienne de la map sans brouillard

Une reformation du terrain a été nécessaire dans de nombreux endroits, car l'eau était omniprésente dans la map. Cette eau est une barrière naturelle agréable, mais lorsqu'elle est trop présente, enlève des possibilités de déplacements pour le personnage. Pour autant, le terrain était ouvert à tous les endroits, certaines parties sont bloquées pour rendre les limites du terrain voulu plus facile à comprendre et que le joueur ne s'écarte pas de trop de la ville initiale.

FIGURE 13 – Avenue principale

4.4.2 Tunnel

Sous l'une des maisons, un tunnel a été fabriqué à l'aide d'un préfabriqué, mais aussi d'un assemblage de pierres et de morceaux de tunnels. Ils sont présents pour permettre une chute après un passage dans une maison, le changement d'ambiance étant plus visuel avec des pierres que l'asset du tunnel directement. Ce tunnel est accessible à l'aide d'un trou dans le sol de la maison, ce trou a été caché par un tapis.

Pour avoir la transition avec notre seconde partie du jeu qui se déroule dans un endroit plus éloigné géographiquement. Ce repère a été construit en créant et aménageant une colline un peu plus loin de la ville. De nombreux éléments de décors qui étaient éloignés du centre de la ville ont été supprimés pour réduire le nombre d'éléments à afficher et ainsi améliorer les performances en jeu.

Figure 14 – Tunnel souterrain

4.4.3 Murs invisibles

La map à subit des changements spécifiques aux ajouts du gameplay. Nous avons tout d'abord ajouté des murs invisibles qui ne laissent désormais plus la possibilité au joueur de sortir de la map, pour autant il peut toujours admirer le paysage plus éloigné.

Ces barrières sont disposées dans le but d'être bloqué de manière réaliste. Elles sont présentes sur des barrières, murs ou encore sur les plans d'eau auxquels le joueur n'a intentionnellement pas accès. De ce fait, des éléments de décors sont présents pour bloquer des passages, tel que des voitures accidentées, des piliers en travers du chemin ou encore des portails fermés. Pour autant le joueur à accès à de nombreuses parties de la carte bien que les lieux ne soient pas utiles lors de la réussite du jeu.

FIGURE 15 – Voitures accidentées permettant de bloquer le passage

4.4.4 Labyrinthe

Le labyrinthe a été modelé à partir de notre imagination. Il utilise des buissons présents dans notre asset principal. Ces buissons ont été modifiés pour obtenir le labyrinthe que nous voulions. Ce labyrinthe est donc assez large et haut pour permettre au joueur de ne pas se sentir enfermé tout en maintenant l'idée qu'il pourrait se perdre.

FIGURE 16 – Vue aérienne du Labyrinthe

Nous avons par ailleurs ajouté de l'herbe ou des arbres lorsque le scénario du jeu le requiert. Des éléments d'indications ont été rajoutés pour permettre de pouvoir jouer au jeu, mais aussi de se plonger dans l'univers de l'histoire. Ainsi, nous avons installé un panneau à l'entrée de la ville avec le nom de la ville et des panneaux directionnels. Il y a aussi le schéma du labyrinthe et une note dans l'hôtel de ville à l'étage.

FIGURE 17 – Schéma du Labyrinthe dans l'hôtel de ville

Nous avons aussi rajouté des fenêtres cassées ou d'autres ayant des composants de collisions. Ainsi, il n'est plus possible d'entrer dans un endroit sans que cela ne soit fait exprès. Il en va de même pour les portes : certaines peuvent être ouvertes, certaines nécessitent une clé et d'autres sont bloquées.

4.4.5 Bunker

La partie finale a été aménagée à l'aide d'un autre asset. Celui-ci est un bunker et est donc fermé. Il a été posé de telle manière qu'il soit dans sous la carte principale et que l'on ne puisse pas le voir de l'extérieur. Nous avons utilisé un modèle préfabriqué de ce bunker, mais nous l'avons fortement aménagé. Les pièces étant vides, tout a été ajouté de telle manière que le joueur puisse s'immerger dans le jeu et puisse combattre avec les ennemis. Les lumières ont aussi été retravaillées sur toute la carte et principalement dans le bunker, car elles demandaient trop de ressources et le jeu n'était plus optimal dans la partie du bunker.

FIGURE 18 – Vue extérieure du Bunker

4.4.6 Musiques

Nous avons aussi ajouté de nombreuses musiques libres de droits pour celles permettant à donner une ambiance dans les différentes parties de la carte. Les musiques sont présentes en tant que son spatial et permettent donc de changer d'ambiance en fonction des lieux. La musique commence dès lors que nous entrons dans la ville, puis change dans chaque bâtiment important. La musique est tout d'abord assez calme lors du début de la carte et devient plus intense au fur et à mesure que le joueur avance dans la partie.

4.5 Site web

Nous avons eu l'idée de créer un site internet structuré autour de quatre parties. Les quatre parties sont : "Notre jeu", "À propos", "Nous outils" et enfin "Nos documents".

Pour réaliser ce dernier, nous avons opté pour l'utilisation simple des langages HTML, CSS et JavaScript permettant la création d'un site statique idéal dans notre cas. Notre site est hébergé sur l'intégration continue de GitLab.

Nous avons commencé par la création d'une ébauche grâce à Figma (un logiciel de design de site web). Après avoir conçu l'idée globale de notre site, nous avons commencé par développer les éléments principaux tels que la barre de navigation et le pied de page. La barre de navigation est séparée en trois parties. À gauche le logo avec le nom du jeu, au milieu les quatre parties citées précédemment et à droite le bouton de téléchargement de notre jeu. Le pied de page quant à lui regroupe l'ensemble des réseaux que nous entretenons (Instagram et une adresse mail pour nous contacter).

La première partie est "Notre jeu" qui a pour objectif de présenter ce dernier à l'aide d'un texte et d'un carrousel de photo. Ce carrousel est très simple d'utilisation grâce à ses nombreuses fonctionnalités. On peut passer les images en cliquant sur les flèches ou en cliquant sur une des images en bas. Mais aussi grâce à un système de défilement fonctionnel avec le doigt, très utile sur petit écran.

En suit une partie "À propos" pour introduire la motivation de notre projet et notre groupe. Puis une partie sur "Nos outils" énumérant l'ensemble des outils principaux qui nous auront permis de réaliser ce projet. Chaque outil est cliquable et redirige vers le site officiel. Dans cette même partie, se situent nos ressources dans lequel, nous avons trouvé les sons et animations pour notre jeu.

Enfin, une partie archivant tous nos documents réalisés tout au long de notre projet permettant de retracer notre parcours et voir l'avancement du projet. Pour archiver nos documents, nous avons privilégié une présentation sous forme de frise chronologique avec un système d'étiquette pour plus facilement choisir les documents voulus. Nous avons cinq étiquettes qui sont : Cahier des charges, Rapport de soutenance, Plan de soutenance, Rapport de projet et dossier d'exploitation. Chaque étiquette est cliquable et permet d'activer ou non l'affichage des documents correspondants. Par défaut, chaque type est activé et chaque document possède une de ces étiquettes. De plus, tous les documents sont triés par ordre chronologique (du plus récent au plus ancien). Et le bouton bleu permet de télécharger le document sur son ordinateur.

Quand aucun document n'est sélectionné, un message est alors affiché et il disparaît lorsque au moins un document est sélectionné.

De même, nous avons limité le nombre de documents affichés à 3. Et un bouton pour afficher tous les éléments apparaît si plus de trois documents sont affichés.

Enfin, notre site est totalement adapté pour tout type de taille d'écran, aussi bien téléphone, tablette et ordinateur. Et toutes nos images ont été optimisées et compressées grâce au format *WEBP* qui est très adapté pour les sites internets.

URL de notre site : *https://a2nt.gitlab.io/ms-402-website*

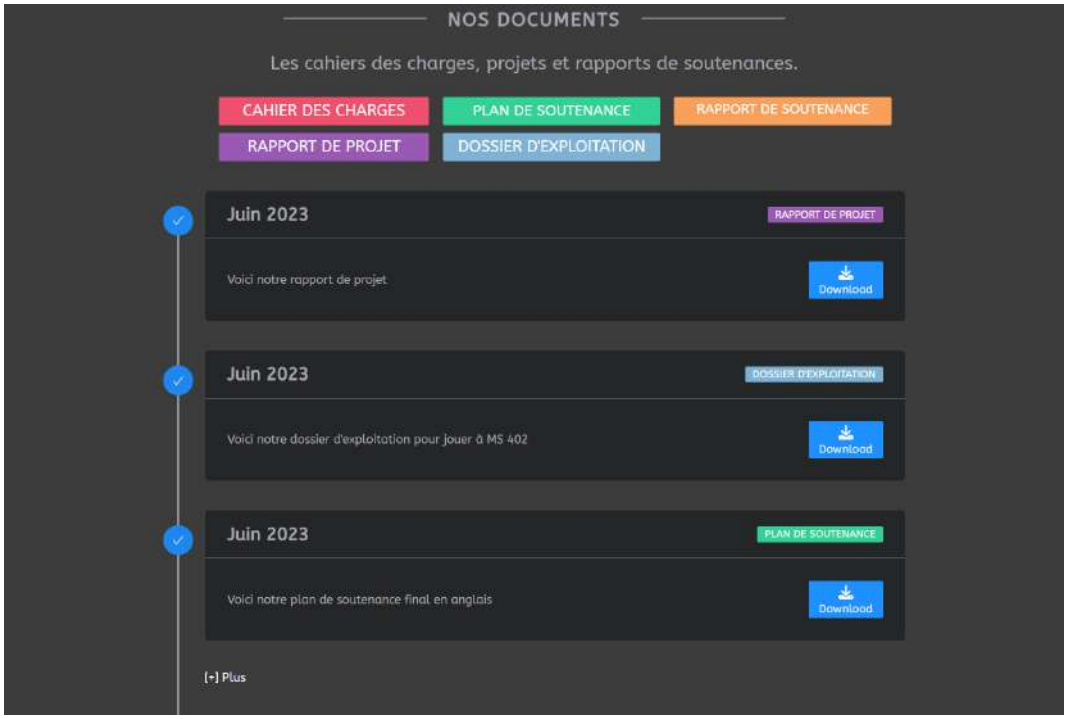

FIGURE 19 – Partie document - site Web

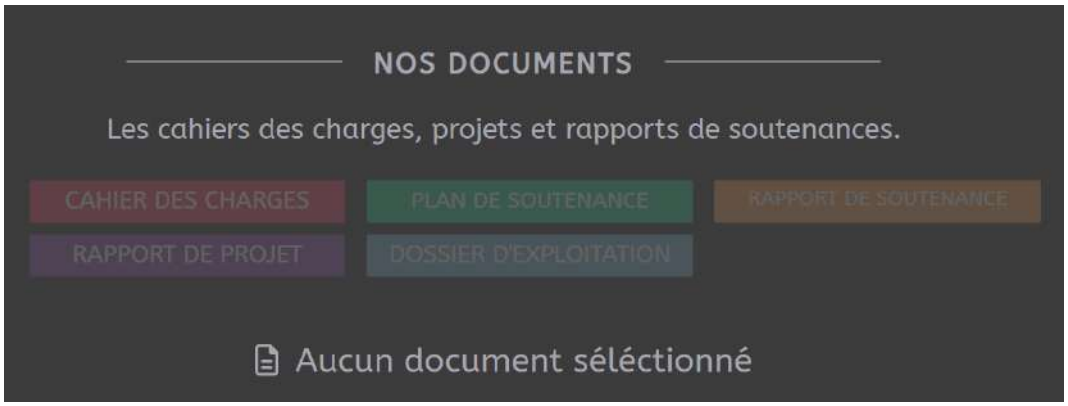

FIGURE 20 – Aucun document sélectionné

4.6 Menus et HUD

4.6.1 Menu principal

Nous avons créé un menu principal dynamique en 3D incrusté dans le décor du jeu.

FIGURE 21 – Menu principal soutenance 2

FIGURE 22 – Menu principal définitif

Concernant le fonctionnement du menu principal, celui-ci est composé d'un bouton "Jouer" qui se balance sur un bateau pour rendre le menu plus vivant. Il comporte également un garde (IA de notre jeu) qui reste sur place et possède une animation simplement dans le but de le rendre vivant. La patrouille du garde a été désactivée exprès pour le menu principal. Enfin, le menu possède un panneau de bois sur lequel sont superposés plusieurs objets textes. Il y a des objets sous-menus qui englobent chacun plusieurs objets textes, ce qui permet de gérer la navigation dans les menus avec du code en jouant sur la visibilité de ces objets. Pour l'implémentation de la navigation dans les menus, nous avons utilisé une pile de sous-menus, ce qui permet de parcourir les sous-menus et revenir au sous-menu précédent plus facilement.

FIGURE 23 – sous-menu Jouer

sous-menu Jouer –> Trouver une partie

4.6.2 Menu pause et paramètres

Un souci majeur que nous avons rencontré est le manque de fluidité du jeu lié aux assets importés qui étaient plutôt gourmands en ressources, ce qui impactait l'expérience de jeu ainsi que la capacité à tester le jeu pour certains membres du groupe (PC sans carte graphique dédiée). Pour rendre le jeu accessible à tous les membres du groupe et pour améliorer l'expérience de jeu des joueurs, nous avons ajouté un menu déroulant permettant de modifier la qualité de rendu avec six niveaux (Très basse / Basse / Moyenne / Haute / Très haute / Ultra), améliorant ainsi la fluidité du jeu.

FIGURE 24 – Menu déroulant - choix de la qualité de rendu

Nous avons ajouté un menu **pause** depuis lequel on peut quitter la partie, lire l'histoire du jeu et accéder aux paramètres.

FIGURE 25 – Menu Pause

Dans le menu **paramètres**, on retrouve le choix du niveau de qualité du jeu, le volume sonore, le réglage de la sensibilité de la souris en mode normal et en mode visée, l'activation/désactivation du mode plein écran ainsi que la possibilité de changer les touches associées aux différentes actions du jeu. Lorsque l'on est dans le menu pause ou dans les paramètres, les mouvements du joueur sont désactivés et le curseur apparaît. À l'inverse, lorsqu'on est en jeu, le curseur de la souris est désactivé et restreint à la fenêtre et les mouvements du joueur sont activés.

FIGURE 26 – Paramètres (21 touches paramétrables en défilant dans le menu)

4.6.3 Gestion des touches

Lors de la création de l'option de customisation des touches, nous nous sommes rendus compte qu'il n'était pas possible de modifier les touches de l'Input Manager par défaut de Unity en cours de partie. Il était seulement possible de les modifier au lancement du jeu, ce qui n'est pas très pratique. De plus, la majorité des jeux modernes offrent la possibilité à l'utilisateur de modifier les touches en jeu. C'est donc pour cela que nous avons créé notre propre Input Manager :

```
public static class MyInputManager
```

Le fait que cette classe soit statique nous permet d'y accéder facilement aussi bien depuis les paramètres (afin de changer les touches) que dans le système de déplacements du joueur et d'interactions avec les objets (afin de vérifier si une touche correspondant à une action a été pressée). Un attribut important de cette classe est un dictionnaire dont les clés sont des enums GameAction et dont les valeurs sont des KeyCode, c'est-à-dire des codes qui correspondent à des touches du clavier ou à des boutons de la souris.

Voici la liste des actions disponibles en jeu (la possibilité de voler est uniquement disponible pendant la phase de développement) :

Actions	Touches par défaut
Avancer	**Z**
Reculer	**S**
Aller à gauche	**Q**
Aller à droite	**D**
Sauter	**Espace**
Sprint	**R**
Toggle Sprint	**T**
S'accroupir	**Maj. gauche**
Voler (dev uniquement)	**F**
Interagir	**E**
Jeter un objet	**A**
Attaquer/Tirer	**Clique gauche**
Viser	**Clique droit**
Afficher les quêtes	**Tab**
Inventaire	**de 1 à 8 et molette de la souris**

4.6.4 Interactions

Par ailleurs, nous avons ajouté une fonctionnalité très utile pour le joueur, c'est la possibilité d'afficher un texte pour les différentes interactions. Ce texte est affiché grâce à un raycast partant depuis la tête du joueur et suivant la direction de la caméra. Pour une aide visuelle, nous avons rajouté une petite croix exactement en direction du raycast afin de plus facilement pouvoir effectuer nos interactions.

Par exemple, devant une porte, on peut effectuer plusieurs actions comme ouvrir et fermer la porte grâce à la touche d'interaction. Mais la porte peut aussi être verrouillée, dans ce cas aucune touche n'est renseignée comme aucune action ne peut être effectuée. Le fond est en noir pour un meilleur contraste.

FIGURE 27 – Exemple de messages d'interactions

De plus, l'affichage des touches d'interactions est dynamique. C'est-à-dire que pour n'importe quelle touche inscrite dans le dictionnaire, un affichage personnalisé est disponible. Par exemple, pour l'interaction de la récupération d'éléments, nous pouvez affecter plusieurs touches, comme la touche "w" mais aussi "contrôle gauche" tout en gardant un affichage agréable. Pour faire cet affichage personnalisé, nous avons superposé une image de fond de touche blanche et nous écrivons la valeur de la touche dans un texte au-dessus de l'image. La valeur de la clé étant renseignée dans le dictionnaire qui associe une touche à sa représentation sous forme de chaine de caractère.

FIGURE 28 – Messages d'interactions lorsqu'on change la touche de l'action

4.6.5 Choix des personnages (Avant/Après)

De plus, le choix des personnages (Ellie ou Jack) qui se faisait via l'interface Mirror, se fait désormais via le menu principal dans un sous menu créé à cet effet.

FIGURE 29 – Choix des personnages - Avant

FIGURE 30 – Choix des personnages - Après

4.6.6 Inventaire

Enfin, notre jeu est en coopération avec plusieurs énigmes, ce qui nécessite l'utilisation d'un inventaire. Cet inventaire est simplement composé d'un élément possédant 8 enfants pour les 8 emplacements disponibles. Ce qui permet plus facilement de changer de case sélectionnée en récupérant le i-ème fils.

Pour sélectionner la bonne case, il est possible soit de passer par la molette, soit par la touche de la case de l'inventaire correspondante. Visuellement, la case sélectionnée possède des bords plus clairs que les autres pour rapidement la différencier.

FIGURE 31 – Différence visuelle entre case sélectionnée et non sélectionnée

Lorsque Ellie possède les quatre bouts de papier, ces derniers se transformeront en message lisible. Seule Ellie a cette aptitude pour traduire les messages de son frère disparu. Le contenu du message est affiché uniquement lorsque son emplacement est sélectionné, ce qui permet de pouvoir le relire par la suite ou même de le donner à Jack pour qu'il puisse lui aussi le regarder.

4.6.7 Quêtes et dialogues

Lorsque nous avons mis en place les différentes quêtes de notre jeu, nous nous sommes rendus compte qu'il manquait des indications pour l'utilisateur et un fil conducteur à notre jeu. Pour pallier ce problème, nous avons décidé de mettre en place plusieurs dispositifs visuels.

Tout d'abord, nous avons mis en place des dialogues entre nos deux personnages en lien avec l'histoire du jeu. Ces derniers se déclenchent à des moments clés, c'est-à-dire lorsqu'un joueur/lorsque les deux joueurs entre/entrent dans un endroit donné, ou bien lorsqu'un joueur récupère un objet, ou encore lorsqu'il s'éloigne trop d'une zone de recherche, ce qui permet de restreindre le joueur à une certaine zone sans pour autant casser l'immersion.

Pour ce qui est de l'implémentation des dialogues, nous avons tout d'abord écrit les dialogues sur Microsoft Word en notant l'évènement déclencheur pour chaque dialogue. Ensuite, pour stocker et réutiliser les dialogues dans Unity, nous avons mis en place un parseur qui nous permet d'écrire directement les dialogues dans un fichier texte en le délimitant par blocs. Le début d'un bloc de dialogue est identifié par un '#' suivi d'un nom de bloc afin de pouvoir le lancer dans le jeu en réutilisant cet identifiant. La fin d'un bloc correspond au début du bloc suivant ou bien à la fin du fichier texte. Il est possible de faire un retour à la ligne dans un dialogue afin de ne pas avoir des lignes trop longues. Les dialogues sont séparés par des sauts de ligne. Il est également possible de spécifier la durée du dialogue en mettant un '|' suivi de la durée exprimée en seconde à la fin du dialogue en question. Si la durée n'est pas spécifiée, la durée par défaut est calculée en fonction du nombre de mots dans la phrase et d'un débit de parole que nous avons défini à 170 mots/minute, ce qui est légèrement plus lent que la moyenne, de telle sorte que le joueur ait le temps de lire les dialogues. Enfin, il est possible de définir des temps de pause au sein même d'un bloc de dialogues en utilisant '/' suivi de la durée en seconde du temps de pause voulu. Voici un exemple d'un dialogue présent dans notre jeu au tout début de partie :

Arrival

/5

Ellie – Je crois qu'on y est enfin Jack, c'est Canaan.

Jack – J'ai bien cru qu'on n'y arriverait jamais,ce village est tellement isolé du reste du monde.Mais... qu'est-ce que l'on fait maintenant ?

Ellie – Je ne sais pas trop, Tom nous a dit de ne pas venir mais je suis persuadée qu'il nous a laissé quelque chose avant que ça tourne mal pour lui.

Jack – Tu as sûrement raison, explorons l'entrée de la ville, mais restons discrets ! Cet endroit ne me dit rien qui vaille...

Ellie – Je suis bien d'accord avec toi...|**2**

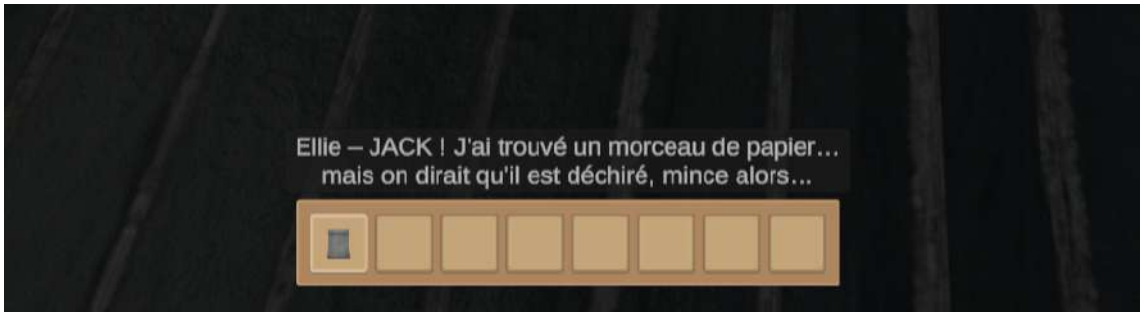

FIGURE 32 – Exemple de dialogue

De plus, nous avons implémenté un système d'affichage des quêtes qui permet de garder un œil sur la quête en cours afin de guider un minimum le joueur. Tout comme les dialogues, les quêtes se déclenchent à des moments clés : par exemple, la première quête est de se rendre au portail pour commencer le jeu. Une fois que les deux joueurs sont devant le portail, le portail s'ouvre et le joueur doit fouiller la zone. Lorsqu'un des deux joueurs trouve le premier papier (bout de message codé), la quête indique de trouver les autres morceaux.

FIGURE 33 – Exemple de dialogue

La zone de recherche est naturellement restreinte par les dialogues. En effet, lorsque le joueur s'éloigne de la zone de recherche, un dialogue se lance dans lequel le joueur se fait une réflexion à lui-même : *"Il y a peu de chances que les papiers soient autant dispersés, j'ai certainement oublié de regarder à certains endroits."* L'affichage des quêtes est activable/désactivable via une touche changeable depuis les paramètres (par défaut, il s'agit de la touche Tab).

4.6.8 Avertissements et aides

Nous avons également mis en place un système d'avertissements qui s'affichent lorsque le joueur utilise le mauvais outil ou bien lorsque le chargeur d'un pistolet est vide. À noter que lorsque le chargeur est vide, un son se déclenche pour en avertir le joueur afin de rester dans l'immersion, l'avertissement se déclenche seulement si le joueur continue d'essayer de tirer.

Enfin, pour que le joueur se familiarise dès le début avec les touches importantes, des aides sont affichées pour activer les quêtes ou encore accéder aux paramètres.

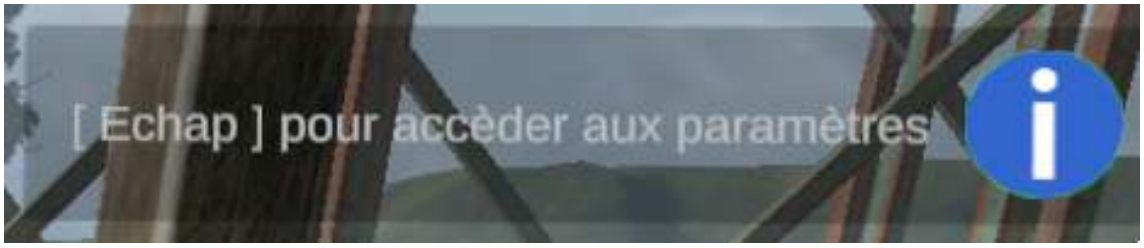

FIGURE 34 – Aide paramètres

FIGURE 35 – En attente d'un 2ème joueur et aide pour l'affichage des quêtes/missions

4.7 Mécaniques du jeu

4.7.1 Déplacements de base

Les mécaniques du jeu sont essentielles pour se déplacer dans un jeu. Nous les avons donc implémentées dès le début en commençant par les déplacements (marcher, courir, sauter, s'accroupir et temporairement, voler). Nous avons de plus créé notre propre gestionnaire de touche afin de pouvoir les modifier à notre guise au cours d'une partie.

L'application des mouvements se fait grâce à un vecteur en 3D qui s'applique sur le rigidbody (un composant Unity qui gère la physique). Ce dernier permet de garantir un réaliste par rapport au déplacement et collision.

Ensuite, pour le saut, nous appliquons une gravité qui évolue en fonction du saut pour garantir une allure parabolique. De plus, pour éviter les doubles sauts, nous avons implémenté un raycast vers le bas dépassant un tout petit peu de la taille du personnage (de 0.1). Ainsi, il est possible par exemple de sauter seulement si cette fonction retourne vrai.

```
public bool CheckIfOnGround()
{
    return Physics.Raycast(transform.position, -Vector3.up,
      transform.localScale.y + 0.1f);
}
```

Les mouvements pour courir et s'accroupir sont assez similaires à ceux de marcher hormis la vitesse qui est multipliée respectivement par 1.5 et 0.5. La direction du mouvement quant à elle est toujours dirigée selon les axes de la caméra.

Nous avons aussi développé une fonctionnalité qui ne sera pas accessible dans la version définitive, il s'agit de la touche pour voler. Une fois en vol, toutes les collisions sont désactivées et la vitesse multipliée de façon progressive jusqu'à un facteur 20 pour pouvoir parcourir la carte assez rapidement.

4.7.2 Rotation des caméras

En parallèle, il a été indispensable de créer toute la logique autour de la rotation des caméras. La rotation sur l'axe horizontal se faisant naturellement en fonction de l'axe "Mouse X" présente de base sur l'input manager. Cette rotation est ensuite appliquée au rigidbody afin de pouvoir être vu au travers du multijoueur. La rotation sur l'axe verticale dépend de l'axe "Mouse Y" mais n'est cependant pas appliqué au rigidbody. En effet, nous ne voulons pas que notre personnage puisse tourner selon cet axe sous le risque de pouvoir voir son personnage à l'envers. Nous avons donc simplement appliqué la rotation sur les caméras en précisant des limites de 80 degrés et -80 degrés pour éviter de pouvoir faire des tours de caméra sur l'axe verticale.

Par ailleurs, nous avons ajouté les possibilités de ramasser un objet, soulever des pierres, ouvrir des portes, creuser et monter aux échelles. Nous en reparlerons plus en détail dans la partie **Animations**.

4.7.3 Actions liées à l'inventaire

La fonctionnalité de ramassage est fortement liée à l'inventaire. Par exemple lorsque le personnage récupère un objet, il est supprimé de la carte et apparait dans son inventaire. De même pour faciliter la coopération et les échanges d'éléments entre Jack et Ellie, nous avons rajouté la possibilité de poser un élément de son inventaire au sol. L'élément posé est soumis à la gravité et tombe jusqu'au sol. Toutes ces mécaniques autour de l'inventaire ont été synchronisées en multijoueur. Ainsi si un joueur pose un élément ou prend un élément, l'autre joueur le verra et pourra récupérer un élément posé par l'autre joueur dans son inventaire.

Toujours pour augmenter la coopération, nous joueurs ont des habiletés différentes, Ellie possède la possibilité de traduire le message de son frère tandis que Jack peut soulever les pierres.

Toutes ces mécaniques permettent de mettre en place un gameplay avec des énigmes poussant à la coopération et à la recherche.

Tous les outils de notre jeu sont affichés dans la main du joueur lorsqu'il les sélectionne. Ce qui est indispensable comme ces outils sont utilisés au travers d'animations. Comme pour casser le tonneau avec la bêche, couper les lianes avec le couteau, ou tirer avec le pistolet.

4.7.4 Gestion du système de tir

Notre jeu fait enfin intervenir des IA qui joue le rôle de Guard et doivent nous empêcher de récupérer le manuscrit caché. Nous avons donc implémenté des systèmes de tir. Lorsque le joueur possède une arme dans son inventaire et qu'elle est sélectionnée, il est possible de viser (clique droit par défaut). En visé l'arme se place

en bas à droite de l'écran avec la vision sur les mains pour renforcer encore une fois l'immersion dans l'environnement. En visé, la touche clique gauche par défaut permet de tirer. Il apparait alors des particules à la sortie du pistolet en plus d'un son. Si la balle touche un élément dans sa distance de tir, des particules d'étincelles y seront affichées rapidement.

4.8 Intelligence Artificielle

4.8.1 Préparation de l'implémentation

Nous avons commencé par nous documenter sur comment implémenter une intelligence artificielle (IA) grâce aux outils de Unity. Puis, nous avons commencé la réalisation de celle-ci sur notre carte de développement du jeu.

Pour ce faire, nous avons utilisé le "navMesh" de Unity qui permet de mémoriser, dans un fichier, tous les endroits de la carte où l'ennemi pourra se déplacer. Pour détecter si un joueur est dans le champ de vision de l'IA, nous utilisons des "raycast" qui permettent de vérifier en ligne droite si le joueur est dans cette direction. En ajoutant une multitude de "raycasts" devant l'IA, nous simulons donc son champ de vision.

FIGURE 36 – Visualisation des raycasts de l'IA

4.8.2 Développement de l'IA

Nous avons créé une logique, permettant de donner des ordres à chaque IA. Lorsqu'un joueur est détecté, l'IA se déplace vers lui et passe en alerte "élevé", pendant ce temps qui dépendra de chaque IA afin de créer des niveaux de difficulté différents (5 à 15 secondes), l'IA connait la position du joueur et se déplace vers celui-ci. Si au bout de ce temps, le joueur n'est plus dans le champ de vision, l'IA baisse son niveau d'alerte jusqu'à retourner à sa position initiale ou de reprendre sa patrouille.

Ensuite, nous avons amélioré le système de raycast, en réalisant aussi un balayage haut-bas, afin de détecter un joueur qui serait un peu plus au-dessus ou en dessous de nous. Nous avons ajouté le système qui permet à l'IA de tirer sur les joueurs et de leur infliger des dégâts. Le système de patrouille a été revu et chaque IA a une patrouille associée. Quand celle-ci tire, il y a un pourcentage de réussite qui dépend du niveau de difficulté qu'on l'on attribue. Plusieurs autres paramètres permettent de régler la difficulté des IA, dedans, on retrouve la distance de tir et la distance maximale pour détecter d'un joueur dans son champ de vision. De plus, nous avons implémenté, en relation avec les autres tâches, les animations, les bruits, la gestion de la vie du garde, de la mort de celui-ci et tous les autres éléments qu'ont besoin les IA et les joueurs pour jouer et s'amuser.

Figure 37 – Schéma d'une patrouille d'une IA au domaine de Kelley

4.9 Animations

Dans l'optique de réaliser un jeu plutôt réaliste et en multijoueur, après avoir trouvé les modèles 3d de nos deux personnages, nous devions implémenter des animations sur ces derniers. Nous avons trouvé nos animations et nos personnages majoritairement sur le site de Mixamo répertoriant une vaste gamme de personnages et d'animations gratuites ainsi que pour certaines sur l'Asset Store de Unity.

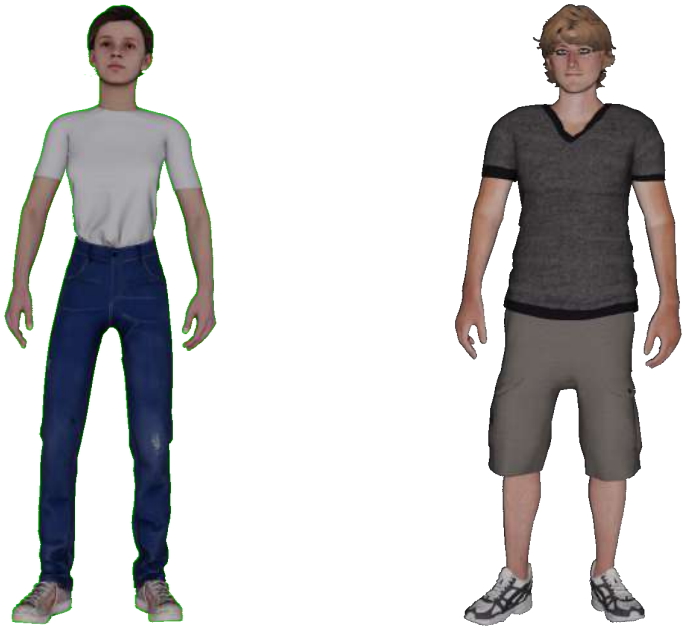

FIGURE 38 – Nos deux personnages Ellie et Jack

4.9.1 Gestion des caméras des joueurs

Nous avons commencé par mettre la caméra du joueur dans la tête du personnage, mais nous avons rencontré le bug dit du "clipping" lorsqu'une animation était jouée. La caméra étant située à l'intérieur du personnage (le problème persiste également si la caméra se situe devant les yeux du personnage), ce dernier obstrue la vue de la caméra sur l'extérieur. Ainsi, pour contourner cette contrainte, au lieu de désactiver la vision du joueur pour le personnage hôte de son propre corps uniquement, nous avons opté pour une option plus complexe afin que le corps du personnage soit tout de même visible par son personnage. Cette option permet entre autres de voir le bas du corps du personnage quand celui-ci regarde vers le bas ainsi que les bras lorsqu'il court par exemple. Tout cela étant impossible avec l'option évoquée précédemment avec une seule caméra.

Pour ce faire, nous avons utilisé deux caméras, une caméra située devant les yeux du personnage qui affiche tout sauf le corps du personnage hôte et une autre qui affiche seulement le joueur hôte

Par la suite, nous avons dû relier cette gestion des caméras avec le multijoueur pour exécuter l'action d'attribution de "layer" spécifique seulement sur le personnage hôte. Les layers permettent aux différentes caméras de capturer uniquement les bonnes parties du jeu voulu (dans notre cas soit seulement le personnage, soit tout sauf le personnage). Cette dernière permet de renforcer l'immersion du joueur dans l'environnement, comme voir les bras lorsque le personnage court pour accentuer le sentiment de vitesse. Une fois les caméras fusionnées, on obtient une vision plutôt réaliste sans clipping apparent. De plus cette méthode avec différentes couches permet de donner une priorité lors du rendu sur le joueur comme la caméra qui sélectionne seulement le joueur est mise en premier plan par rapport à la caméra qui affiche le reste. Ainsi si le joueur rentre une partie de son corps dans un objet du décor, il verra toujours tout son corps.

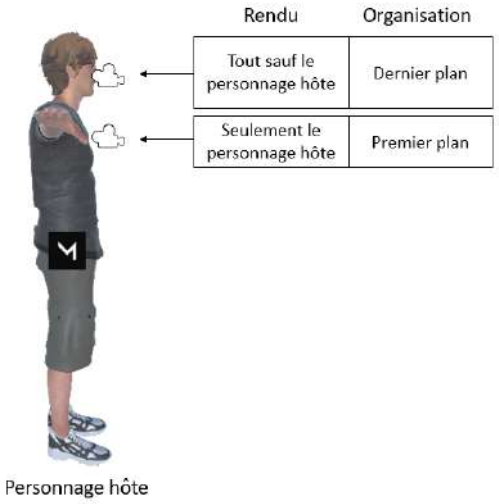

FIGURE 39 – Système de double caméra

4.9.2 Animations des joueurs

Après avoir réfléchi sur le système de gestions de caméras, il a fallu gérer les transitions entre les différentes animations. Nous avons ainsi répertorié l'ensemble des touches du jeu sur un input manager que nous avons nous même codé pour pouvoir changer aisément nos touches par la suite, car l'input manager de Unity ne permet pas de modifier nos touches une fois la partie lancée.

Sous Unity, les transitions entre les animations se gèrent grâce à un système de nœuds semblable à un graphe. Nous avons donc commencé par énumérer l'ensemble des animations de base de notre personnage pour pouvoir réaliser plus facilement la logique du graphe qui en découle et éviter des comportements inattendus ou des animations inaccessibles.

Par exemple, nous ne pouvons pas courir si le joueur ne marche pas, mais seulement s'il marche et appuis sur la touche pour courir. De même pour la priorité de certains déplacements. Par exemple s'accroupir est prioritaire devant courir.

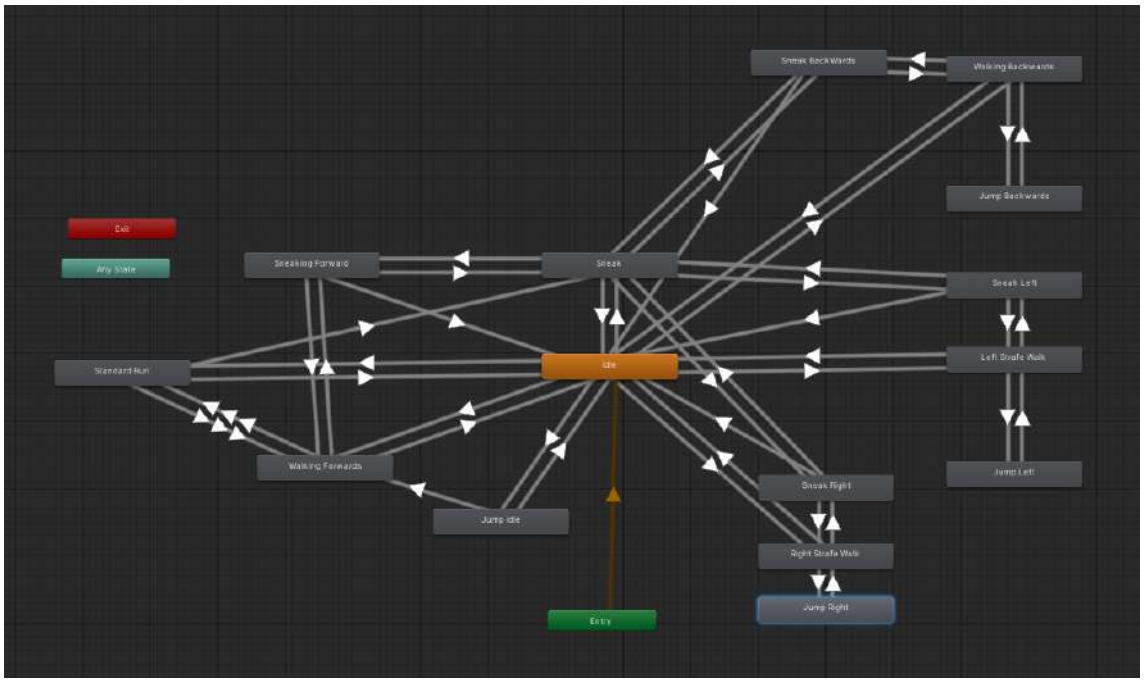

FIGURE 40 – Aperçu du graphe des animations

Pour certaines animations, il était important d'instaurer un ordre de priorités plus important. Par exemple, nous devrions pouvoir ouvrir les portes depuis n'importe quels mouvements de base. Pour ce faire, nous avons créé plusieurs niveaux dans l'animator (composant de Unity pour gérer les transitions entre animations) du joueur. Plus le niveau est bas, plus il a une forte une priorité, ce qui permet de basculer sur une animation d'un niveau supérieur depuis n'importe quelles animations d'un niveau inférieur.

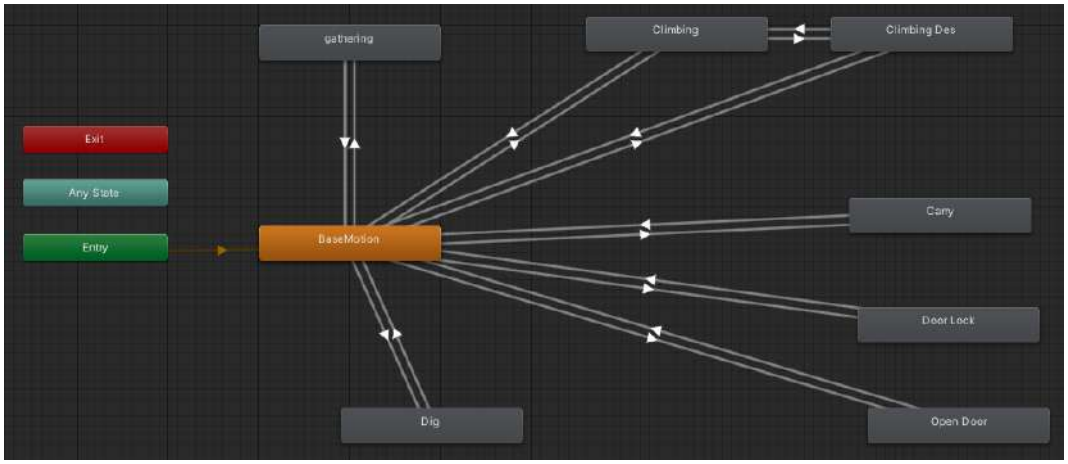

FIGURE 41 – Animator des mouvements prioritaires du joueur

Ce système est très performant surtout combiné à un autre outil de Unity qui sont les "Avatar masks". Ce dernier permet de sélectionner une partie spécifique à animer et de le fondre avec les autres parties du corps qui pourront être animées avec d'autres animations. Ce qui est très utile et augmente le réalisme dans certains cas. Par exemple pour une animation de tir, il faut garder seulement la partie haute du corps. La partie basse quant à elle devra conserver les animations de base de déplacement. Ce qui permet ainsi d'avoir une animation de tir qui fonctionne peu importe les déplacements du joueur. Si le joueur tir en marche ou en course l'animation des pieds qui marche ou court sera bien respectée.

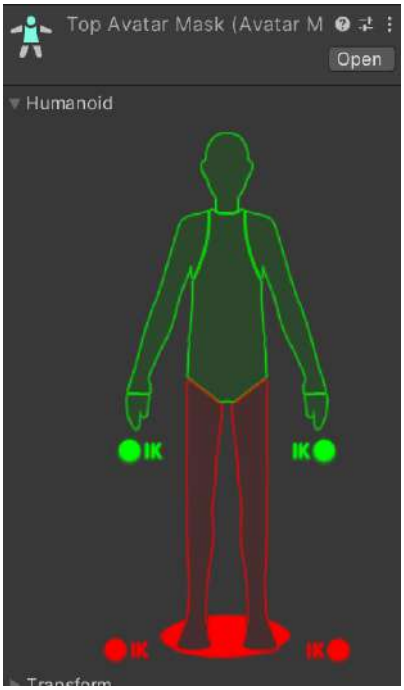

FIGURE 42 – Aperçu d'un Avatar Mask utilisé pour animer seulement la partie haute du corps.

Nous avons utilisé ce principe pour beaucoup d'animations comme pour l'animation qui soulève le rocher ou encore pour l'animation du couteau

Pour certaines animations, il est nécessaire d'effectuer une certaine action à un instant précis, pour ce faire, on a utilisé les "Animations Event" de Unity qui permettent cela. Cette dernière permet d'appeler une fonction dès que l'animation atteint une certaine image clé. Par exemple, pour l'animation de récupération d'objet, on a 3 "Animations Events". Celui au début et à la fin permettent respectivement de désactiver/activer les déplacements pour le joueur et éviter qu'il parte avant que l'animation ne se termine. Tandis que celui du milieu permet de supprimer l'élément ciblé à partir du moment où l'animation de la main semble prendre l'élément.

Nous avons utilisé ce principe pour d'autres animations qui ont besoin d'exécuter une action durant une image clé très précise d'une animation. Comme pour l'animation du couteau où les "dégâts" doivent être envoyés seulement quand le couteau est en train de réellement couper et non dès le début de l'animation. Ou encore pour l'animation de tir où la munition et tout le système d'animations/particules doivent se lancer au moment propice.

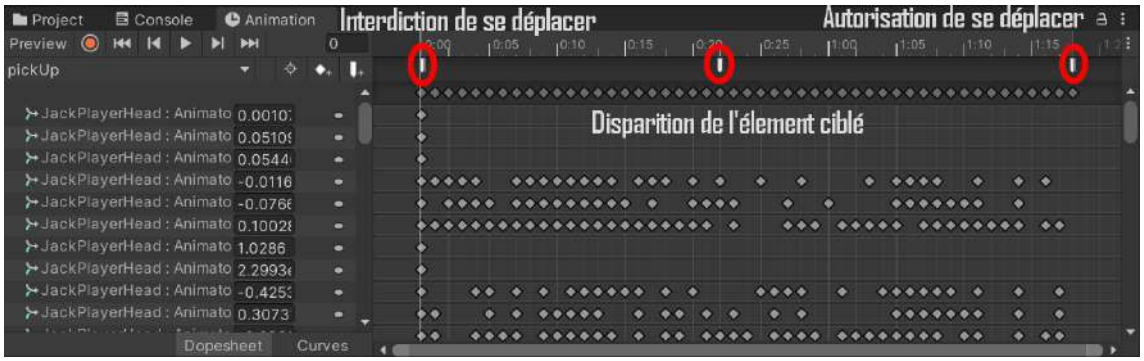

FIGURE 43 – Chronologie de l'animation de ramassage avec les "Animation Events"

4.9.3 Éléments exterieurs

Par ailleurs, nous avons dû ajouter des animations sur des éléments extérieurs aux joueurs comme les portes ou la pierre que l'on peut soulever.

Premièrement les portes, il est composé de principalement de deux animations, l'ouverture et la fermeture. Cependant, il arrive que l'ouverture soit dans le mauvais sens. Une porte qui donne sur l'extérieur devrait plutôt s'ouvrir de l'intérieur. Ainsi, nous avons rajouté un paramètre pour savoir si la porte doit s'ouvrir normalement ou dans l'autre sens grâce à la variable "invert".

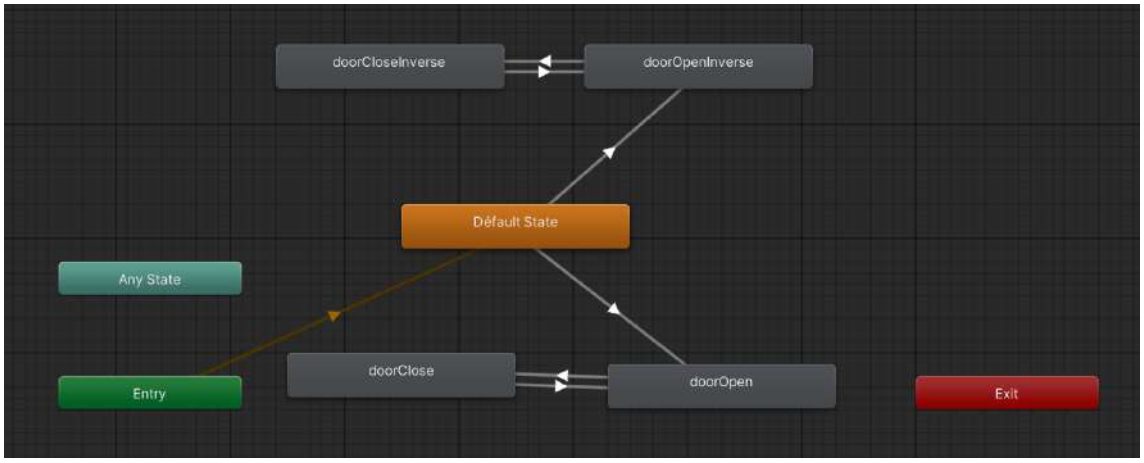

FIGURE 44 – Animator des portes

De même, l'animation de la pierre est commandée par un animator. Ce dernier est un peu différent avec l'animation principale qui est l'animation de la montée (la chute est l'animation montée dans le sens inverse). Lorsque la touche d'interaction (toujours récupérée depuis notre inputManager par la class MyInputManager) est pressée, l'animation "lift" est jouée. Et dès que cette touche est relâchée, la pierre retombe.

On a suivi le même principe pour toutes les autres animations extérieurs comme la trappe ou la pince.

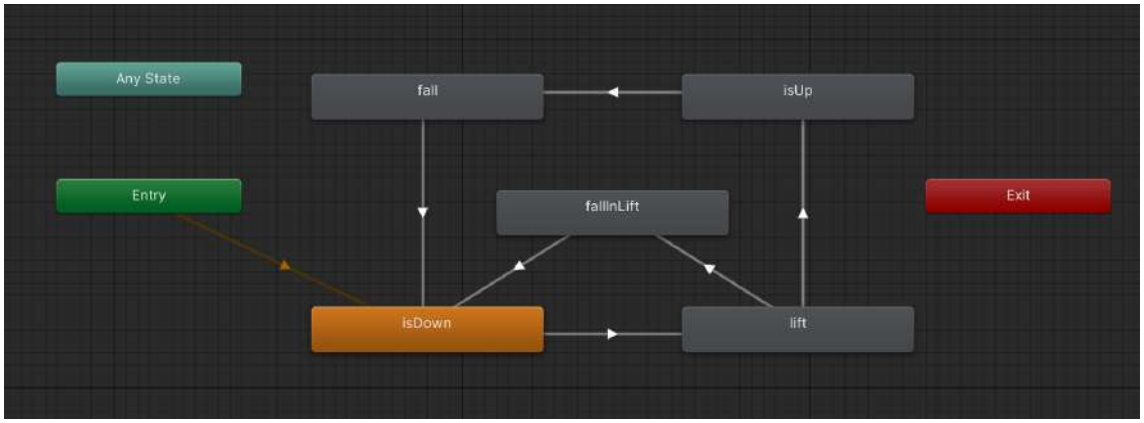

FIGURE 45 – Animator des rochers

4.9.4 Caméras

Par ailleurs, certaines animations n'avaient pas un rendu satisfaisant à la caméra, ou ne respectaient pas une logique réaliste. Par exemple lorsque le personnage s'accroupit, la caméra doit se baisser. Lorsqu'il monte à l'échelle, nous devons voir les bras du joueur en haut de la caméra...

Pour ce faire, nous avons développé des fonctions permettant une gestion parfaite de nos deux caméras. Ces dernières permettent de faire des translations sur l'axe Y de Unity de façon très douce. Ces translations ont été très utiles pour les animations pour s'accroupir, pour monter aux échelles ou encore lors de l'animation de récupération d'objet avec une phase dans laquelle les caméras descendent avant de remonter à la fin de l'animation.

4.9.5 IA

Par ailleurs, il a fallu implémenter le système d'animation pour les IA. Il s'inspire fortement de l'animator des joueurs, mais il a une particularité, l'animation de repos est aléatoire. Ce qui est important pour éviter des animations qui pourrait apparaitre trop synchronisé et donc pas naturel de nos IA. Pour réaliser ces animations aléatoires, il a été utile d'utiliser les "sub state machine" associé à un index qui change à chaque changement d'animation. Et lorsque l'animation de repos est de nouveau active, elle sélectionne là sous animations de repos qui dépend de l'index aléatoire.

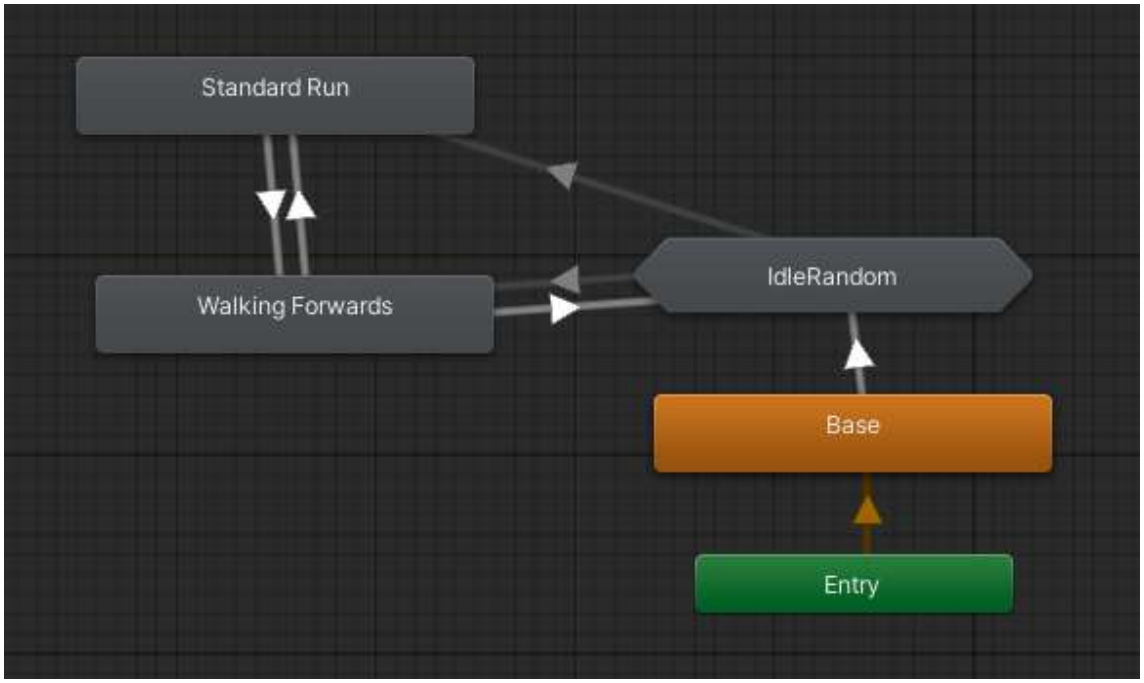

Figure 46 – Animator des IA

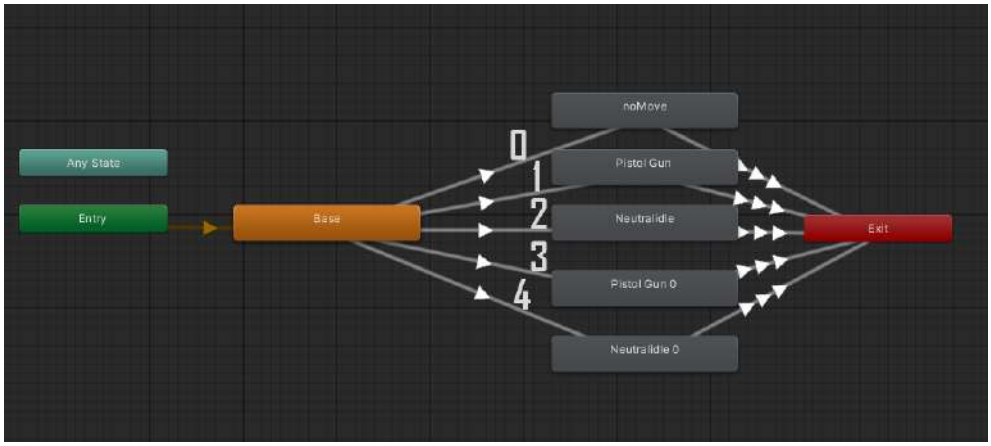

FIGURE 47 – Gestion des animations aléatoires des animations de repos des IA

4.9.6 Animation spécifique des os

Enfin pour l'animation de tir, il a fallu trouver une solution pour faire une rotation sur un os du bassin afin qu'en multijoueur l'autre joueur puisse comprendre la direction vers laquelle il vise. Mais nous nous sommes confrontés à un problème : Pour effectuer une transformation sur un os qui est déjà animée par un animator, il est obligatoire d'effectuer la transformation après l'application de l'animation par l'animator. Ainsi la seule solution est d'effectuer la rotation dans la fonction "LateUpdate" qui est la dernière à être exécutée, donc après la mise à jour de l'animator.

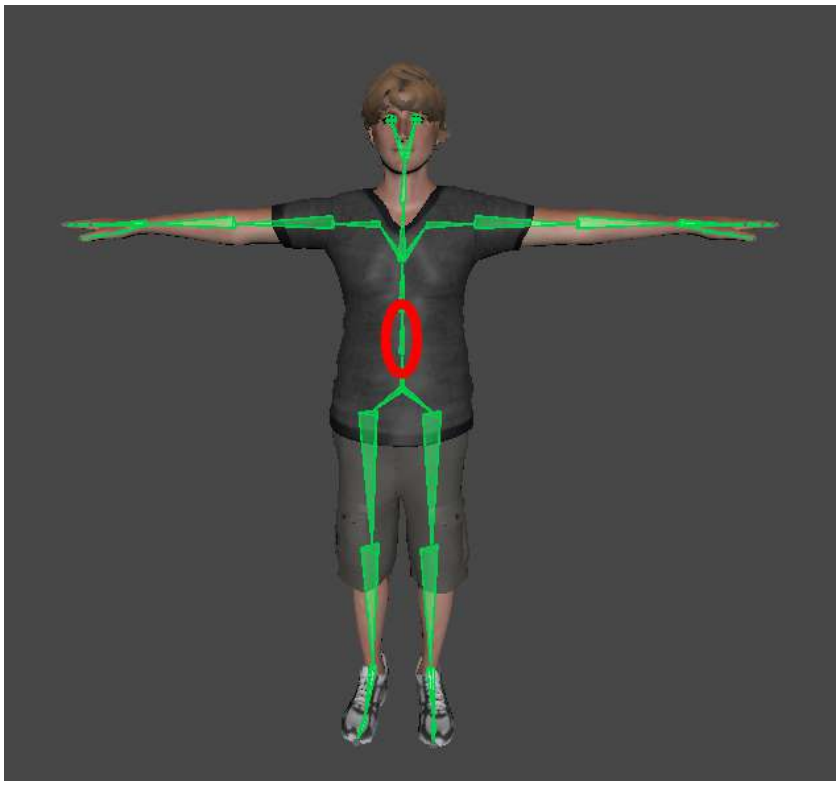

FIGURE 48 – Visuel de l'os qui subit une rotation lorsque le joueur est en train de viser

4.9.7 Animation de la cinématique

Nous avons créé la cinématique sur une scène différente de notre map principale. Cela nous a permis de créer les éléments que nous voulions pour cette cinématique.

Tout d'abord, nous avons déplacé le bateau à l'aide d'images clés. Ces images ont été placées en fonction de la trajectoire que nous voulions faire prendre à ce bateau. Nous lançons cette animation avec l'animator *controller*.

La caméra se déplace aussi à l'aide d'images clés, elle lui permet d'avancer et d'effectuer des rotations.

4.10 Organisation

Afin de synchroniser les fichiers de travail, nous avons créé deux dépôts GitLab : un pour le projet Unity et l'autre pour le site web. Pour le dépôt du projet Unity, nous avons ajouté un fichier ".gitignore" dans le but de ne pas synchroniser les fichiers "poubelles". Ces fichiers sont créés automatiquement, mais ne sont pas utiles. De plus, ils peuvent poser des problèmes lorsque l'on fusionne les différents travaux ensemble.

Nous travaillons aussi avec une branche de travail par tâche que nous fusionnons plusieurs fois par semaines sur notre dépôt *GitLab*.
. Pour le site web, nous avons implémenté une intégration continue. Celle-ci permet d'utiliser "GitLab Pages" et donc d'héberger automatiquement le site.

4.11 Communication

Dans l'optique de créer un jeu, il est essentiel de pouvoir le faire connaître au plus grand nombre. D'où notre intention de créer des comptes sur Instagram ainsi qu'une adresse mail de contact. Ces derniers sont accessibles sur notre site web au niveau du pied de page.

Compte Instagram : ms402project
Adresse mail : ms402.contact@gmail.com

Nous avons essayé d'alimenter notre compte Instagram avec une première bande-annonce et des captures de notre jeu pour présenter notre projet et notre groupe.

Figure 49 – Visuel de notre compte instagram

4.12 Jaquette

Nous avons aussi créé notre propre jaquette pour ce projet en utilisant une charte graphique semblable à l'ambiance de notre jeu.

FIGURE 50 – Visuel de notre jaquette

FIGURE 51 – Visuel de notre maquette

5 Post-mortem

Ce projet de deuxième semestre à Epita était notre premier long projet en groupe. Il nous a été en globalité bénéfique et a permis de découvrir le travail de groupe pendant une longue durée de six mois ainsi que la façon de s'organiser efficacement.

5.1 Erwan

Pour ma part, j'ai beaucoup apprécié la manière dont nous nous sommes organisés, cela a permis à chacun de développer des compétences dans les différentes tâches que l'on s'était réparties en travaillant efficacement grâce aux branches Git. J'ai beaucoup appris quant à l'utilisation de Git, du langage LaTeXet bien évidemment de Unity. En dehors de l'aspect technique, il y avait une très bonne ambiance dans le groupe.

5.2 Alexis

Cela a été un réel plaisir de réaliser ce projet au sein de notre groupe. Cela m'a permis d'apprendre plein de choses, qu'elles soient techniques, mais pas que. Nous étions très complémentaires et cela nous a permis d'avancer ensemble, d'apprendre des uns et des autres et de mener jusqu'à sa fin un projet à long terme. L'ambiance était très agréable, mais elle permettait de travailler efficacement et régulièrement. Nous étions organisés et nous communiquions beaucoup.

Au final, c'était une très bonne expérience, nous repartons avec des connaissances techniques et de gestion de projet ainsi qu'avec un projet terminé et un groupe d'amis (mais quelques heures de sommeil à rattraper).

5.3 Nathan

J'ai trouvé cette expérience très enrichissante aussi bien au niveau technique que au niveau du travail de groupe. Nous avons réussi à mettre en place une bonne communication au sein de notre groupe ce qui nous a permis d'avancer efficacement. Apprendre à s'organiser, à rechercher par ses propres moyens et faire face aux difficultés ont été pour moi mes axes de progressions majeurs. J'ai également pu apprendre à utiliser de façons plus approfondies de nombreux logiciels ou notions comme Git ou Blender pour la modélisation 3D. Mais également la compréhension de la logique autour des animations. La multijoueur a aussi été une nouveauté avec la gestion d'un client et d'un serveur.

Pour résumer, ce projet de groupe a été pour ma part bénéfique et remplit de nouvelles compétences qui me seront utiles dans le futur.

5.4 Thomas

J'ai adoré le fait d'avoir pu mettre à profit ma créativité pour créer une carte qui plaise aux autres membres et qui soit intéressante tant visuellement que lorsqu'on se déplace dedans. Elle correspond aux attentes que nous avions et ça a été un gros défis, car je ne connaissais pas l'utilisation d'Unity avant ce projet. Pouvoir créer et accorder le scénario de l'histoire avec les autres membres a aussi été très appréciable, car cela demandait une certaine subtilité de manier idées et réalisation possible.

Le travail de groupe fourni a été génial, car la communication a toujours été présente et nous nous aidions mutuellement. J'ai désormais une meilleure connaissance des outils tels que Unity, LaTeXet git qui parait tout de suite plus logique lorsqu'on doit résoudre des conflits à longueur de journée.

6 Bibliographie

6.1 Assets utilisés

Pour notre projet, nous avons utilisé des assets et modèles 3D gratuits que nous avons par la suite mélangé pour créer notre jeu. La majorité des assets vient de l'Unity Store pour le décor et de Mixamo pour les personnages et animations.

- "Mirror" par vis2k (Standard Unity Asset Store EULA)
- "Flooded Grounds" par Sandro T (Standard Unity Asset Store EULA)
- "Mine Pack" par Gregory Seguru (Standard Unity Asset Store EULA)
- Le personnage "Remy" sur le site Mixamo
- Le personnage "Megan" sur le site Mixamo
- Toutes les animations proviennent de Mixamo ou de l'Unity Asset Store

6.2 Tutoriels et documentations

- Guide Projet de S2 : Guide de survie - Benoît M. et Luca C
- Documentation dotnet - Microsoft
- Documentation officielle de Unity
- Documentation officielle de Mirror
- Série de vidéos "Créer un FPS MULTIJOUEUR avec Mirror sur Unity" - TUTO UNITY FR
- Vidéo Unity : Utiliser le NavMesh (Pathfinding / IA) - TUTO UNITY FR
- Vidéo Settings Menu in UNITY ! 2021 Tutorial - GDTitans
- Vidéo Unity : Télécharger des centaines de personnages et d'animations gratuitement (Mixamo et Unity) - TUTO UNITY FR

6.3 Outils utilisés

- GitLab
- Discord
- Inkscape
- Blender
- Unity
- Overleaf
- Rider
- VS Code
- Figma

Annexes

A Histoire

En 2035, Tom Voy et Jack Nich, deux amis et collègues journalistes américains, sont missionnés pour écrire un nouvel article, l'article d'une vie selon leur chef. Il s'agit d'enquêter et de faire un reportage sur le *MS 402*, un très ancien manuscrit dit indéchiffrable. Pour ce faire, ils se rendent à son lieu de conservation, dans le Connecticut, à la bibliothèque de Beinecke.

Durant deux longues années, ils vont travailler sans relâche pour leur reportage, questionnant de nombreuses personnes, et ce, jusqu'au 1er juin 2037.

Ce lundi-là, leurs recherches ont subitement pris fin. La bibliothécaire leur apprit que des hommes du gouvernement, accompagnés du directeur de l'université, étaient venus chercher le livre. Leur étonnement fut de courte durée, car ils reçurent à cet instant un appel de leur chef leur disant que pour des raisons économiques, il n'était en fin de compte plus nécessaire de produire leur reportage.

Deux ans, deux longues années pour un travail qui n'aboutira finalement à rien, c'est difficile pour les deux amis. Tom cependant ne l'entendait pas de cette manière. Pour lui, le manuscrit avait tourné à l'obsession et cette coïncidence n'en était pas une. Après ces années de labeur, il tenait enfin quelque chose. Le manuscrit allait lui parler et il se doutait que ce n'était pas rien. Ils avaient bien avancé avec Jack dans l'élaboration de leur reportage certes, mais le déchiffrer n'était pour autant pas leur travail, cela relevait uniquement de la curiosité de Tom.

Suite à ces évènements, Jack et Tom retournèrent chez eux, emportant leur immense déception. Mais à peine arrivé chez lui, Tom reparti secrètement à Beinecke pour questionner ses amis de l'université, bien qu'aucun ne put lui apporter de réponses, même le Professeur Cabon, qui avait voué quinze années de sa vie au manuscrit, était dans l'incompréhension. Il allait donc devoir faire cavalier seul.

Presque trois ans plus tard, Jack reçut un courrier inhabituel dans sa boîte aux lettres, c'était Tom.

"Salut Jack, ça fait longtemps. Tout d'abord, sache que je suis désolé. Ma disparition, c'était pour votre sécurité, crois-moi. Ne fais confiance à personne en dehors d'Ellie, ma grande sœur. Tout est dangereux dehors, Jack. Je t'envoie cette lettre, car tout se résume en cinq caractères mon ami : *MS 402*. Oui, je sais, c'était censé être de l'histoire ancienne, mais j'ai continué mes recherches après la fin subite de notre reportage.

Je savais qu'il avait quelque chose à nous dire, j'en étais certain et vois-tu, je ne m'étais pas trompé. Je vais bientôt en avoir la confirmation et cela va changer la face du monde telle que nous la connaissons. Sauf que le temps me manque, ils sont à ma poursuite, ils ne veulent pas que j'y arrive. Je suis aux alentours de l'ancienne maison de Kelley et je crois qu'ils sont là, qu'IL est là. Il s'agit certainement de mon dernier contact avec toi. Désolé mon vieil ami. J'avais besoin d'écrire cette lettre et tu étais le mieux placé pour la comprendre. Prends soin de toi et veille sur ma sœur s'il te plaît. Ton ami, Tom."

B Aperçu site Web