

Collège Madame-de-Staël

Travail de Maturité
Réalisation artistique

2014-2015

Par:

Yves Zumbach

Accompagné par:
Mme Marie-France Purro

Un jeu vidéo,
de l'esprit à l'écran !

Remerciements

Je tiens à remercier pour leur aide précieuse durant la réalisation de ce travail :

Mme Purro, ma professeure accompagnante, qui aura accepté de me suivre malgré un sujet qui, du premier abord, lui paraissait complètement étranger.

Ma famille, pour leur support inébranlable et leurs conseils nombreux et pertinent – notamment concernant les points sur lesquels j'aurais dû me concentrer, ce que je n'ai pas fait et ce qui n'était pas très malin.

Mathilde, pour m'avoir écouté débiter des monologues surement interminables à propos de ce projet et pour ton avis éclairé sur mes multiples problèmes.

Les communautés web et forums sans lequel ce travail n'aurait jamais été possible.

Typographie de ce document

Les mots suivis d'une étoile* sont des termes, souvent relatifs au domaine du jeu vidéo ou de l'informatique, qui peuvent ne pas faire partie du vocabulaire courant. Ils sont définis dans l'annexe A « [Glossaire](#) » afin de lever toute incertitude.

Blocks bleus

Les paragraphes typographiés de cette façon indique les informations utiles.

Blocs jaunes

C'est cette mise en page qui indiquera les informations importantes.

Blocs rouges

Les informations ainsi présentées sont critiques, absolument nécessaires pour ce travail.

Ce document est typographié avec .

Table des matières

6	Chapitre 1 Introduction
10	Chapitre 2 Création d'un univers
19	Chapitre 3 Gameplay, ou la façon de jouer
23	Chapitre 4 Scénario et level design
29	Chapitre 5 Réalisation technique
43	Chapitre 6 Pour conclure
46	Chapitre A Glossaire
49	Chapitre B Steampunk

Chapitre

1

Introduction



Sommaire

- 1.1 Objectifs du TM, 7
- 1.2 Motivations personnelles, 7
- 1.3 Utilisation de logiciels, 7
 - 1.3.1 Les logiciels, libres ou propriétaires ?, 7
 - 1.3.2 Quels logiciels utiliser ?, 8
- 1.4 Les jeux vidéo, 8
 - 1.4.1 Définition, 8
 - 1.4.2 Les jeux vidéo au fil des années, 9

1.1 Objectifs du TM

Durant ce Travail de Maturité, je tenterai de réaliser un jeu vidéo – ou plutôt une version de démonstration (abrégée « demo »*) – en 3 dimensions. Mon objectif sera de comprendre ce qui fait un bon jeu vidéo : graphisme, story-telling*, scénario, musique. Il me faudra trouver les outils nécessaires (éditeur d'image, éditeur 3D, moteur de jeu, langages de programmation) et passer à la réalisation pratique.

Mais plus que réaliser un jeu vidéo parmi déjà tant d'autres, je m'efforcerai de conférer un sens à l'histoire, d'ajouter des critiques et des pensées sur notre monde. En effet, pourquoi le livre serait-il le moyen d'expression le plus utilisé quand un jeu vidéo pourrait faire de même, l'interactivité – pour ne pas dire le fun – en plus ? Certains titres déjà sortis comme *Mortal Combat* ou le plus moderne *Call of Duty* ont certainement contribué à la création d'une réputation négative qui aura empêché que des jeux plus philosophiques ou critiques apparaissent. L'ajout d'une nouvelle dimension plus intellectuelle à ce média sera donc un objectif principal de ce travail.

1.2 Motivations personnelles

Le monde des jeux vidéo m'a toujours passionné, il est, pour moi, une brèche dans la vie de tous les jours, un moyen de s'échapper de notre quotidien, de vivre des histoires extraordinaires dans des univers fantastiques. La diversité des jeux est telle, que chacun peut y trouver son bonheur ; ils font travailler notre imagination, testent nos réflexes, nous détendent et nous font réfléchir. Et s'ils sont des divertissements très efficaces, ils sont aussi, bien que cela ait été relégué au deuxième plan jusqu'ici, des moyens d'expression redoutables.

Par ailleurs, depuis ma première année au collège, je me suis intéressé à l'informatique. J'ai ainsi touché à la robotique, la programmation, la modélisation 3D, la retouche photo, aux réseaux et à d'autres domaines techniques.

Ces deux éléments m'ont donné l'envie et la possibilité de me lancer dans la réalisation d'un jeu vidéo. Je me suis documenté sur ce genre de projet : quels outils utiliser ? quel langage de programmation employer ? en 2 ou 3 dimensions ? Mais l'ampleur et la complexité d'une telle création m'ont rapidement arrêté et j'ai abandonné l'idée pour me concentrer sur d'autres programmes plus réalisables... jusqu'au moment de définir le sujet de mon TM. Il me fallait un projet suffisamment vaste et complexe pour m'intéresser durant une année. Quel meilleur choix que la réalisation d'un jeu vidéo !

1.3 Utilisation de logiciels

A mieux introduire

1.3.1 Les logiciels, libres ou propriétaires ?

Un logiciel libre désigne un programme dont le code source est accessible. Cela permet à la communauté entourant ce programme d'apporter des améliorations et de le partager légalement avec le reste du monde. Ces programmes tendent à être gratuits mais ne le sont pas forcément. On parle aussi de logiciels open source ; ce terme désigne également des logiciels libres, mais si la désignation « libre » s'intéresse à l'aspect philosophique et politique, « open source » se rapporte à la partie technique ainsi qu'à la diffusion du programme. Parmi les logiciels libres, on pourra trouver : Linux,

Firefox, Thunderbird, Gimp, Blender et LibreOffice par exemple.[\[1\]](#)[\[2\]](#)

L'opposé des logiciels libres sont les logiciels propriétaires ; son code source n'est pas accessible. Ils représentent la plus grande partie des programmes. Citons par exemple : Microsoft Windows, la suite Microsoft Office, la suite Adobe à laquelle appartient Photoshop et tous les logiciels Apple. Certains d'entre eux peuvent être gratuits mais c'est l'inverse qui se vérifie le plus souvent.

1.3.2 Quels logiciels utiliser ?

La réalisation d'un jeu vidéo se fait normalement sur plusieurs années, avec une équipe de professionnels et des budgets importants – pour ne pas dire colossaux, suivant les titres. Mes ressources sont évidemment plus restreintes : je suis seul, dispose de moins d'une année et surtout, mon budget est limité.

De plus, le logiciel libre est, pour moi, une des révolutions les plus importantes et bénéfiques de notre temps. C'est un modèle économique complètement différent, basé sur l'entraide et, le plus souvent, la gratuité. C'est une autre façon, un peu à la manière d'Internet, de rendre la connaissance gratuite et disponible pour tous.

Pour toutes ces raisons, je n'emploierai que des logiciels gratuits et tenterai d'utiliser le plus possible des programmes libres pour soutenir ce mouvement.

1.4 Les jeux vidéo

1.4.1 Définition

Même si tout le monde a au moins une petite idée de ce qu'est un jeu vidéo, il me paraît important de rappeler la définition que Wikipédia donne à ce sujet et de la compléter. Selon l'encyclopédie en ligne :

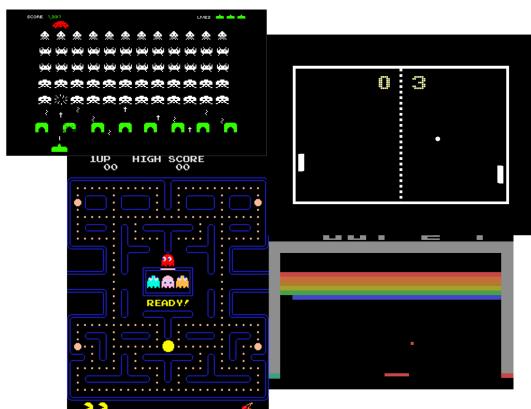
« Un jeu vidéo est un jeu électronique qui implique une interaction humaine avec une interface utilisateur dans le but de générer un retour visuel sur un dispositif vidéo. Le joueur de jeu vidéo dispose de périphériques pour agir sur le jeu et percevoir les conséquences de ses actes sur l'environnement virtuel. Le mot « vidéo » dans le jeu vidéo, fait traditionnellement référence à un dispositif d'affichage de trame, mais à la suite de la vulgarisation du terme, il implique désormais tout type de dispositif d'affichage. »[\[3\]](#)

Cette définition, très complète du point de vue technique, fait cependant abstraction de la partie humaine des jeux vidéo. Ils sont bien, d'un point de vue purement logique, des interactions homme-machine. Mais ils sont aussi, et pour moi avant tout, une façon de se divertir, de découvrir, d'imaginer et même de partager (jeux multijoueurs). Pour qu'un jeu ait du succès, il doit faire ressentir aux joueurs des émotions, il doit communiquer une histoire, stimuler notre réactivité, nos capacités de réflexion et de logique. Un jeu vidéo, c'est aussi l'occasion de rêver, de vivre des aventures dans d'autres mondes, tout comme un livre, à cette différence près que la narration y est interactive.

Au cours de ce travail, j'aborderai ces différents aspects : la partie humaine sera approfondie dans les premiers chapitres avec la création du monde, le scénario et le gameplay ; la partie technique le sera dans les derniers chapitres avec la création des niveaux, la réalisation pratique et, finalement, la diffusion du jeu.

1.4.2 Les jeux vidéo au fil des années

1970



1980

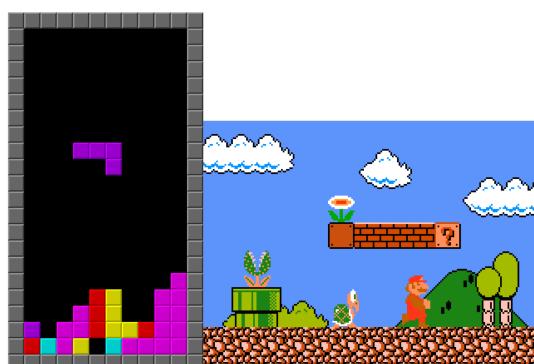


Fig. 1.2 : Tetris et Super Mario Bros

Fig. 1.1 : Space Invaders, Pong, Pacman et Breakout

1990



Fig. 1.3 : Street Fighter II, Doom, Myst et Final Fantasy VI

2000



Fig. 1.4 : Zelda, Call of Duty, Sonic et Angry Birds

Chapitre

2

Création d'un univers



Sommaire

- 2.1 Les thèmes importants, 11
- 2.2 Un monde divisé entre deux peuples, 11
- 2.3 Les Humains, 11
 - 2.3.1 L'histoire d'un peuple, 12
 - 2.3.2 La ville, 12
 - 2.3.3 La religion humaine, 12
 - 2.3.4 Propagande gaamoniste, 13
- 2.4 Les Telurans, 13
 - 2.4.1 Leur histoire, 14
 - 2.5 Énergies, 15
 - 2.5.1 Humains, 15
 - 2.5.2 Telurans, 15
 - 2.6 Bestiaire, 16
 - 2.6.1 Créatures humaines, 16
 - 2.6.2 Personnalisation de la Nature, 17

Dans cette partie, j'expliquerai les détails de l'univers du jeu mais aussi les raisons qui m'ont poussé à faire les choix qui y ont mené. Il est à noter que l'univers n'est pas terminé ; s'il suffit largement aux besoins du jeu, il pourrait, dans l'optique de la réalisation d'une version complète, être agrandi et complété. De plus, certains détails, exprimés dans ce texte, dépassent déjà largement la portée de la démo.

2.1 Les thèmes importants

Comme je l'ai déjà indiqué dans la section [1.1](#), ce jeu est l'occasion pour moi de développer des thématiques qui me porte à cœur. C'est un jeu d'opinion qui se veut le défenseur de certains points de vue, les miens. Le jeu évolue donc autour de quelques thématiques clés, chères à mes yeux.

Le point le plus important est la relation humaine à la Nature. J'ai toujours été fasciné par sa beauté, sa cruauté, à la fois son ordre parfait et son désordre total. Pour moi, un retour à la Nature est une condition nécessaire pour atteindre le bonheur. La relation que nous entretenons avec elle, collaborative ou dominante, sera donc au cœur du jeu et ce choix déterminera de nombreux aspects de l'histoire et du gameplay*.

Il existe cependant de multiples contradictions entre ma passion pour la Nature et l'univers qui m'en-toure. Comment ne pas constater les dégâts que nous infligeons à notre environnement ? Mais plus encore, comment ne pas constater que notre mode de vie (que certains nomment consumérisme) a perdu toute sa simplicité, son sens, sa « naturalité » ! Ce jeu sera donc l'occasion de mettre le doigt sur ces paradoxes et l'opportunité de se poser certaines questions sur nos façons de faire.

Un autre thème conducteur sera la relation fraternelle (sororale dans le cas du jeu) ; sujet important à mes yeux, étant donné que j'ai moi-même un frère. Le jeu gravitera autour de la question de l'utilisation bénéfique et responsable ou destructrice et irresponsable de la technologie, des améliorations qu'elle peut apporter ou des dommages qu'elle peut causer et des abus auxquels elle peut conduire. Finalement, la question de la religion sera abordée, bien que dans une moindre mesure : quelle utilité a-t-elle ? à quelles dérives peut-elle mener ?

2.2 Un monde divisé entre deux peuples

L'univers dans lequel se déroule le jeu, nommé Éluria, est divisé entre deux peuples : les Humains et les Telurans.

2.3 Les Humains

Les Humains sont une faction terne, grise et soumise à un tyran : Lord Gaamon. Le despote, atteint de folie, a une peur viscérale de la Nature et a convaincu le reste de la population de sa nocivité. Ce peuple perdu et sombrant dans la peur a, pour sa plus grande majorité, oublié son histoire et exécute avec soumission les ordres de son maître. L'utilisation de technologies mécaniques, polluantes et brutales est monnaie courante et permet à cette faction de dominer les autres ; mais cela ne peut se faire sans ravager l'environnement.

Ce peuple vit enfermé dans Murtos, une ville énorme, entourée de hautes murailles. Personne ne sort de ses frontières, de peur d'être contaminé par la Nature. Les Humains représentent un peuple perverti, brutal, apeuré et aveugle.

2.3.1 L'histoire d'un peuple

Quelques décennies avant le début du jeu, la ville était encore un petit village agréable. La vie agricole, bien que rude et laborieuse, y était appréciée et rares étaient les histoires qui venaient troubler la quiétude générale. Mais tout cela changea le jour où le jeune Gaamon arriva...

L'étranger, apparemment venu de contrées lointaines, décida de s'installer dans le bourg et fonda la première usine d'Eluria. Son succès fut rapide : il apportait du travail et surtout produisait beaucoup grâce aux machines qu'il construisait. Il agrandit bientôt son usine... puis en construisit une deuxième. Grâce à cela, les conditions de vie s'améliorèrent, tout le monde avait du travail et la population était heureuse de cette venue providentielle.



Fig. 2.1 : Symbole de Gaamon

Cela aurait pu rester ainsi pour de longues années, mais l'ambition de Gaamon était grande. Il fit construire d'autres usines, créa d'autres machines, employa bientôt tout le village et draina les populations des alentours dans ses halles de production. Les affaires allaient bon train, l'argent arrivait en quantité et son influence ne cessait de s'étendre.

Avec les années, il fédéra tous les Humains sous la bannière unique de ses usines et devint le maître incontesté de ce qui était devenu une ville. Mais Gaamon n'était pas complètement sain d'esprit ; cachée au fond de lui résidait une folie, une phobie haineuse de la Nature. Le pouvoir et l'argent permirent à cette dernière, jusqu'alors refoulée, de s'exprimer de plus en plus pleinement et ouvertement. Il avait réussi à industrialiser, mécaniser le village, il voulait maintenant étendre son emprise hors des limites de la cité.

Il fortifia la ville en construisant des murailles et bâtit, en son centre, une immense tour d'acier et de verre qui devint l'icône emblématique de son empire. Il imposa à tous l'interdiction de sortir de l'enceinte des murs et brûla les dernières traces de verdure dans la cité. Il s'engagea alors, depuis le haut de sa nouvelle forteresse, dans un projet d'éradication plus vaste, qui annihileraient toute Nature, plantes ou animaux, des terres environnant les fortifications.

Le jeu débute quelques années après le commencement de ce funeste projet, à l'intérieur de la ville.

2.3.2 La ville

La ville est le bastion de la race humaine, l'empire de Lord Gaamon. Son décor relève de l'univers steampunk (voir annexe B). Il s'agit d'un monde urbain de l'ère industrielle où l'horizon est rythmé par les cheminées des hauts fourneaux qui crachent leur fumée nauséabonde et où la suie recouvre les architectures de métal et de verre. Cet environnement, métaphoriquement, représente l'utilisation abusive de la technologie. C'est aussi la haine envers la Nature, sa négation totale. Un plan de Murtos est donné à la figure 2.2.

2.3.3 La religion humaine

La religion humaine est, à l'image de la ville, corrompue. Elle prône le bonheur par l'aisance matérielle, la richesse et la consommation. C'est même une apologie de la consommation – si possible inutile – que soutient ce dogme. De plus, comme les prêtres professent que l'argent ne s'acquiert que par le travail, Gaamon utilise la religion pour asservir les masses et motiver un travail acharné dans ses usines. Le symbole de ce culte possède une forme très caractéristique (voir figure 2.3a) qui s'emboite avec l'icône de Gaamon pour représenter la collusion Religion-État. Quelques images de propagande que l'on peut trouver placardées sur les murs de la ville sont données par la figure 2.3.

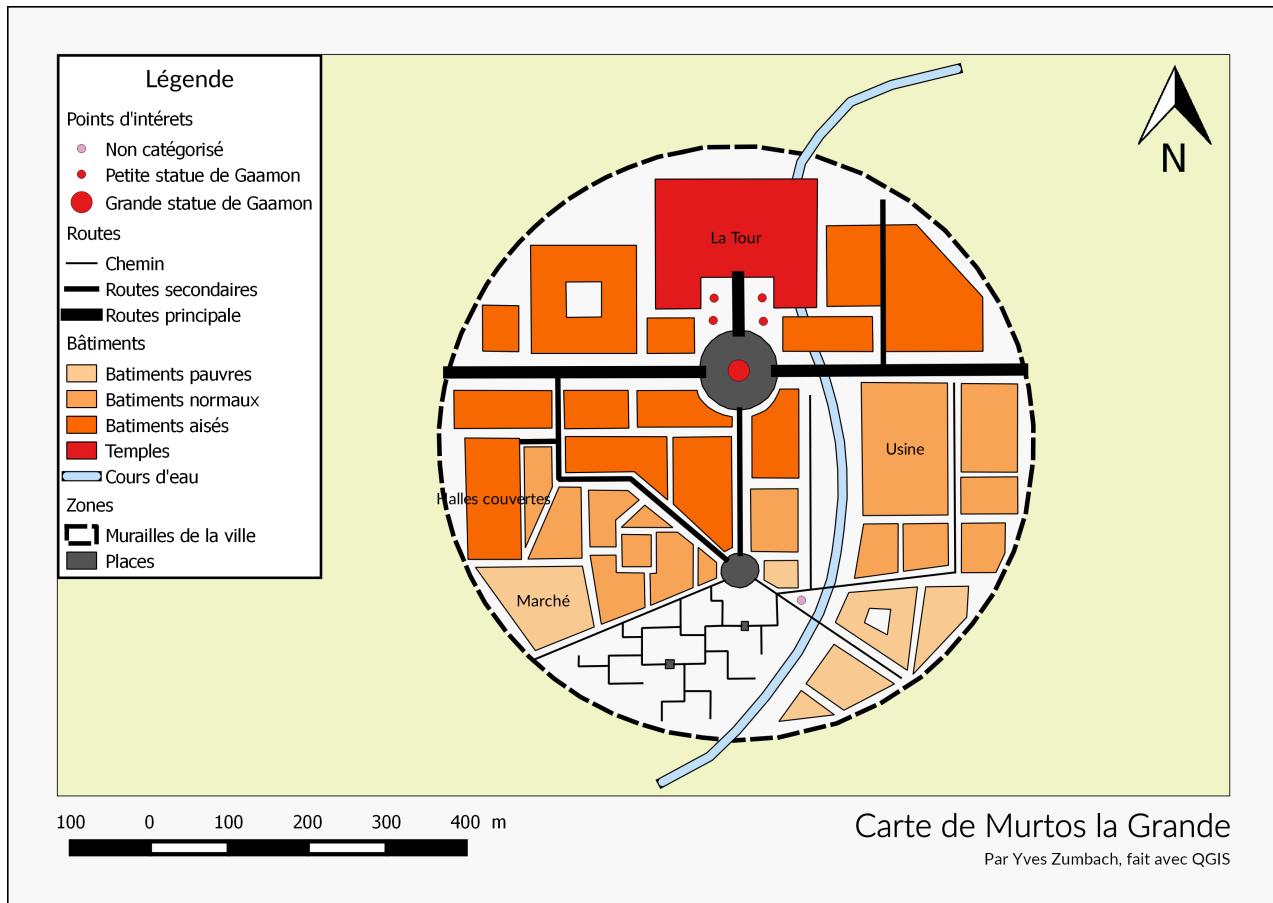


Fig. 2.2 : Plan de la ville actuelle (Réalisé à l'aide du programme Quantum GIS)

2.3.4 Propagande gaamoniste

Gaamon utilise également les affiches, parmi d'autres vecteurs, pour répandre ses idées. Un aperçu de ces affiches est donné à la figure 2.4. Les images 2.4a et 2.4b répandent les volontés « anti-naturelles » étatiques, quand à elles, les figures 2.4c et 2.4d défendent l'industrialisation massive du peuple humain pour la puissance qu'un tel apport fourni.

2.4 Les Telurans

Le nom « Telurans » est une anagramme de « naturels » et pourrait se rapporter à l'adjectif « telurique » qui signifie « de la terre ». Les Telurans, à l'inverse des Hommes, représentent un peuple pacifique. Ils accordent beaucoup d'importance à la Nature et tentent de vivre en accord avec elle.

Ils sont une espèce différentes des humains. Les humains croyant en la Nature sont, quant à eux, appelés « Naturels ». Se faire traiter de naturel dans Murtos est une des pires insultes qui soit.

2.4.1 Leur histoire

Les Telurans étaient, par le passé, le peuple le plus brillant et le plus avancé d'Eluria. Ils avaient trouvé une source d'énergie formidable, l'énergie verte, qui leur permettait de vivre heureux et prospères. L'extraction de cette ressource naturelle était cependant un processus compliqué que seuls les shamans de l'énergie maîtrisaient (voir section 2.5). Grâce à cette source, leurs cités étaient magnifiques, leurs manières connues pour leur raffinement et leur art reconnu pour sa délicatesse.

Ce peuple jouissait d'une prospérité jusque-là sans égale dans l'histoire et répandait sa sagesse, ses connaissances et son amour de la nature de par le monde. L'ascension au pouvoir de Gaamon marqua la fin de cette période ; afin d'empêcher la propagation d'un idéal qui mettait la Nature au centre des préoccupations, il fit assassiner les prêtres de l'énergie. Les Telurans se retrouvèrent subitement privés de leur source vitale – l'énergie verte – sans protection et virent leur civilisation s'effondrer brutalement.

Les Telurans vivent maintenant reclus sur de hauts plateaux, cachés de la haine ainsi que des persécutions de Gaamon et rêvent de leur gloire passée. Si leur idéologie d'amour envers la Nature perdure, cette civilisation décline de jour en jour.



(a) Accède au Bonheur!



(b) Consommer pour la santé



(c) La Richesse c'est le Bonheur!

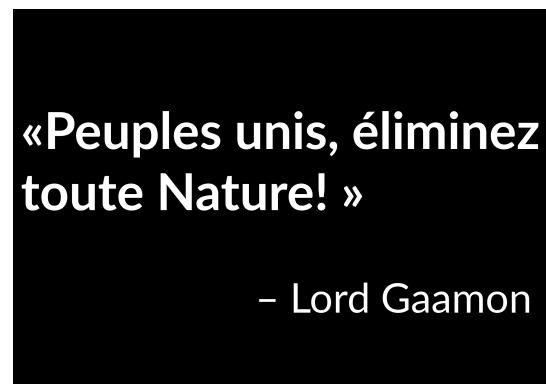


(d) L'aisance matérielle par le travail

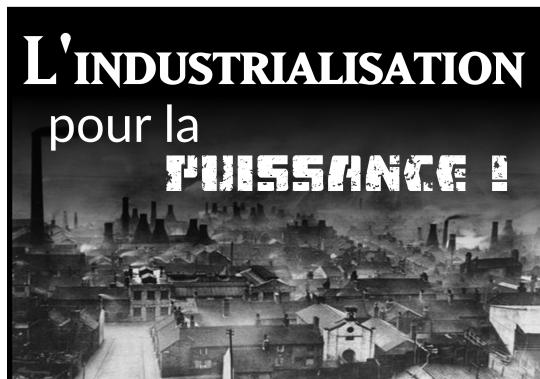
Fig. 2.3 : Propagande religieuse



(a) Gaamon pour vous sauver de la Nature



(b) Éliminez toute Nature



(c) L'industrialisation pour la Puissance



(d) Électricité

Fig. 2.4 : Propagande étatique

2.5 Énergies

Les énergies sont un élément clé de ce monde, elles déterminent quels peuples survivront et quels seront ceux qui disparaîtront. C'est un parallèle de notre dépendance à l'énergie. En effet, on ne peut plus imaginer aujourd'hui vivre sans électricité. À Éluria, les sources d'énergies sont diverses et représentatives des peuples.

2.5.1 Humains

Chez les Humains, l'énergie est tirée du charbon pour créer de la vapeur et de l'électricité. C'est une source très polluante. La couleur qui lui est attribuée est le rouge du feu des hauts fourneaux et le noir de la fumée nauséabonde qui remplit le ciel de Murtos. Elle fait partie intégrante du Steampunk, caractéristique de ce peuple.

2.5.2 Telurans

À l'inverse, les Telurans ont exploité l'énergie des arbres, appelée énergie verte. Les plantes fournissaient un apport énergétique et, en contrepartie, les Telurans portaient une attention toute particulière à leur santé, reproduction, protection et longévité. Une symbiose existait entre les Telurans

et la Nature, un équilibre grandement profitable autant pour l'un que l'autre des partis. L'énergie verte était extraite dans la grande salle, dont l'accès était réservé aux grands shamans de l'énergie. Ils étaient les seuls à pouvoir en maîtriser le fonctionnement complexe. Si ce secret était aussi bien gardé, c'est que cette énergie très puissante, extraite en trop grandes quantités, aurait pu causer la mort de milliers d'arbres et animaux. Un tel cas de figure était à craindre car la Nature se serait retournée contre les personnes ayant abusé de ses ressources et, dans sa fureur, aurait annihilé les profiteurs. Les Telurans, craignant une pareille situation, s'en étaient prémunis en cachant les mécanismes de la grande salle pour ne les révéler qu'aux seuls élus, les shamans.

Cette énergie pouvait également être maîtrisée, dans une moindre mesure, à titre personnel. Les personnes dotées de telles capacités pouvaient alors lancer des « sorts ». Ceci est un des éléments clé du gameplay (voir section 3.4). On pourrait comparer l'énergie verte à de la magie mais cela ne serait pas très judicieux : l'énergie verte n'est ni illimitée, ni gratuite. L'utiliser de façon disproportionnée signifie tuer toute vie alentour. Et si la Nature est généreuse elle peut aussi refuser de donner son bien, voire punir ceux qui en abusent. Il est également possible de l'extraire de soi-même, mais une fois de plus, la réalisation d'une telle opération est dangereuse car elle peut causer la mort de l'exécutant.

Depuis la disparition mystérieuse des shamans (voir section 2.4.1), les Telurans ont perdu le secret de l'énergie verte. Ils utilisent donc les énergies hydraulique et éolienne, facilement disponibles dans les montagnes où ils vivent. Elles sont cependant bien moins puissantes que l'énergie verte et ne leur permettent de loin pas les miracles qu'ils pouvaient accomplir avec la première.

2.6 Bestiaire

L'univers d'Eluria est peuplé de créatures. Du point de vue des genres, ce monde est un croisement entre la science-fiction et la Fantasy.

2.6.1 Créatures humaines

Les sbires de Gaamon sont des créatures nommées *Gnobols*. Il existe beaucoup de variantes de la créature « de base », mais ils sont tous équipés d'un chapeau duquel une fumée noire s'échappe .Certains, cependant, auront une armure quand d'autres seront munis d'un arc ou d'un fusil.



Fig. 2.5 : Gnobil

2.6.2 Personnalisation de la Nature

Dans *Eluria's Chronicles*, la Nature est personnalisée par trois esprits, chacun représentatif d'un aspect de la Nature qui définit leur apparence :

Esprit	Description	Mots-clé	Apparence
Leo	Il est la destruction renouvelatrice, à la fois créateur et destructeur	Massif, sans distinction, abrupte, rapide, mort génératrice de vie	Golem, de pierre et de lave en bas, de pierre et d'eau en haut, avec un duvet de plante sur les épaules
Tamund	Il est l'intelligence aveugle, l'évolution lente	Évolution, lent, ciblé, mort des plus faibles	Grand séquoia aveugle
Tia	Elle est la vivacité sauvage et profite parfois sans pitié	Sauvage, vif, rapide, profiteur, sans-pitié, survie du plus adapté et mort des plus faibles	Écureuil avec des runes sur le dos

Tab. 2.1 : Les trois esprits et leurs caractéristiques



Fig. 2.6 : Une esquisse de Leo, basée sur le travail original de Sinto-risky [9]

Chapitre

3

Gameplay, ou la façon de jouer



Sommaire

- 3.1 Idée générale, 20
- 3.2 Mode de jeu *stealth*, 20
- 3.3 Les ressources, 20
- 3.4 Les sorts, 21
- 3.5 Les objets, 21
 - 3.5.1 Bâton de bois, 21
 - 3.5.2 Sarbacane, 22



Une question de vocabulaire

Ce chapitre porte sur un sujet relativement technique et récent. L'univers du jeu vidéo étant principalement anglophone, le vocabulaire de ce domaine est composé de beaucoup d'anglicismes, tel *gameplay*. La langue française n'a pas encore eu le temps de s'adapter. Pour cette raison, les anglicismes seront conservés et typographiés de cette façon.

3.1 Idée générale

Ce jeu tente de promouvoir la paix et la résolution de problème par des solutions non-violentes. De ce parti pris découlent bien des aspects. Ainsi, le joueur sera « puni » s'il utilise ses armes mais il est à noter qu'elles restent à disposition et peuvent être améliorées, tout comme d'autres items. C'est aussi une question qui est posée au joueur : quelle voie préfère-t-il utiliser ? Pacifique ou brutale ? Libre à lui de choisir mais le jeu prend parti pour celle qui n'utilise pas la violence et répercute cela sur le score et certaines ressources.

3.2 Mode de jeu stealth

Stealth, en français, peut se traduire par furtif, discret, ruse ou encore dissimulation. Dans les milieux francophones, on parlera de jeu d'infiltration à la place de *stealth game*, cependant le sens n'est pas exactement le même. Ce terme, sans bonne traduction, représente bien le style de gameplay que l'on peut rencontrer dans *Eluria's Chronicles*. Le jeu défendant une approche pacifique, il est bien sûr hors de question de faire tirer le joueur sur ses ennemis ou de lui permettre de les démolir à coups de poings ou d'épée.

À la place, c'est un mode de jeu où il faut éviter les opposants, se cacher et rester discret, qui est privilégié. Un autre aspect développé est la résolution de problèmes, ainsi que la complétion de sous-quêtes. Finalement, on notera l'utilisation relativement répandue de puzzles*, qui nécessiteront de collecter des objets au fil des aventures pour pouvoir être résolus (*item collecting*).

3.3 Les ressources

Un élément important dans les jeux vidéos est représenté par les ressources. L'argent, l'expérience (souvent abrégée XP), les munitions ou encore le mana (représente une quantité de magie) sont des exemples de ressources. Elles forment le « système économique ». Un jeu où les ressources sont disponibles en quantité suffisante mais pas en excès, ce qui rendrait le jeu facile, ou en manque, ce qui le rendrait trop difficile, peut faire toute la différence entre un bon et un mauvais titre.[4]

Dans *Eluria's Chronicles*, les ressources principales sont les suivantes :

- Argent
- Énergie verte
- Coefficient de naturalité (score)

L'énergie verte est une ressource un peu spéciale. Elle est décrite dans la section [2.5 : Énergies](#) du point de vue de l'univers. Dans le jeu, elle est principalement utilisée pour lancer des sorts.

Les succès* font aussi partie du système économique mais pas au même titre que les ressources précédemment citées. En effet, on ne peut que gagner ou débloquer un succès, à l'accomplissement

d'objectifs supplémentaires par exemple ; impossible de les utiliser ou de les vendre ensuite. Ils font cependant partie du système de récompense du joueur.

Les graines sont un autre type de ressource, un peu particulier cette fois. Ces dernières sont rares et dispersées dans le jeu. Lorsque le joueur en découvre ou gagne une, il peut la planter. Il doit ensuite l'arroser et la nourrir. Plus il entretient la plante, plus elle grandit vite. Une fois adulte, la plante donne des fruits que le joueur peut cueillir, ce sont des bonus de vie, énergie verte, ingrédients, etc. La quantité de fruit dépend également de l'attention apportée à la plante. Les plantes sont accessibles dans un menu séparé en 2D.

Ressource	Comment en gagner	Comment en perdre
Argent	Compléter des quêtes, Dans les coffres	Achats
Énergie verte	Planter une graine, Se régénère avec le temps	Utiliser la violence, Lancer un sort
Coefficient de naturalité	Faire preuve de naturalité, Compléter des sous-quêtes	Utiliser la violence

Tab. 3.1 : La gestion des ressources

3.4 Les sorts

Les sorts sont les principaux outils mis à la disposition du joueur. Il en existe 9 majeurs, soit 3 par esprit. Seuls ces derniers peuvent offrir les totems qui permettent de les contrôler. Les sorts majeurs ne s'obtiennent donc qu'àuprès des esprits. Ces sorts sont coûteux en énergie verte mais nécessaires pour avancer dans l'histoire.

Les sorts mineurs en revanche ne nécessitent pas de totem. On peut les découvrir un peu partout dans l'univers et ils nécessitent nettement moins d'énergie verte pour pouvoir être lancé.

3.5 Les objets

Les objets sont un élément important du jeu dans le sens qu'ils permettent au joueur d'interagir avec l'environnement virtuel. Gagner un nouvel objet sera souvent la clé pour terminer un niveau car il permettra d'accéder à de nouveaux endroits, de débloquer certains passages, etc. Le joueur pourra s'équiper de deux objets à la fois et passer très rapidement de l'un à l'autre à l'aide de touches de raccourcis. Pour changer les objets dont il sera équipé, il devra passer par le menu.

3.5.1 Bâton de bois

! Objet le plus important

Le bâton de bois est sans conteste l'objet le plus utile et le plus important. Il est le catalyseur nécessaire pour lancer des sorts. Ces pouvoirs sont sans aucun doute le moyen d'interagir le plus puissant conféré au joueur.

C'est sur le bâton de bois qu'il faut placer les totems représentant les sorts afin de les activer. Il pourra, de plus, être personnalisé par l'ajout de runes (leur couleur pourra aussi être spécifiée) ou par l'ajout de décos (débloquées grâce aux succès). Le bâton pourra aussi être amélioré : au début du jeu, il ne peut porter, au maximum, que trois totems, mais cette valeur peut être augmentée au fil des niveaux. C'est un item particulièrement intéressant du gameplay car il offre au joueur la possibilité de personnaliser un objet, et met en avant les succès du joueur. Ce dernier se sentira alors d'autant plus impliqué ce qui le motivera à poursuivre le jeu.

3.5.2 Sarbacane

La sarbacane est un objet qui permet de tirer au loin de petites fléchettes. Cela ajoute une nouvelle distance, plus éloignée, au gameplay. Les fléchettes et leur éventuel contenu doivent être créés par le joueur. Il doit ainsi trouver le bois nécessaire aux dards (disponible en grande quantité, là ne réside pas le problème) mais surtout les recettes et ingrédients pour fabriquer les poisons à insérer dans les projectiles. À ce moment, le joueur peut créer des mixtures mortelles, soporifiques, empoisonnées, etc. L'idée derrière cet objet, le deuxième plus important du jeu, est de permettre au joueur de choisir entre une résolution de conflit « pacifique » ou meurtrière.

Pour le joueur, les recettes de poison sont de nouvelles récompenses et les ingrédients, éparpillés dans l'univers, devront être collectionnés (*item collecting*) afin de pouvoir finalement réaliser la potion désirée.

Chapitre

4

Scénario et level design



Sommaire

4.1	Premier acte, 24
4.1.1	Redel, le quartier pauvre, 24
4.1.2	Niveau 1 – Course-poursuite avec la garde, 24
4.1.3	Niveau 2 – À la recherche de Lenaï, 25
4.1.4	Niveau 3 – Dans l'antre du démon, 26
4.2	Deuxième acte, 26
4.3	La suite, 28
4.3.1	Jouer Lenaï, 28
4.3.2	Des batailles aériennes, 28

Level design

Le *level design* est l'art de créer des niveaux bien équilibrés, fun à jouer et variés. C'est l'art de bien disposer les obstacles, récompenses et bâtiments pour que la difficulté soit adaptée. La carte doit présenter suffisamment de challenge pour motiver les joueurs, sans être excessivement difficile ce qui les empêcherait d'avoir une bonne expérience du jeu.

Niveaux déjà préparés

Seuls les premiers niveaux du jeu sont définis de manière détaillée. Le reste du jeu n'est qu'hypothétiquement décrit à la fin de ce chapitre.

4.1 Premier acte

4.1.1 Redel, le quartier pauvre

Le premier niveau se déroule dans le quartier pauvre, nommé Redel (voir les cartes [2.2](#) et [4.1](#)). Il se situe à l'extrême sud de Murtos. C'est le quartier le plus sale, où vivent les gens les plus pauvres.

Le coin sud-est du quartier est occupé par une énorme usine gouvernementale. L'accès à cette dernière est formellement interdit et des gardes surveillent en continu son entrée. De plus, elle crache sur le quartier des fumées toxiques, particulièrement malsaines à respirer.

Un « bastion vert »

Les dernières personnes à croire en la Nature habitent les maisons les plus isolées, au sud de Redel. Ces gens cachent cependant leur croyance, de peur d'être persécutés. N'ayant plus vu une plante depuis des années, ils commencent à oublier ou à sombrer dans la superstition.

Un des trois sanctuaires de la Nature, celui de Tia, est caché dans ce quartier. Pour y accéder, un passage secret existe dans la maison située juste au nord de ce jardin caché. Cette dernière est habitée par Maïnin, un vieil homme sage qui a gardé l'esprit clair et lucide.

4.1.2 Premier niveau – Course-poursuite avec la garde

Un niveau d'importance

Le premier niveau est crucial, c'est à la fois la scène d'exposition, l'opportunité de découvrir l'univers du jeu et l'apprentissage nécessaire des contrôles. S'il est bien réussi, fluide, le joueur pourra prendre plaisir à jouer et s'attacher à son personnage.

Dans le premier niveau d'*Eluria's Chronicles*, le joueur incarne Kida, l'héroïne, et est accompagnée de Lenaï, sa grande sœur. Les deux premiers instants servent à l'apprentissage des commandes : se déplacer, courir, s'accroupir, sauter, etc.

Puis viens l'élément déclencheur : deux gardes qui passent par l'avenue marchande, oublient négligemment un petit coffre au bord de l'étalage où ils viennent d'acheter de quoi manger. Kida ne peut

s'empêcher de voler le coffre. Mais au moment de commettre son délit, les gardes reviennent, ayant noté leur inadvertance. À la vue des deux sœurs, ils leur hurlent d'arrêter et la course-poursuite s'engage immédiatement.

Le joueur doit passer à temps les obstacles qui lui sont présentés afin de pouvoir s'enfuir. Le niveau ne doit présenter qu'une faible difficulté pour permettre au joueur à la fois de s'habituer aux commandes et d'avoir une première impression aussi bonne que possible.

À la fin du niveau, Kida s'enfuit avec le coffre mais Lenaï est capturée. Cet élément sera la cause des multiples péripéties qui s'ensuivront dans cette histoire.

Niveau de type *alley*

Ce niveau est dit de type *alley*. À savoir, le joueur n'est pas libre de se balader selon sa bonne volonté. À l'inverse, pour réussir le niveau, il doit suivre avec succès un chemin prédéfini.

4.1.3 Deuxième niveau – À la recherche de Lenaï

Objets à débloquer dans ce niveau

- Bâton de bois
- Trois sorts majeurs
- Tyrolienne
- Sarbacane
- Une recette de poison

Le deuxième niveau débute le jour suivant. Kida part à la recherche de sa sœur. Cette quête va cependant être entravée par le fait que la garde a bouclé les sorties du quartier pour retrouver le coffre volé ; tout le monde est fouillé. Ainsi, l'héroïne est bloquée dans Redel. Faute de mieux, elle cherche à ouvrir le coffre volé, qui semble avoir nettement plus de valeur que ce qui aurait pu être supposé dans un premier temps.

La première tâche du joueur sera d'ouvrir le coffre. Pour cela, il faudra trouver du métal, s'introduire dans la boutique du serrurier absent par les toits et utiliser ses machines pour créer une clé adaptée. Lorsque le coffre s'ouvre, un étrange animal en bondit. C'est la première rencontre avec un esprit de la Nature : Tia, l'écureuil vif et sauvage. La créature dit à Kida d'aller voir Maïnin, le vieux sage, puis s'enfuit en courant.

Le vieil homme, qui est en fait le gardien de l'accès au sanctuaire, y introduit la jeune fille (il faut résoudre un puzzle pour activer le passage). Kida rencontre Tia, qui lui explique les bases du fonctionnement de l'énergie verte, lui confie les trois premiers totems et le bâton de bois (voir les sections sur le [Gameplay, ou la façon de jouer](#)). Elle lui parle également d'un passage secret au cœur du quartier naturel. Il donne accès à la rivière, puis à la tour de Gaamon dans laquelle Lenaï est sûrement enfermée.

Pour accéder à ce quartier, il faut résoudre toute une série de quêtes et de sous-quêtes pour finalement débloquer le passage indiqué par un « ? » sur la maison mitoyenne de celle du cordonnier.

Les Naturels confieront à Kida la sarbacane, quelques dards et sa première recette de poison. Si le joueur a déjà collectionné suffisamment d'ingrédients, il peut confectionner des dards supplémentaires. Ils ouvrent ensuite le passage situé sous la statue de Gaamon, au milieu de la place. L'héroïne peut ainsi accéder à la rivière.



Niveau de type *island*

Ce niveau, à l'inverse du précédent, représente un niveau de type *island*. Cela signifie que le joueur est libre de ses mouvements et peut aller où bon lui semble. Un autre terme pour décrire ce type de niveau serait semi open-world*.

4.1.4 Troisième niveau – Dans l'antre du démon

Kida pénètre dans le château de Gaamon par les souterrains. Ce niveau est un temple*. Il s'agira donc de trouver les cachots où Lenaï pourrait être enfermée. Cela en résolvant des puzzles* pour accéder aux pièces suivantes et éviter les ennemis (se référer à la partie « [Gameplay, ou la façon de jouer](#) » pour plus de détails).

Les objectifs intermédiaires du niveau seront de trouver la carte, collecter des objets et des clés afin de pouvoir ouvrir les portes vers les salles suivantes.

Finalement, en arrivant aux cachots, Kida ne trouvera que des cellules vides... Aucune trace de Lenaï. C'est à ce moment que le boss* apparaît sous la forme d'une grande femme en armure. Le combat s'engage immédiatement. Si Kida se défend du mieux qu'elle peut, elle ne peut résister face à son adversaire parfaitement équipée pour le combat, aux armes aiguisées et techniques évoluées. Elle finit par se faire renverser et atterrit durement sur le sol, désarmée. L'antagoniste lève son arme afin de porter le coup de grâce. Mais l'espace d'un instant, un éclair de doute traverse les yeux de la guerrière. Instantanément, Kida reconnaît sa grande sœur sous le masque terrifiant qu'elle porte : c'est contre Lenaï qu'elle se bat ! Lainée ne peut se résoudre à assassiner sa petite sœur. Kida profite de l'ouverture pour s'enfuir à toute vitesse, se précipite vers la fenêtre la plus proche et saute droit dans les douves... à l'extérieur de la ville.

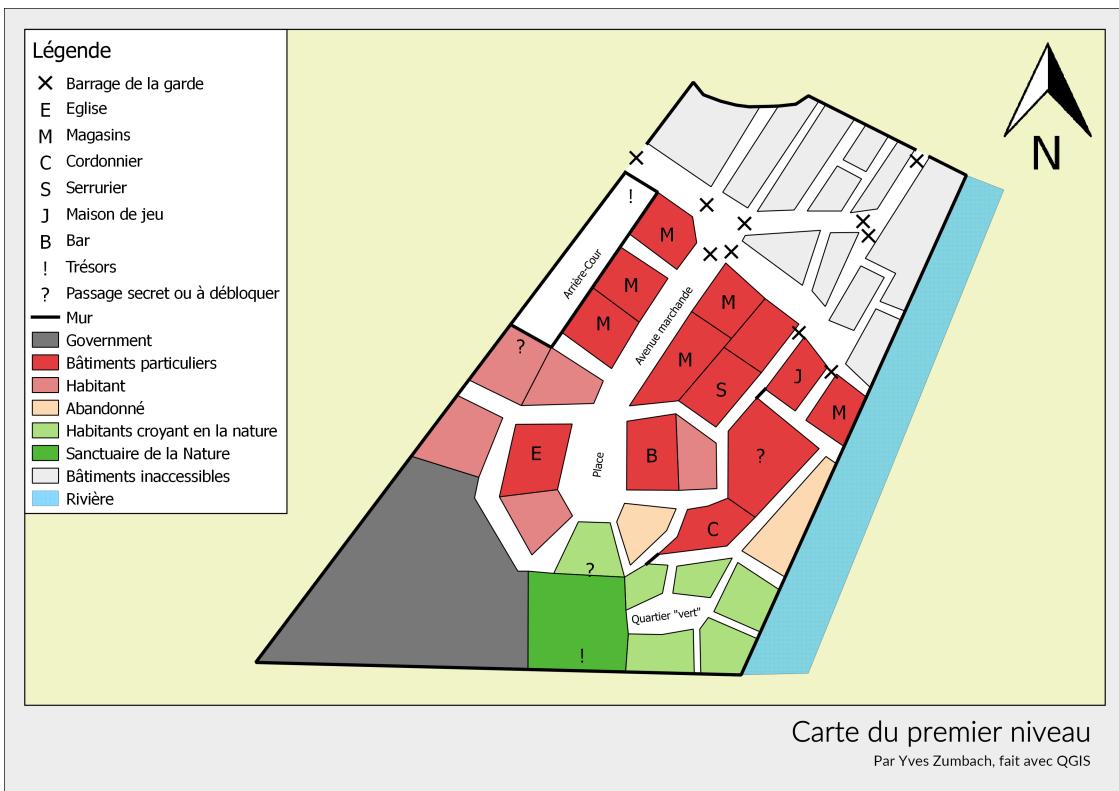


Importance du troisième niveau

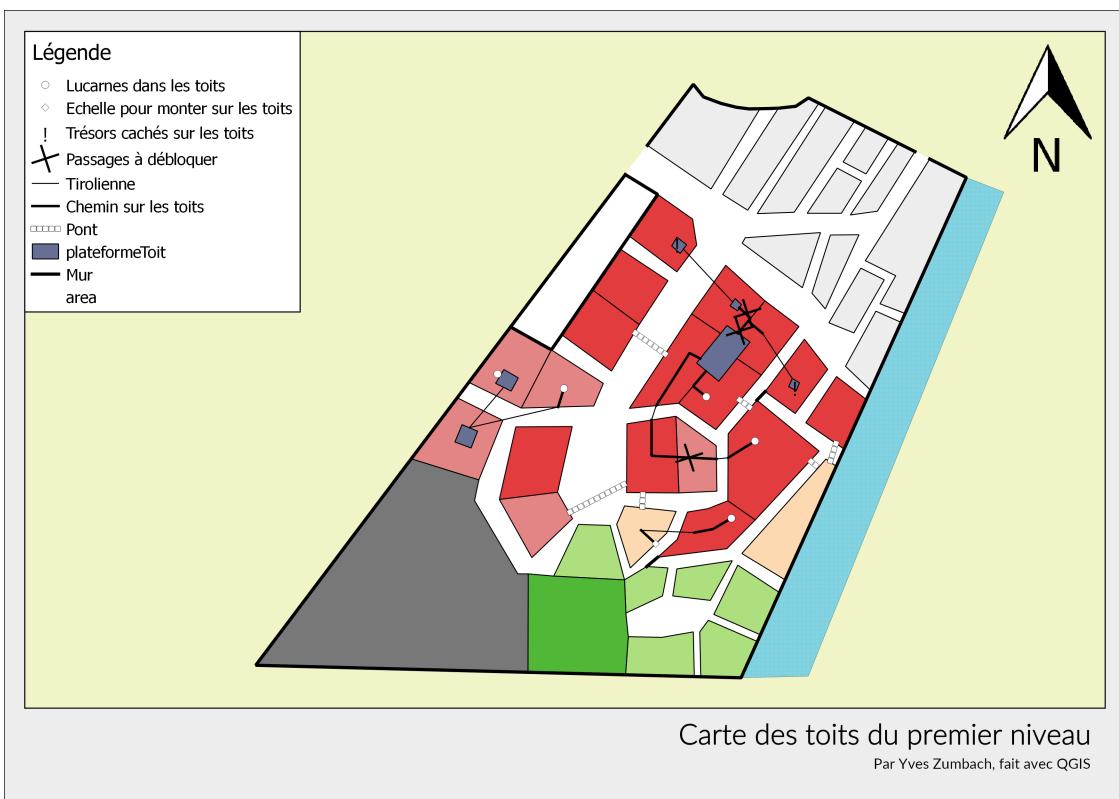
Le troisième niveau est véritablement la clé de voute de cette histoire. On y découvre que Lenaï est passée du côté de Gaamon. Cet élément capital sera le point de départ de l'histoire d'*Eluria's Chronicles*.

4.2 Deuxième acte

Kida se retrouve hors des murs pour la première fois de sa vie. Elle va devoir s'habituer à un nouveau mode de vie plus rural, plus sauvage et découvrir les populations environnant Murtos. Ces personnes sont pour la plupart des renégats ou des paysans ayant fui lors de la création de la ville. La vie est très dure. Elle apprendra ici la légende fantastique du peuple des Telurans. Pour sauver sa sœur, elle se mettra à leur recherche et finira par découvrir, caché au fond d'une forêt, un très ancien temple à l'architecture étrange, gracieuse mais sortie tout droit d'un autre âge : un temple Teluran. C'est en haut de ce dernier qu'elle découvrira un engin volant aussi vieux que le temple. Il la conduira sur les hauts-plateaux, où le peuple persécuté a trouvé refuge.



(a) Plan du quartier pauvre



(b) Plan des toits

Fig. 4.1 : Cartes du premier niveau

4.3 La suite

Une fois arrivée chez les Telurans, Kida découvre que le peuple magnifique en lequel elle espérait n'est plus que l'ombre de lui-même. Les persécutions incessantes de Gaamon ont fini d'achever les dernières merveilles de ce peuple : le palais royal tombe en ruine, les secrets de l'énergie verte, cette science magnifique, se perdent chaque jour, un par un.

A terminer !!!!

4.3.1 Jouer Lenaï

La plupart des niveaux sont joués avec Kida. Cependant, certains utiliseront Lenaï comme personnage ; ce qui apporte de multiples avantages : réutilisation des contenus, points de vue diversifiés, aperçu de la vie de Lenaï. Ce dernier point est celui qui a le plus de valeur pour moi. Il permet d'approfondir le gameplay : lorsqu'on joue Lenaï, on est forcé d'exécuter les ordres de Gaamon, il faut donc compléter les missions données par le grand ennemi du jeu. Le tyran commande à ses soldats des missions à l'éthique hautement critiquable : collecter des impôts énormes chez les habitants, aller éradiquer la Nature sur une surface toujours grandissante autour de la cité, etc. Ces missions seront l'occasion de proposer un choix au joueur quant à leur résolution. Par exemple pour la collecte des impôts, le joueur pourra retourner les maisons des habitants incapables de payer pour trouver jusqu'à la dernière pièce, une solution simple et efficace, ou alors faire l'effort de trouver lui-même des pièces pour aider les personnes les plus en difficulté. Ces niveaux seront aussi l'occasion d'explorer la relation entre Lenaï et Gaamon, à la fois paternelle et ambivalente car Lenaï, bien qu'elle soit « ensorcelée », perçoit bien que ce qu'elle fait n'est pas correct.

4.3.2 Des batailles aériennes

Les véhicules, et plus particulièrement les dirigeables, sont un élément important du Steampunk. Il pourrait être intéressant de créer des batailles aériennes entre des aéronefs telurans datant de leur époque glorieuse, propulsé à l'énergie verte et des forteresses volantes humaines.

Chapitre

5

Réalisation technique



Sommaire

- 5.1 Workflow*, 30
- 5.2 Programmes, 30
 - 5.2.1 Modélisation 3D, 30
 - 5.2.2 Le retour de la 2D : Textures et images, 32
 - 5.2.3 Réalisation des personnages, 32
 - 5.2.4 Moteur de jeu, 33
- 5.3 Quelques bases pour survivre dans l'univers des jeux vidéo, 33
 - 5.3.1 Sommets, arêtes et faces, 34
 - 5.3.2 Calcul des coordonnées, 34
 - 5.3.3 Normales, 35
 - 5.3.4 A not so short introduction to GDScript, 35
- 5.4 Techniques utilisées dans *Eluria's Chronicles*, 36
 - 5.4.1 Textures, 36
 - 5.4.2 Highpoly vers Lowpoly, 37
 - 5.4.3 Collision, 38
 - 5.5 Codes d'exemple, 39
- 5.5.1 Gestion de la caméra, 39
- 5.6 Quelques problèmes rencontrés parmi d'autres, 42
 - 5.6.1 Les normales, de Sketchup à Blender, 42

5.1 Workflow*

La création d'un jeu passe par deux étapes principales :

- La modélisation des objets présents dans le jeu.
- Leur intégration dans un moteur de jeu* au moyen d'un langage informatique.

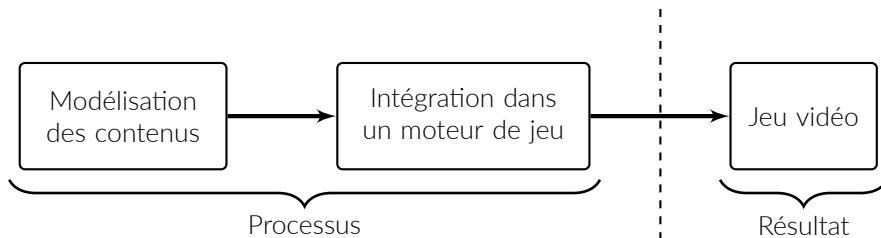


Fig. 5.1 : Workflow de la création d'un jeu vidéo

Un objet désigne n'importe quel élément visible. C'est ainsi une entité qui possède une géométrie ; cela inclut, par exemple, les bâtiments, les meubles, la nourriture ou encore les véhicules. Tous les contenus d'un jeu vidéo, qu'ils soient en 2 ou 3 dimensions, doivent être créés de façon informatique. Dans le cas d'un jeu à 2 dimensions, des logiciels comme Photoshop – ou The Gimp, son équivalent libre – seraient utilisés pour créer des images qui seraient animées directement dans le moteur de jeu. Pour des jeux en 3 dimensions, cela se complique légèrement : il faut utiliser des programmes spécialisés, plus complexes, pour modéliser des objets dans l'espace, donc en trois dimensions. Les animations doivent, elles aussi, être créées dans ces programmes ; les fonctionnalités d'un moteur de jeu ne permettant pas en moyenne de les réaliser. Seul le contrôle (et non pas la réalisation) des animations est effectué depuis le moteur de jeu.

Une fois créés, il est nécessaire de mettre tous les objets ensemble pour qu'ils forment un tout cohérent – un niveau. Il faut ensuite programmer les actions du joueur, vérifier que le jeu suit bien le scénario, gérer le son et travailler sur l'ambiance du jeu. Cela se fait dans le moteur de jeu grâce à la programmation.

5.2 Programmes

5.2.1 Modélisation 3D

J'utilise Blender pour la modélisation des contenus 3D de mon jeu. C'est un programme gratuit, Open source et supporté par une grande communauté. Très complet, il rivalise avec les plus grands logiciels propriétaires. Il permet de créer des objets, de leur donner couleurs et textures (voir section 5.4.1), de les animer et bien plus encore. Initialement prévu pour faire des films d'animation, il est tout à fait recommandé de l'utiliser dans le cadre de la création de jeux vidéo.



Les films officiels de la Blender Foundation, disponibles sur youtube, donnent un bon aperçu des capacités du programme : *Elephant dream* (2006), *Big Buck Bunny* (2008), *Sintel* (2010) ou encore *Tears of steel* (2013).

J'utiliserai également Google Sketchup. Ce programme de modélisation – bien moins puissant et développé que Blender – est cependant nettement plus simple à utiliser et possède l'avantage d'être lié à la 3D Warehouse de Google (littéralement « entrepôt 3D ») : un site web recensant des milliers d'objets réalisés avec ce logiciel.

La réalisation des contenus d'un jeu vidéo est une étape très longue et demande des connaissances poussées de modélisation, design et programmation pour ne citer que quelques-uns des domaines concernés. C'est une tâche dont s'occupe en général une équipe complète ; la réalisation de *Grand Theft Auto V* a nécessité la collaboration de 1000 personnes [5]. Étant seul, je me servirai de la 3D Warehouse pour trouver certains modèles que j'utiliserai afin de tenter de finir une version de démonstration du jeu dans un temps raisonnable. Google Sketchup servira d'intermédiaire de conversion entre le site web et Blender. Il est cependant à noter que, bien que la 3D Warehouse soit une ressource énorme, les contenus qu'elle propose ne correspondent pas toujours parfaitement aux besoins du jeu. Il faut donc modifier la plupart de ces éléments afin qu'ils soient adaptés.

Un autre problème de taille avec ce flux de travail réside dans la compatibilité entre Blender et Google Sketchup. La transition d'un programme à l'autre n'est de loin pas transparente et nécessite un certain nombre de manipulations. Il n'est d'ailleurs pas rare qu'à leur arrivée dans Blender les objets possèdent des défauts (sommets dupliqués, faces mal connectées, etc.).

Modélisation d'une maison

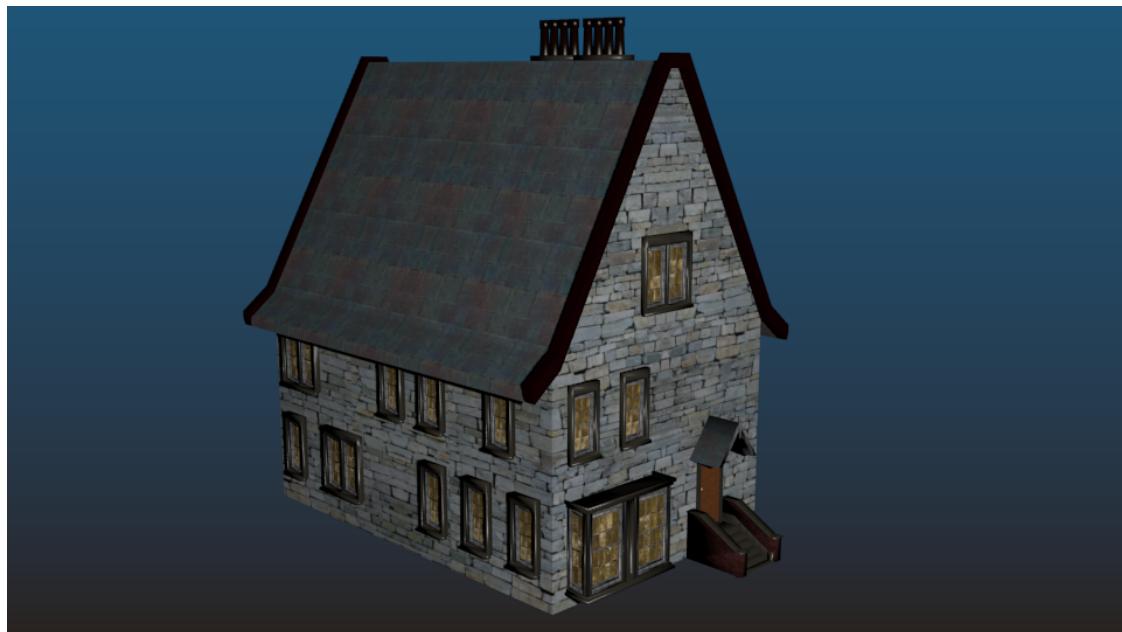


Fig. 5.2 : Une maison réalisée dans Blender

Voilà les éléments qui ont été nécessaires pour modéliser la maison qu'on peut voir à la figure 5.2 :

- Mur de la maison : La géométrie a été réalisée à la main. La texture de mur en pierre provient d'Internet.
- Le toit : Modélisé à la main ; la texture qui permet de réaliser les tuiles vient de la 3D Warehouse.
- La bordure du toit : Modélisée à la main ; cet objet n'a pas de texture mais un seulement un

matériaux réalisés dans Blender.

- La porte : Modélisée à la main, c'est en fait un amalgame de trois objets : le cadre de la porte, la porte et la poignée de la porte. Chacune des parties a son propre matériau.
- Les fenêtres simples (9x), doubles (3x) et doubles proéminentes, les cheminées (2x), l'avant-toit et les escaliers : Proviennent de la 3D Warehouse. Les géométries ont été simplifiées dans Blender ; les textures viennent de la 3D Warehouse et d'Internet, elles ont majoritairement été refaites avec Gimp (augmentation des contrastes, assombrissement). Chaque objet est constitué de plusieurs sous-objets.

Deux objets supplémentaires ont été créés pour ajouter des collisions à la maison. L'un des deux s'applique à la maison entière (pour que le joueur ne puisse pas traverser les murs), l'autre à la porte afin que cette dernière puisse être détectée par raycast (voir section [5.4.3](#) pour plus de détails sur les collisions).

En tout, la maison comporte :

- 9757 sommets
- 11714 faces

À titre de comparaison, un cube simple est formé de 8 sommets et 6 faces.

5.2.2 Le retour de la 2D : Textures et images



www.gimp.org

Un jeu 3D ne peut pas se passer d'images ! Elles donnent vie aux objets en remplissant la fonction de textures*(voir section[5.4.1](#)). Les interfaces du jeu, comme les menus ou le HUD, sont faites exclusivement à partir d'images. Il faut donc un programme de création/édition d'image. Pour cela, j'utiliserai GIMP (*GNU Image Manipulation Program*). C'est un programme Open source, activement soutenu par une grande communauté, ayant atteint un stade de maturité avancé et pour lequel une multitude de tutoriels existent.

5.2.3 Réalisation des personnages



www.makehuman.org

Un des éléments vitaux pour ce jeu est la création des personnages et de leur animation. Il existe un outil libre permettant de faire cela : MakeHuman. C'est donc ce dernier que j'ai utilisé pour modéliser tous les personnages présents dans le jeu. Les add-ons* pour Blender MakeWalk et MakeCloth auront respectivement servi à animer et habiller les personnages.

5.2.4 Moteur de jeu

Godot, le moteur de jeu utilisé, est un programme Open source et gratuit, distribué sous la licence MIT. Il présente un avantage notable : sa compatibilité avec Blender. En effet, l'auteur voulait permettre aux utilisateurs du moteur d'utiliser des outils libres du début à la fin. Il est relativement simple à utiliser comparé à d'autres comme UDK ou UE4 et le langage de script qu'il utilise est un dérivé de Python dont la syntaxe est très simple et très lisible.



En revanche, le moteur est toujours en développement, il n'est donc pas totalement mature, comporte encore certaines erreurs et toutes les fonctionnalités ne sont pas disponibles. Le projet étant jeune, la communauté supportant ce projet n'est pas aussi importante que ce que l'on pourrait souhaiter et surtout, la documentation n'est pas complètement écrite. Ce dernier point est le plus critique : il implique de devoir tester au hasard les options non-documentées jusqu'à en comprendre le fonctionnement, ce qui est coûteux en temps et en énergie.

! Configuration des axes dans Godot

Godot définit les axes de la façon suivante :

- X : droite, gauche
- Y : haut, bas
- Z : devant derrière

À peu près tous les moteurs font pareil. La raison est que, ainsi défini, les axes X et Y représentent les mêmes directions en 2 et en 3 dimensions, ce qui permet une plus grande compatibilité.

Cette convention est utilisée implicitement dans le reste du document.

5.3 Quelques bases pour survivre dans l'univers des jeux vidéo

! Important

Les quelques bases données dans les sections suivantes sont très importantes pour la compréhension de la suite de ce travail. Elles sont utilisées implicitement dans la plupart des explications de ce document.

5.3.1 Sommets, arêtes et faces

Les jeux en trois dimensions sont constitués principalement d'objets, appelés *meshes* en anglais. Ces objets sont, pour leur quasi-totalité, formés exclusivement de faces, d'arêtes et de sommets (*mesh* signifie « réseau » – ici de sommets, d'arêtes et de faces, soit une géométrie). Les programmes comme Blender sont spécialisés dans l'édition de tels objets (voir section [5.2.1](#)). D'autres types d'objets existent ; ils peuvent, par exemple, être formés à partir de courbes. Ces derniers, même s'ils sont utilisés dans les films d'animation, sont en général évités dans les jeux vidéo pour des raisons de performance (voir section [5.4.2](#)).

5.3.2 Calcul des coordonnées

Référentiels

Dès qu'on parle d'univers en 3 dimensions, la notion de référentiels prend une grande importance. Ces derniers permettent de décrire un point dans l'espace grâce à trois coordonnées : x , y et z , indiquant une distance depuis l'origine, toujours située au point $(0, 0, 0)$, de ces mêmes axes (voir figure [5.3](#)).

On peut ensuite définir des sous-référentiels ou « référentiels locaux » dans le « référentiel global », aussi nommé « monde ». Pour définir une telle entité, il faut un point qui donnera l'origine locale et trois vecteurs qui renseigneront l'orientation et l'échelle des axes x' , y' et z' . Ces vecteurs donneront l'unité de base des axes dans le référentiel global (ils sont d'ailleurs nommés « vecteurs-unité »). Cela permet une grande flexibilité ; pour modifier un objet dont les coordonnées sont renseignées localement, il suffit de modifier les vecteurs-unité des axes locaux (rotation et dilatation) ou l'origine (translation).

La figure [5.3](#) donne une idée de ce fonctionnement :

- Le référentiel local, défini par les axes x' , y' et z' , a son origine au point global $(6, 8, 2)$. Les traits rouges indiquent les coordonnées de cette dernière.
- Les axes x' , y' et z' sont définis respectivement le long des axes x , y et z . Leur norme – ou longueur – équivaut à 1.
- L'objet « cube » est défini dans le référentiel local. Les coordonnées d'un des points du cube sont données par les traits bleus.

Il suffit maintenant de changer l'origine du référentiel pour que le cube soit translaté ou de modifier les vecteurs-unité des axes x' , y' et z' pour que ce cube pivote ou change d'échelle.

Exemple : Imaginons un instant qu'on fasse pivoter le référentiel local autour de l'axe z' de 90°. L'axe y' sera orienté horizontalement vers la gauche et x' pointera vers le haut. Les coordonnées du cube sont données dans le référentiel local, cela implique que, si localement cela ne change rien, dans l'espace global, le cube pivote en même temps que ces axes.

Tous les programmes de modélisation ainsi que les moteurs de jeu utilisent cette méthode : chaque objet est défini dans un référentiel local.

Calcul matriciel

Pour calculer les coordonnées à l'écran des sommets d'un objet, il est donc nécessaire de convertir les coordonnées locales dans l'espace global puis de projeter ces dernières sur un plan 2D qui

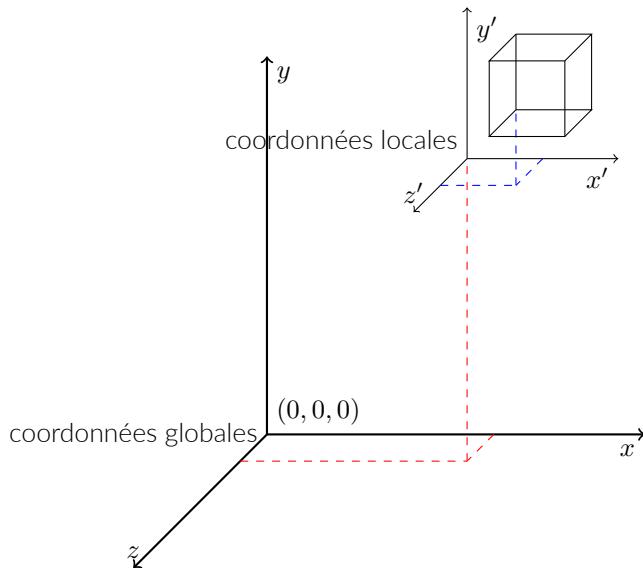


Fig. 5.3 : Référentiels et systèmes de coordonnées

représente l'écran. Ces opérations sont effectuées grâce au calcul matriciel¹ et correspondent à un changement de référentiel ainsi qu'à une projection.

5.3.3 Normales

Les normales sont des vecteurs utilisés pour décrire l'orientation d'une face dans l'espace. Elles permettent de savoir sur quel plan se situe une face. Ce sont des vecteurs normalisés (de longueur 1), perpendiculaires en tout point à la face. Dans les moteurs graphiques comme Godot, elles indiquent aussi dans quelle direction pointe la face (elle a un « haut » et un « bas »).

5.3.4 A not so short introduction to GDScript



Confusion sur le sens du mot « objet »

Le mot « objet » peut prendre deux sens extrêmement distinct selon le contexte dans lequel il est utilisé. Lorsqu'on parle de modélisation, un objet représente une forme en trois dimensions avec des textures, qui sera, selon toute probabilité, intégrée dans le jeu. En programmation, il désigne un concept fondamental de l'informatique : une structure qui peut contenir des fonctions et des variables. La programmation orientée objet est un élément vital dans les logiciels modernes et les jeux vidéos ne font pas exception à cette règle.

GDScript est le langage utilisé dans Godot. Sa syntaxe est très proche de Python et très lisible. Voilà les quelques bases à connaître pour pouvoir comprendre les extraits de code présentés ici.

Le mot-clé **var** sert à déclarer une variable, à savoir une entité qui peut contenir n'importe quel type d'objet : un entier, un nombre à virgule, un tableau, un dictionnaire², etc.

¹Une matrice est une entité mathématique qu'on représente comme un tableau de chiffre

²En programmation, un dictionnaire est un tableau particulier qui fait correspondre des clés (souvent des chaînes de caractères) à des valeurs.

Le mot-clé **func** permet de définir une fonction. Il est immédiatement suivi par le nom de la fonction définie ainsi que d'une paire de parenthèses qui peut éventuellement contenir des arguments.

Tout ce qui suit un **#** (et qui est sur la même ligne) est un commentaire. En programmation, cette notion désigne du texte qui n'est pas considéré comme du code et est ainsi ignoré par l'ordinateur. Ces caractères sont destinés uniquement aux utilisateurs qui liront le programme afin, le plus souvent, de faciliter la compréhension du code. Tous les codes présentés dans ce texte sont largement commentés.

Le langage GDScript utilise le niveau d'indentation pour délimiter les blocs logiques. Le nombre de « tabulations¹ » avant un mot définit son niveau d'indentation, plus il y en a et plus la ligne est indentée. Un bloc logique est composé : de lignes à la suite les unes des autres possédant le même niveau d'indentation, éventuellement de sous-blocs logiques.

⟨⟩ Les bases de GDScript

```
1 var ajouteMoiDeux = 4 #declaration d'une variable
2 plusDeux(ajouteMoiDeux) #appel d'une fonction avec la variable
3 ajouteMoiDeux passée en paramètre
4
5 func maFonction() : #cette ligne déclare une nouvelle fonction
6     nommée maFonction
7     var variable1 = 7 #nouvelle variable dont la valeur est 7
8     variable2 = ['un', 'deux', 'trois'] # variable assignée à un
9         tableau contenant trois chaînes de caractères
10
11 func plusDeux(argument1) :
12     argument1 = argument1 + 2
13
14 #On constate bien que le niveau d'indentation a une signification
15 #importante : les lignes 1 et 2 sont du code normal exécuté
16 #immédiatement; les lignes 5 et 6 appartiennent à la ligne 4 et
17 #forment un groupe logique qui définit une fonction.
```

Code 5.1 : basics.gd

5.4 Techniques utilisées dans *Eluria's Chronicles*

5.4.1 Textures

Les textures sont des images appliquées sur un objet pour lui donner de la couleur, ajouter des détails à la géométrie (voir la section 5.4.2 sur les *normal map* et *bump map*), modifier la quantité ou la couleur des reflets, définir les zones qui émettent de la lumière et sa couleur (*glow map*), etc. Elles sont très largement utilisées dans les jeux vidéos ainsi que les films d'animation car peu coûteuses en matière de calcul.

Pour appliquer une image en 2 dimensions sur un objet en 3 dimensions, la technique la plus souvent

¹La touche nommée « tab » sur le clavier permet d'ajouter des caractères de tabulation. Ces derniers ajoutent une espace blanche plus grande que l'espace habituelle (souvent équivalent à 3 ou 4 espaces)

utilisée est appelé *UV unwrapping*¹. L'idée est de « découper » un objet afin de le déplier et de le poser « à plat » (voir figure 5.4). Une fois cette opération effectuée, il devient simple d'appliquer la texture sur l'objet. Cette méthode à cependant le désavantage de distordre les faces des objets compliqués.

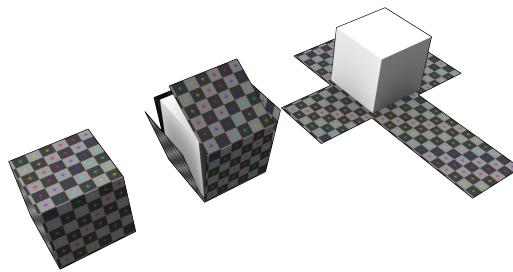


Fig. 5.4 : UV Unwrapping d'un cube

Dans *Eluria's Chronicles*, tous les objets dont la couleur n'est pas simplement unie utilisent une texture de couleur. De même, les objets qui émettent de la lumière comme les enseignes lumineuses utilisent des « textures de rougeoiement » (*glow texture*). Les images sont créées dans GIMP, appliquées dans Blender par *UV unwrapping* et finalement exportées dans le moteur de jeu pour le rendu final.

5.4.2 Highpoly vers Lowpoly

Le temps de calcul par image est un aspect très important lors de la réalisation d'un jeu vidéo. Trop élevé, il empêchera l'ordinateur d'afficher suffisamment d'images à la seconde pour rendre le flux vidéo fluide. Parmi les facteurs qui influent sur cette valeur critique, on notera principalement le nombre de sommet et de faces dans la scène. En effet, il faut calculer pour chaque face la position à l'écran des sommets (voir section 5.3.2), la quantité de couleur émise, les reflets, le z-index ou indice de profondeur², etc.

Les objets utilisés doivent donc être les plus économies possible en matière de géométrie. Une technique très couramment utilisée pour diminuer le nombre de sommets avec un minimum de perte de qualité est celle du *normal mapping*.

On commence par réaliser un modèle très simplifié de l'objet. Ce dernier ne fera que rarement plus de 500 faces dans le cas d'un objet non-animé – c'est le Lowpoly (*low*, bas ou peu ; et *poly* pour polygone ; c'est le polygone à faible résolution). On le duplique ensuite pour ajouter à la copie tous les détails désirés. Ce deuxième modèle possèdera en moyenne des millions de faces. Une fois cette opération terminée, on « cuit » (traduction littérale du terme anglais *bake* qui désigne cette manipulation) les différences de hauteur entre les deux objets sur une texture (voir section 5.4.1). Autrement dit, les différences de hauteur se trouvent renseignées sur une image 2D, codées en couleurs.

Il est possible de générer plusieurs types de texture : des *normal map* (pas vraiment de traduction) ou des *bump map* (« placage de relief » en français). Les premières sont codées en trois couleurs – Rouge, Vert, Bleu – soit trois informations qui représentent les translations sur les axes *X*, *Y* et *Z* de

¹UV ne signifie pas ultra-violet mais plutôt les axes U et V de la texture. En effet, pour ne pas confondre les coordonnées de la texture avec d'autres, les axes ont été nommés différemment.

²Le z-index décrit quelle est la face la plus proche de la caméra afin de n'afficher que cette dernière. Sans lui, ce serait la dernière face calculée qui serait affichée...

chaque point. La deuxième, plus simple, est une image en niveau de gris, soit à une seule dimension (le degré de noirceur) qui déterminera la translation de chaque point le long de la normale (voir section 5.3.3) de la face.

Ces textures sont appliquées sur l'objet Lowpoly. L'ordinateur calcule ensuite les ombres et certains détails comme si les sommets s'étaient réellement déplacés, donnant l'impression d'un niveau de détail très élevé bien que la géométrie de l'objet n'ait pas changé. Ceci permet, sans trop de perte de qualité, de faire passer le nombre de sommets par objet de plusieurs millions à quelques centaines seulement. Cette technique est très appréciée et surtout très utilisée ; on la retrouve dans tous les jeux vidéos modernes.

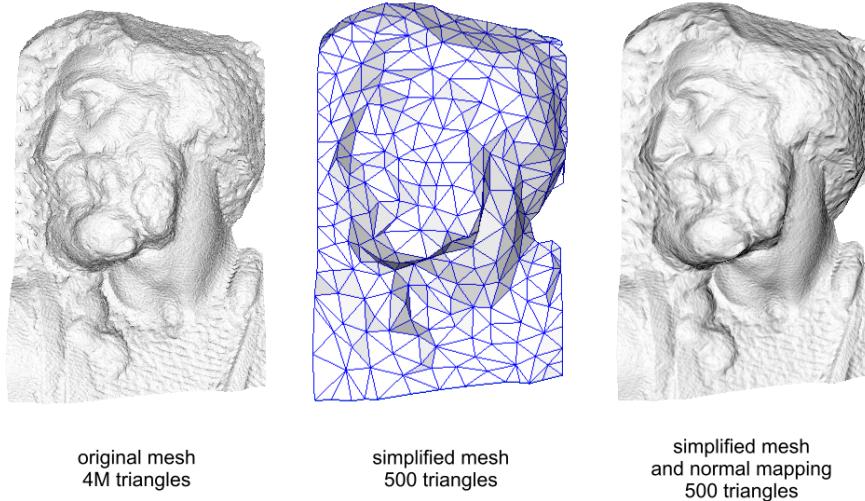


Fig. 5.5 : Exemple de *normal mapping* [10]

5.4.3 Collision

Si nous avons tous une perception intuitive des collisions avec la matière et de leurs effets dans la réalité, l'ordinateur n'a pas cette connaissance. Pour que l'univers virtuel devienne tangible, solide, il faut que l'ordinateur calcule les chocs et les conséquences qu'ils ont dans le jeu. Il faut donc ajouter pour chaque objet des « collisions », soit une géométrie supplémentaire qui exécute du code lorsqu'un autre objet vient à la chevaucher. Il existe différents types de réponses possibles : l'objet peut ne pas bouger du tout, ricocher, faire ricocher l'autre, etc.

Une grande partie de ces calculs compliqués est prise en charge par le moteur physique de Godot. Il peut s'occuper de certains comportements par défaut comme les objets statiques ou ceux effectuant des mouvements simples. Mais les collisions d'un joueur, par exemple, sont nettement plus complexes à gérer : il faut qu'il puisse sauter ce qui implique de la gravité (ajoutée par programmation) ; s'il y a des combats dans le jeu, le joueur doit pouvoir être touché et les dégâts éventuellement calculés en fonction du lieu d'impact, etc. Tout cela doit être codé à la main.

Le *Ray cast* également nécessite des collisions pour pouvoir fonctionner. Cette technique permet de tirer un rayon¹ depuis un objet ou l'écran et de voir quelle est la première géométrie de collision rencontrée. Cela permet de déterminer beaucoup de choses utiles : quel objet se trouve sous le curseur du joueur, l'ennemi peut-il détecter le joueur ou un mur bloque-t-il son champ de vision, etc.

¹Aucun rayon n'est jamais tiré, c'est simplement une métaphore qui représente, dans la réalité, des calculs mathématiques.

5.5 Codes d'exemple

5.5.1 Gestion de la caméra

Dans un jeu vidéo à la première personne¹, l'orientation de la caméra est toujours gérée par l'utilisateur; pour les jeux d'ordinateur, c'est généralement la tâche de la souris.

Il faut donc convertir les mouvements de la souris (translations 2D) en mouvements de caméra (rotations 3D). Les coordonnées x de la souris détermineront les rotations horizontales (autour de l'axe Y) et les coordonnées y , celles verticales (autour de l'axe X). Toutes ces opérations doivent se faire sur le référentiel local de la caméra. Cependant, il est impossible de réaliser cette opération de la façon suivante :

$$\begin{aligned}\delta x \cdot \kappa(v_{rotation}) &= rotation_Y \\ \delta y \cdot \kappa(v_{rotation}) &= rotation_X\end{aligned}$$

Avec $\kappa(v_{rotation})$, le coefficient pour modifier la vitesse de rotation de la caméra, et δx et δy , les mouvements de la souris (considérés comme des angles en radians).

En effet, imaginons le cas suivant :

1. On fait une rotation verticale (selon l'axe local X) pour que la caméra regarde vers le haut avec un angle de 45
2. On fait pivoter la caméra horizontalement (selon l'axe local Y)

L'étape numéro 2 va poser problème. En effet l'axe vertical local (Y) de la caméra a pivoté lors de l'étape 1 et fait un angle de 45. Une rotation autour de cet axe aura pour conséquence de faire tourner la caméra « de biais », donnant l'impression que la caméra tombe de son pivot. Un tel effet n'est évidemment pas désiré.

Il faut donc trouver une autre formule pour convertir les mouvements de la souris en rotation de caméra. Après quelques recherches sur internet, j'ai finalement trouvé la solution que je cherchais : la formule de conversion entre coordonnées sphériques et cartésiennes[6].

Les coordonnées sphériques servent à représenter des points dans l'espace à l'aide de sphères (voir la figure 5.6). Elles sont très souvent utilisées en cartographie. Trois informations sont nécessaires :

- Le rayon ρ de la sphère sur laquelle se situe le point P (soit la distance origine-point).
- Un angle horizontal, compris entre 0 et 2π , noté θ
- Un angle vertical, compris entre 0 et π , noté φ

Ces coordonnées fonctionnent à l'aide de deux angles, ce qui est exactement ce que les mouvements de la souris fournissent.

Les coordonnées cartésiennes sont celles dont nous avons l'habitude, avec trois composantes : x , y et z . La formule de conversion entre les deux systèmes est la suivante :

$$\left\{ \begin{array}{l} x = \rho \cos \theta \\ y = \rho \sin \theta \cos \varphi \\ z = \rho \sin \theta \sin \varphi \end{array} \right. \quad (5.1)$$

Nous pouvons maintenant transformer les mouvements de la souris sur les axes X et Y , que nous considérons comme des angles, en coordonnées cartésiennes. Mais comment faire pivoter la caméra de la bonne façon avec de telle données ?

¹Un jeu vidéo à la première personne est un jeu vidéo dans lequel le joueur est le héros et vit par ses yeux. À l'inverse, dans un jeu vidéo à la troisième personne, le joueur voit le personnage qu'il contrôle ; il est à l'extérieur du corps du personnage.

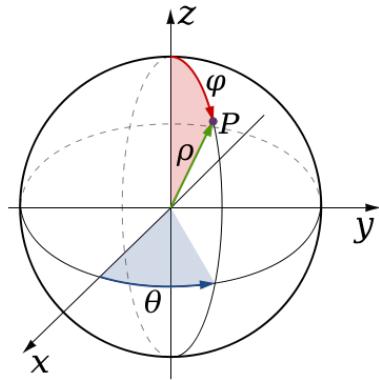


Fig. 5.6 : Coordonnées sphériques

Godot fournit une fonction – `Transform.looking_at(Vector3 point, Vector3 haut)` – qui s'applique à un objet de type `Transform` (qui représente un référentiel, voir section 5.3.2) et retourne une copie de ce référentiel mais pivoté de manière à ce qu'il regarde en direction de point. Il ne reste plus qu'à calculer les coordonnées de point. Pour ce faire, il faut ajouter aux coordonnées globales de la caméra le vecteur normalisé obtenu grâce à la formule 5.1. Point se trouvera ainsi toujours dans une sphère de rayon 1 autour de la caméra et elle le « suivra des yeux ».

i `get_transform` et `set_transform`

Les méthodes `get_transform` et `set_transform` permettent respectivement d'obtenir et de définir le système de coordonnées locales d'un objet (orientation et position). C'est donc ces dernières qui sont utilisées pour faire pivoter la caméra.

`</>` Code de la caméra

```

1 var view_sensitivity = 0.1
2 var angleHorizontal = 0 #angle horizontal de départ
3 var angleVertical = 90 #angle vertical de départ (on regarde droit devant)
4
5 func _ready() : #cette fonction est exécutée lorsque l'objet est créé
6     set_transform(get_transform().looking_at(Vector3(0, 0, 1) +
7         get_transform().origin, Vector3(0, 1, 0))) # définit le premier angle de la camera
8
9     Input.set_mouse_mode(2) #souris invisible, fixée au centre
10    set_process_input(true) #la fonction _input sera exécutée à chaque fois que la caméra reçoit des inputs (dont les mouvements de souris)
11
12 func _input(ev) : #cette fonction est exécutée à chaque fois que la souris bouge
13     # gère la rotation de la caméra en fonction des mouvements de la souris
14     if ev.type==InputEvent.MOUSE_MOTION :

```

```

14     angleHorizontal = fmod(angleHorizontal + ev.relative_x *
15                             view_sensitivity, 360) #cette ligne ajoute à
16                             angleHorizontal le déplacement de la souris sur l'axe x
17                             multiplié par la sensibilité de la caméra puis retourne
18                             le reste de la division entière de angleHorizontal par
19                             360 afin que cette variable ait toujours une valeur entre
20                             0 et 360
21     angleVertical = angleVertical + ev.relative_y *
22                             view_sensitivity #on ajoute à angleVertical le
23                             déplacement de la souris sur l'axe y multiplié par la
24                             sensibilité de la caméra
25
26     #les quatre lignes suivantes empêchent angleVertical d'avoir
27     #une valeur supérieure à 180 ou inférieure à 0
28     if angleVertical>180 :
29         angleVertical = 180
30     if angleVertical<0 :
31         angleVertical = 0
32
33     set_transform(get_transform().looking_at(where_to_look_at(
34         angleHorizontal, angleVertical) + get_transform().origin,
35         Vector3(0, 1, 0))) #cette ligne est très importante,
36         elle change l'orientation de la caméra pour qu'elle
37         pointe vers le point obtenu en ajoutant l'origine de la
38         caméra au résultat de where_to_look_at
39
40     func where_to_look_at(horizontal, vertical) :
41         return Vector3(sin(deg2rad(vertical)) * sin(deg2rad(horizontal)),
42                         cos(deg2rad(vertical)), -1 * sin(deg2rad(vertical)) * cos(
43                         deg2rad(horizontal)))
44
45     #c'est cette fonction qui fait la conversion entre coordonnées
46     #sphériques et cartésiennes C'est le cœur de l'algorithme. On
47     #remarquera d'ailleurs que la formule utilisée est la même que
48     #celle décrite juste au dessus
49     # (horizontal : 0°=regarde droit devant, positif a droite, negatif
50     # a gauche)
51     # (vertical : 90°=regarde a l'horizontale, 0°=regarde en haut, 180=
52     # regarde en bas)
53     # Les angles doivent êtres compris entre -360 et 360 pour
54     # horizontal, entre 0 et 180 pour vertical

```

Code 5.2 : cameraControls.gd

5.6 Quelques problèmes rencontrés parmi d'autres

5.6.1 Les normales, de Sketchup à Blender

Afin de réaliser le moins de calculs possible, certains moteurs de jeu n'affichent que les faces qui pointent vers la caméra ; à savoir, uniquement les faces dont la normale est orientée dans sa direction. Ceci permet, théoriquement, de diviser par deux le nombre de faces dont il faut faire le rendu et ainsi le temps de calcul pour une image. Bien que les cartes graphiques qui font ces calculs soient de plus en plus puissantes et que les techniques d'optimisation de calcul soient de moins en moins nécessaires, beaucoup de moteurs de jeu utilisent ce procédé.

Godot n'est pas une exception. Il est également possible d'afficher toutes les faces, quelle que soient leur direction, mais cela n'est pas recommandé ; en effet, pour que le jeu reste fluide, il faut, de manière standard, 60 images par seconde (ou fps pour *Frame Per Second*), 24 au minimum.

Cette spécificité des jeux vidéos ne pose généralement pas de problème. Par exemple, dans le cas d'objets modélisés dans Blender, étant donné que tout s'y fait à partir de formes primitives (cubes, cylindres, etc.) dont les faces sont bien orientées, la transition vers un moteur de jeu se fait sans problème. Mais prenons maintenant le cas d'objets importés de la 3D Warehouse de Google (voir section [5.2.1](#)) puis convertis pour Blender. Le programme utilisé par défaut pour le visionnement et l'édition des fichiers de la librairie en ligne, Google SketchUp, n'est pas (ou peu) sensible à l'orientation des faces. Ainsi, certains modèles arrivent dans Blender avec une partie de leurs normales mal orientées... Mais Blender n'est pas touché par ce problème : tout comme Google SketchUp, il affiche toutes les faces, quelle que soit leur disposition. Et au moment de passer l'objet dans le moteur de jeu, la moitié des faces disparaissent, car le programme, lui, est sensible à la direction des normales !

Le modèle, dans cet état, est inutilisable dans le moteur de jeu. Deux solutions s'offrent alors : inverser à la main ou de façon semi-automatisée les faces dans Blender, ce qui représente un travail long et fastidieux. Ou alors, changer carrément d'objet, par exemple en le modélisant à la main. Heureusement, tous les fichiers n'ont pas de ce problème. Mais ceux qui en souffrent sont souvent irrécupérables et font perdre un temps précieux. De plus, ce problème, lorsqu'il est rencontré pour la première fois surprend beaucoup (et peut causer un certain énervement).

5.6.2 Importation des textures dans Godot

A ecrire !

Chapitre

6

Pour conclure

Sommaire



- 6.1 Diffusion, 44
- 6.2 Réaliser un jeu vidéo libre ?, 44
- 6.3 Réaliser un jeu vidéo seul ?, 44
- 6.4 Bilan personnel, 44

6.1 Diffusion

Si *Eluria's Chronicles* vient un jour à être distribué ce sera gratuitement et librement, bien sûr ! Chacun pourra télécharger le jeu et l'essayer sans payer. Mais plus encore, chacun sera libre de télécharger et modifier son contenu puis de le redistribuer (sous la même licence). Il pourrait même être envisageable de rendre le jeu public avant qu'il ne soit terminé et demander de l'aide à la communauté pour le terminer mais cela implique beaucoup de coordination et une certaine perte de contrôle sur la version officiel du jeu.

6.2 Réaliser un jeu vidéo libre ?

Pour la réalisation de ce jeu, je m'étais fixé comme condition de n'utiliser que des logiciels gratuits. Atteindre cet objectif fut relativement simple. Il existe, de par le Net, une multitude de programmes gratuits et c'est sans difficulté que j'ai trouvé les outils qui m'étaient nécessaires.

Pour ce qui est du deuxième objectif, à savoir utiliser le plus possible des logiciels libres, je fus agréablement surpris de constater qu'ils sont également nombreux et divers. En effet, il existe, pour la plupart des tâches que nécessite la création d'un jeu, des outils open source. Un seul programme a fait exception à la règle pour *Eluria's Chronicles*: Sketchup. J'avais besoin de ce dernier pour convertir les modèles téléchargés d'Internet vers des formats plus aisément éditables. Impossible de s'en passer donc et impossible de trouver une alternative. Mais ce n'est qu'une goutte au milieu de l'océan.

À mon goût, les objectifs fixés au début de ce document sont atteints. C'est pour moi la preuve qu'Internet pourrait apporter une révolution dans notre mode de fonctionner : un monde où les gens travailleraient pour la communauté, partageraient et amélioreraient sans attendre de contrepartie, où l'information serait libre et gratuite et où les outils seraient disponibles pour tous ceux qui en auraient besoin.

6.3 Réaliser un jeu vidéo seul ?

L'expérience fut moins concluante de ce point de vue. S'il était évident que je ne pourrais finir un jeu complet en moins d'une année, je suis cependant un peu déçut de n'avoir pu mener ce projet plus loin, dans le temps imparti. Certaines tâches qui me semblaient aisées ont parfois demandé des heures de recherches et certains bugs m'ont coûtés un temps énorme, précieux, passé à tenter de comprendre la source du problème. C'est donc une esquisse un peu moins avancée que ce que j'espérais que je rends.

Le Travail de Maturité se termine donc, mais pas le projet *Eluria's Chronicles*. Je n'ai aucune raison de m'arrêter maintenant ; j'ai déjà un scénario et les bases du jeu. Il ne me reste « plus » qu'à finir ce qui est déjà débuté. Et qui sait, dans quelques années *Eluria's Chronicles* sera peut-être enfin prêt.

6.4 Bilan personnel

Pour être honnête, ce Travail de Maturité fut pour moi une source de stress ; j'aurais voulu porter la réalisation du jeu plus loin, ajouter plus de détails, de fonctionnalité. S'engager dans un travail d'une telle ampleur avec la volonté de le mener à bout peut causer des tensions considérable.

Mais il est évident que ce Travail m'a aussi beaucoup apporté. J'ai dû persévérer malgré les difficultés

et parfois l'énerverment. Ce fut l'occasion d'acquérir toutes sortes de connaissances techniques : modélisation, programmation, édition de son, création de cartes, etc. J'ai aussi découvert l'univers du game designer, je suis en quelque sorte passé « de l'autre côté de l'écran ».

Annexe

A Glossaire

Mot	Définition
Add-on	Un add-on, ou plug-in, est un morceau de code qui est ajouté à un programme afin de lui ajouter une ou plusieurs fonctionnalités spécifiques qui ne sont pas fournies par défaut.
Boss	Anglicisme qui désigne l'ennemi principal du jeu ou du niveau. Il peut y avoir plusieurs boss s'ils sont propres aux niveaux. Ces personnages sont, en général, rencontrés en combat au moins une fois. L'exemple le plus connu est « Bowser », un dragon diabolique, qui enlève la belle « Peach » dans le jeu « Mario »
Démo	Abbréviaction de « version de démonstration » ; désigne une version incomplète d'un jeu vidéo. Il peut s'agir d'un nombre diminué de niveaux, de possibilités de jeu bridées ou de toute autre méthode réduisant la version intégrale du jeu.
Gameplay	Anglicisme désignant la façon dont le joueur peu interagir avec le jeu, les actions qui lui sont permises, les objets qui sont à sa disposition, les façons de faire permettant de surmonter les défis du jeu, etc.
HUD	Acronyme de <i>Head-Up Display</i> . C'est toute l'interface 2D présente sur l'écran du joueur alors qu'il est dans le jeu (vraiment en train de jouer, pas les menus). Cela peut comprendre par exemple la barre de vie, l'argent disponible, les armes équipées ou encore la minimap. Ce sont des informations qui apparaissent comme s'il y avait une vitre devant vos yeux sur laquelle on projetait des images. Initialement, cela désignait bien des vitres sur lesquelles étaient projetées des informations de vol... dans les cockpit des avions de chasse.
Moteur de jeu	Un moteur de jeu est un programme chargé de centraliser plusieurs aspects de la création de jeu vidéo : importation de contenu (3D ou 2D), calculs pour le rendu des images (dans le cas d'un univers en 3D : projection de cet environnement sur un espace 2D – l'écran), gestion du son, programmation des actions ou des personnages (ou scripting), exportation vers diverses plateformes, etc.
Open-world	Anglicisme ; littéralement « monde ouvert ». Désigne un jeu vidéo dans lequel le joueur peut se déplacer librement à travers un univers virtuel très vaste. Il y dispose d'une grande liberté d'action.
PNJ	Abréviation tirée du domaine du jeu de rôle, signifiant Personnage Non Joueur, soit n'importe quel personnage qui n'est pas contrôlé par un homme. Ils désignent, dans un jeu vidéo, les protagonistes de l'histoire contrôlés par l'ordinateur.

Mot	Définition
Puzzle	Une énigme, une devinette ou un mini-jeux à résoudre pour pouvoir avancer dans le jeu, débloquer des trésors cachés ou encore compléter des objectifs secondaires. Ils peuvent être de formes très diverses : épreuve de rapidité, d'adresse, de réflexion, etc. Par exemple, dans le jeu <i>The Legend of Zelda : Spirit Tracks</i> il faut reproduire un morceau de musique avec un instrument débloqué au fil de l'aventure pour accéder à certains lieux.
Screenshot	Anglicisme signifiant « capture d'écran » ; autrement dit, une photo de ce qui s'affiche à l'écran.
Semi open-world	De l'anglais, « monde semi-ouvert ». C'est un niveau ou type de jeu dans lequel le joueur est libre de ses mouvements. Cependant des éléments du décor limitent la taille de l'univers. On trouvera, par exemple, des îles (la mer est l'élément bloquant), des villes fortifiées (les murailles sont les éléments bloquants) ou encore une maison dont on ne peut sortir. Les niveaux de type <i>alley</i> désigne exactement la même chose mais semi open-world ou open-world sont des termes que l'on peut entendre en français. Voir également la définition de « Open-world »
Succès	Traduction de <i>achievement</i> , ce sont des récompenses honorifiques attribuées au joueur à l'accomplissement d'un tâche donnée, d'objectifs supplémentaires, etc.
Story-telling	Anglicisme ; décrit l'art de raconter une histoire efficacement. Cela inclut par exemple : le point de vue narratif, l'intrigue, les personnages mais aussi l'intonation, les gestes, etc. Dans le cadre d'un jeu vidéo, cela décrira les techniques narratives, les libertés laissées au joueur, etc.
Temple	Un temple fait référence, dans l'univers du jeu vidéo, au dernier niveau d'un chapitre ou acte du jeu. Il se déroule généralement dans l'entre de l'ennemi principal ou tout du moins d'un adversaire de grande importance. Ce type de niveau se conclu généralement par un affrontement entre le joueur et l'ennemi cité avant – appelé alors boss. On citera ainsi Super Mario Bros dans lequel chaque dernier niveau se déroule dans un château de Bowser ou encore Zelda où chaque temple conclu une partie du jeu et coïncide généralement avec l'obtention d'un nouvel objet.
Workflow	Anglicisme, que l'on pourrait traduire par « flux de travail ». Cela désigne l'ensemble des étapes que vont subir des informations, des documents ou des produits pour passer d'un premier état à un deuxième.

Annexe

B Steampunk

B.1 Définition

Selon Wikipédia : « L'expression steampunk, qui signifie littéralement “punk à vapeur”, parfois traduite par “futur à vapeur”, est un terme inventé pour qualifier un genre de littérature né à la fin du XXe siècle, dont l'action se déroule dans l'atmosphère de la société industrielle du XIXe siècle. Le terme fait référence à l'utilisation massive des machines à vapeur au début de la révolution industrielle puis à l'époque victorienne. Aujourd'hui le steampunk est considéré comme un esthétisme pouvant intéresser à la fois des œuvres littéraire fantastique, de fantasy, d'anticipation et certains sous-genres de la science-fiction. »[7]

D'autres définissent ce genre comme étant : « la technologie moderne – iPads, ordinateurs, robots, avions – alimentée par la vapeur et insérée dans le 18^e siècle ». [8]

Le Steampunk est un des univers fantastiques que j'apprécie le plus, il regroupe en même temps science-fiction, fantaisie et caractéristiques rétros. Il possède à la fois un côté technologique et une ambiance sombre qui en font un environnement mystérieusement attrant.

B.2 Quelques images



<http://godsofart.com/wp-content>



frpnet.net



www.pinterest.com



<http://pocketguys.com>



www.roleplaygateway.com



www.pinterest.com



www.howtogeek.com

Bibliographie

- [1] *Logiciel libre*. In : Wikipédia. 10 fév. 2015. url : http://fr.wikipedia.org/w/index.php?title=Logiciel_libre&oldid=111729358 (visité le 14/05/2015).
- [2] *Open source*. In : Wikipédia. 8 avr. 2015. url : http://fr.wikipedia.org/w/index.php?title=Open_source&oldid=113672946 (visité le 14/05/2015).
- [3] *Jeu vidéo*. In : Wikipédia. 13 mai 2015. url : http://fr.wikipedia.org/w/index.php?title=Jeu_vid%C3%83%C2%A9o&oldid=114978590 (visité le 14/05/2015).
- [4] Scott Rogers. *Level Up ! The Guide to Great Video Game Design*. John Wiley & Sons Publication. 2010. 492 p.
- [5] Rockstar : More than 1,000 people made GTAV. GameSpot. url : <http://www.gamespot.com/articles/rockstar-more-than-1000-people-made-gtav/1100-6415330/> (visité le 24/08/2015).
- [6] Coordonnées sphériques. In : Wikipédia. 3 août 2015. url : https://fr.wikipedia.org/w/index.php?title=Coordonn%C3%83%C2%A9es_sph%C3%83%C2%A9riques&oldid=117412011 (visité le 13/08/2015).
- [7] Steampunk. In : Wikipédia. 7 mai 2015. url : <http://fr.wikipedia.org/w/index.php?title=Steampunk&oldid=114797617> (visité le 15/05/2015).
- [8] What is Steampunk ? url : <http://www.ministryofpeculiaroccurrences.com/what-is-steampunk/> (visité le 16/05/2015).

Iconographie

- [9] Sinto-risky. Stone Golem. url : <http://sinto-risky.deviantart.com/art/Stone-Golem-199577323>.
- [10] Normal mapping used to re-detail simplified meshes. url : https://en.wikipedia.org/wiki/Normal_mapping.
- [11] Wikipédia. A diagram of spherical coordinates, defining a point by colatitude, φ , longitude, θ , and radius, ρ . url : [https://upload.wikimedia.org/wikipedia/commons/5/51/Spherical_Coordinates_\(Colatitude,_Longitude\).svg](https://upload.wikimedia.org/wikipedia/commons/5/51/Spherical_Coordinates_(Colatitude,_Longitude).svg) (visité le 13/08/2015).
- [12] Matt69ers_sl. Above City Wallpaper - Steampunk Wallpaper. url : <http://godsofart.com/wp-content> (visité le 23/05/2015).
- [13] City of Steam Açık Beta Başlıyor. url : frpnet.net (visité le 23/05/2015).
- [14] Steampunk Teslapunk on Pinterest. url : www.pinterest.com (visité le 23/05/2015).
- [15] The Scavengers. url : www.roleplaygateway.com (visité le 23/05/2015).
- [16] Late Afternoon Train Travelling Through a Steampunk City. url : www.howtogeek.com (visité le 23/05/2015).
- [17] Steampunk City on Pinterest | Steampunk Airship, Steampunk Illustration and Lady Mechanika. url : www.pinterest.com (visité le 23/05/2015).

Table des figures

1.1	Space Invaders, Pong, Pacman et Breakout	9
1.2	Tetris et Super Mario Bros	9
1.3	Street Fighter II, Doom, Myst et Final Fantasy VI	9
1.4	Zelda, Call of Duty, Sonic et Angry Birds	9
2.1	Symbole de Gaamon	12
2.2	Plan de la ville actuelle (Réalisé à l'aide du programme Quantum GIS)	13
2.3	Propagande religieuse	14
2.4	Propagande étatique	15
2.5	Gnobil	16
2.6	Une esquisse de Leo, basée sur le travail original de Sinto-risky [9]	18
4.1	Cartes du premier niveau	27
5.1	Workflow de la création d'un jeu vidéo	30
5.2	Une maison réalisée dans Blender	31
5.3	Référentiels et systèmes de coordonnées	35
5.4	UV Unwrapping d'un cube	37
5.5	Exemple de <i>normal mapping</i> [10]	38
5.6	Coordonnées sphériques	40

Liste des tableaux

2.1	Les trois esprits et leurs caractéristiques	17
3.1	La gestion des ressources	21

Liste des codes

5.1	basics.gd	36
5.2	cameraControls.gd	40