

PROFESSEUR : VIVIANE GAL

Murmur

Une rumeur en marche

ENJMIN M1

05/12/2008

L'exercice était de développer un jeu autour de la technologie Game Maker. Une première version quasi définitive devait être remise au bout de 14 jours et une présentation finale faite 7 jours plus tard. Le thème proposé était « Dérober » avec tous ses homophones.

SOMMAIRE

| | | |
|----------|--|-----------|
| 1 | Introduction..... | 4 |
| 1.1 | Rappel de la consigne..... | 4 |
| 1.2 | Note d'intention | 4 |
| 1.3 | Présentation de l'équipe..... | 4 |
| 2 | Game design | 5 |
| 2.1 | Descriptif..... | 5 |
| 2.2 | La carte de la ville | 5 |
| 2.3 | Déroulement du jeu..... | 5 |
| 3 | Conception visuelle | 6 |
| 3.1 | la charte graphique | 6 |
| 4 | Conception sonore | 12 |
| 4.1 | Utilisation de FMOD..... | 12 |
| 4.2 | Première partie | 12 |
| 4.3 | Seconde Partie | 13 |
| 4.4 | Habillage sonore des « cinématiques »..... | 14 |
| 5 | Programmation | 14 |
| 5.1 | Généralités..... | 14 |
| 5.2 | Décomposition du jeu..... | 14 |
| 5.3 | Gestion du son | 15 |
| 5.4 | Gestion des collisions avec les bâtiments de la map..... | 15 |
| 5.5 | Gestion des jauges | 15 |
| 5.6 | Niveau 1 : la room Ville | 15 |
| 5.7 | Niveau 2 : la room Siège | 17 |
| 5.8 | Problèmes rencontrés | 18 |
| 5.9 | Améliorations à apporter..... | 19 |
| 6 | Ergonomie..... | 19 |
| 6.1 | Ergonomie vis-à-vis de ce projet..... | 19 |

6.2 Recommandation en phase de conception..... 19

6.3 Recommandations après Beta-Test 20

7 Conclusion 22

1 INTRODUCTION

1.1 RAPPEL DE LA CONSIGNE

L'exercice était de développer un jeu autour de la technologie Game Maker. Une première version quasi définitive devait être remise au bout de 14 jours et une présentation finale faite 7 jours plus tard. Le thème proposé était « Dérober » avec tous ses homophones.

1.2 NOTE D'INTENTION

DÉROBER - SE DÉROBER : UNE RUMEUR EN MARCHÉ.

Deux sens de *dérober* nous ont intéressés :

- « *dérober* » comme synonyme d'un vol à l'insu d'autrui.
- « *se dérober* » comme synonyme d'une échappée, d'une fuite.

Cette polysémie nous est apparue comme un atout majeur. Nous souhaitions en offrir une vision plurielle. Pour ce faire, nous voulions décliner « *dérober* » de deux manières distinctes.

Nous avons pris comme point de départ **la rumeur** qui semblait accueillir ce double sens de *dérober* :

- « *dérober* » : voler le libre arbitre par propagation de la rumeur.
- « *se dérober* » : éviter la rumeur qui s'est propagée.

Ces deux sens correspondent à un changement de point de vue, et donc à un changement de gameplay.

« *Dérober* », « *se dérober* » nous permet de faire passer le joueur du rôle de bourreau, comme agent propagateur de la rumeur, à celui de victime.

Cette ambivalence met en place un gameplay réflexif qui pousse le joueur à s'interroger sur son rôle et sa responsabilité de joueur.

1.3 PRESENTATION DE L'EQUIPE

L'équipe était composée de 7 personnes comme suit :

- Conception sonore : Arnaud Noble
- Conception visuelle : Ronan Kergosien et Pierre Lemasson
- Ergonomie : Raphaël Sautron
- Game design : Romain Bonnin
- Gestion de projet : Mickaël Godard
- Programmation : Antoine Sarafian

2 GAME DESIGN

2.1 DESCRIPTIF

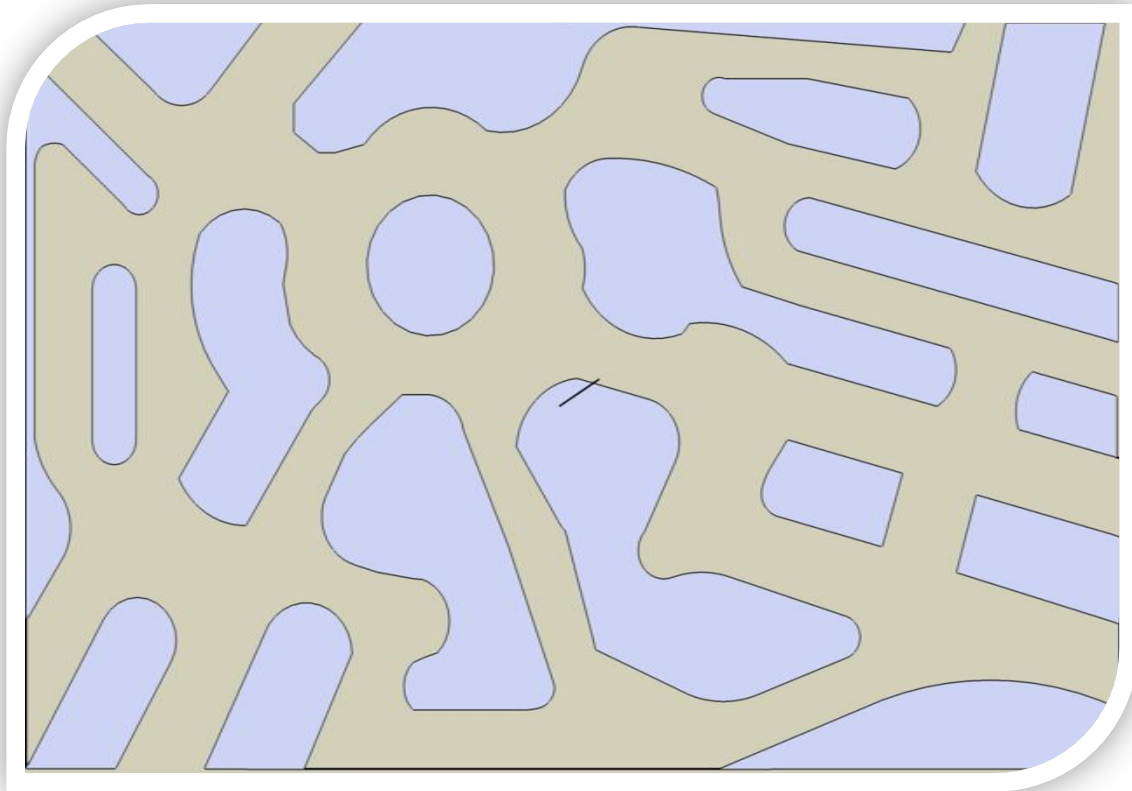
En annexe tous les documents de game design sont présentés (dans le dossier « Annexes ») afin de montrer l'évolution du concept.

Murmur est un jeu de rumeur. Le jeu se déroule en deux étapes :

Dans une première phase le joueur doit propager la rumeur, il est la rumeur (Mur 1), et dans la seconde, le joueur incarne la victime de cette rumeur (Mur 2). Le joueur joue donc successivement les rôles du bourreau (Mur 1) puis de la victime (Mur 2).

Murmur est un jeu en une partie unique. Il comporte un court récit avec un dénouement ouvrant sur une réflexion du joueur.

2.2 LA CARTE DE LA VILLE



2.3 DEROULEMENT DU JEU

2.3.1 INTRODUCTION

Le jeu démarre sur une courte introduction (écrite ou cinématique) qui nous présente le contexte et le cadre de l'histoire : M.Mur habite une cité où tout le monde se connaît. Un

jour, des policiers l'emmènent sans ménagement. Des témoins de cette intervention vont colporter une rumeur transformant notre M. Mur, présumé innocent, en un dangereux criminel.

Le jeu démarre au début de la propagation de cette rumeur.

2.3.2 PREMIERE PHASE DE GAMEPLAY

Le joueur doit propager la rumeur en cliquant sur des personnes qui se croisent. Chaque personne supplémentaire ayant connaissance de la rumeur fait augmenter la jauge « Rumeur ».

Les variables de gameplay sont :

- La vitesse des personnages.
- La vitesse du curseur.
- Le nombre de personnage.

2.3.3 PHASE DE TRANSITION

Une courte cinématique explicite la transition entre les deux phases et donne au joueur la consigne de jeu de la deuxième phase.

2.3.4 DEUXIEME PHASE DE GAMEPLAY

Le joueur incarne la victime de la rumeur. La rumeur a provoqué chez lui une forte dépression. A chaque personne qu'il croise, il se sent opprimé et sa jauge de dépression augmente.

Le joueur cherche donc à fuir les personnes qu'il croise, en vain. Les rues semblant mener vers une « sortie » sont bloquées par des foules.

Le joueur finit par perdre obligatoirement.

2.3.5 EPILOGUE

Une courte cinématique (non définitive sur la version du 5/12) explique le dénouement.

3 CONCEPTION VISUELLE

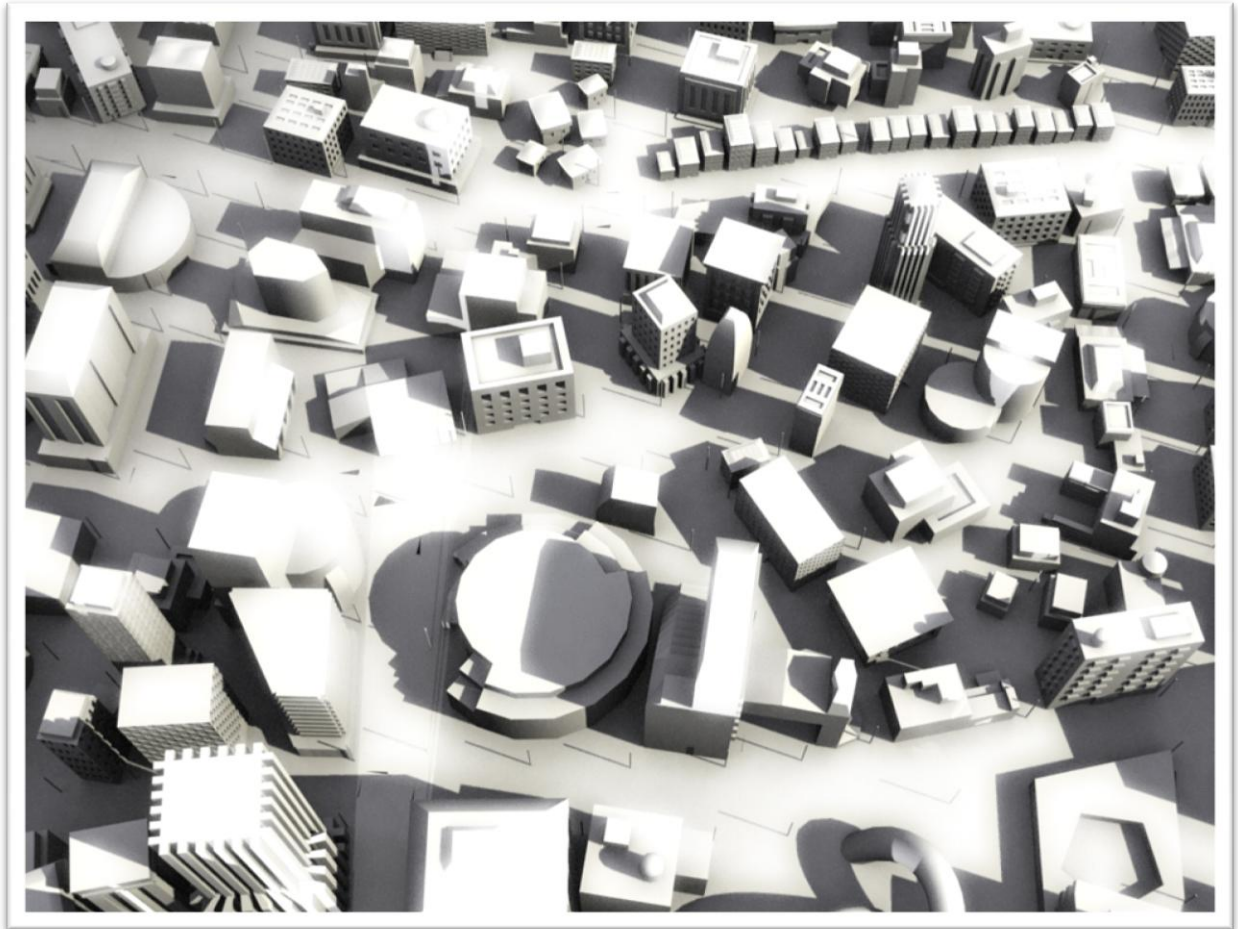
3.1 LA CHARTE GRAPHIQUE

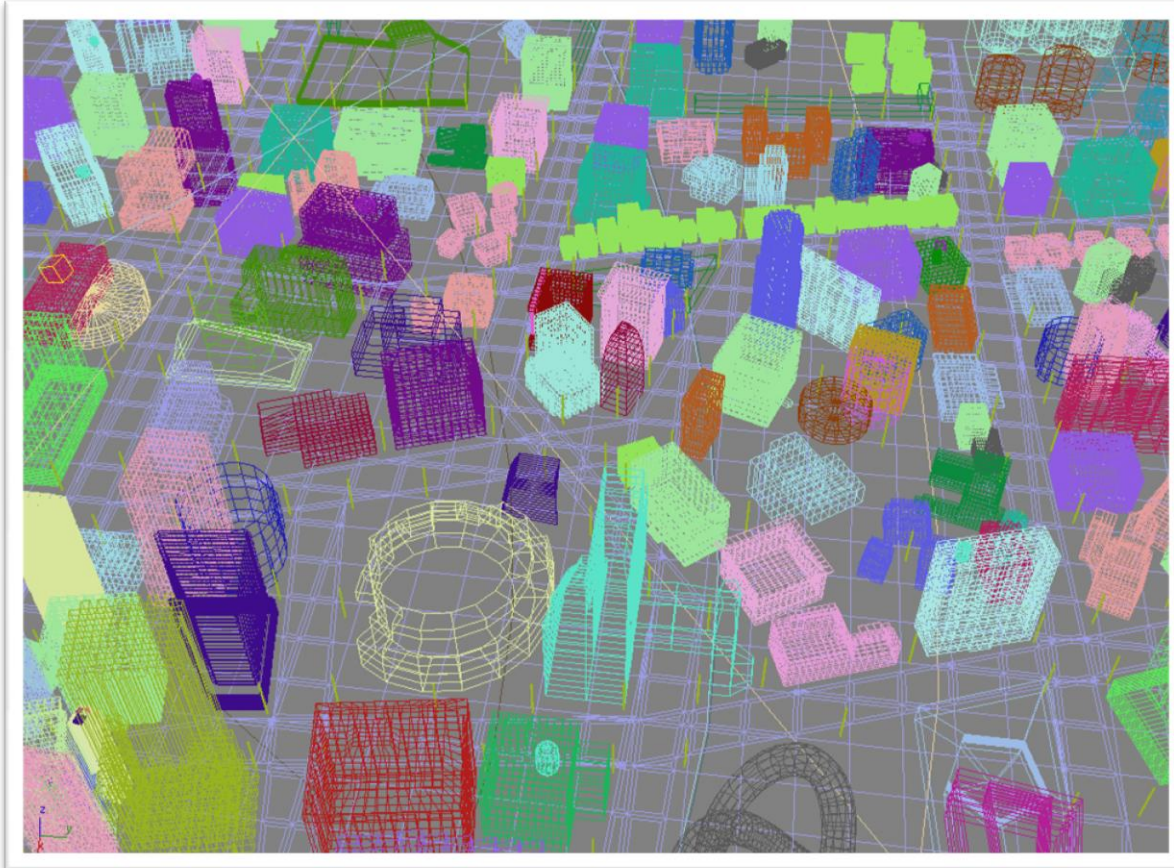
3.1.1 RECHERCHES VISUELLES

Dès que le projet a été formulé, nous avons travaillé à traduire le contenu en image. Nous avons à produire de nombreux éléments : une carte, des *sprites* animés pour chaque personnage, une interface et des contenus linéaires (vidéos ou images/textes d'introductions). Nous réutilisons un point de vue traditionnel dans le jeu vidéo, celui des jeux de stratégie ou de certains jeux de rôle, à savoir une vue isométrique, zénithale,

survolant la carte et les personnages. Cependant le projet ne se basait pas, au niveau du gameplay ou de la narration, sur un schéma classique – il était donc essentiel de ne pas concevoir des visuels trop communs pour pouvoir donner une identité propre à cette expérience interactive.

Plusieurs idées ont été suggérées. Nous n'avons pas retenue l'utilisation de la 3D car en temps réel, elle est difficilement gérée sous *Game Maker*. Nous aurions pu l'utiliser en précalculé, mais elle aurait perdu sa propriété d'objet tridimensionnel et donc une grande part de ce qui lui donnait un intérêt. Il faut aussi noter qu'avec le temps imparti, il aurait été complexe de développer tout les *sprites* via *3Dsmax*. Quand à une intégration *sprites* 2D/décors 3D, cela aurait eu des effets négatifs sur la cohérence de l'ensemble.





Exemples de recherche préliminaire sur un rendu de la carte en 3D.

3.1.2 CHOIX DU MEDIUM

Notre choix s'est porté sur un médium plus léger, peut être aussi plus onirique : le dessin. Il ne s'agissait pas ici d'utiliser le dessin comme quelque chose faisant partie d'un *workflow* de production classique. Nous ne voulions pas le considérer comme une phase préparatoire, suite à quoi nous l'aurions complètement retravaillé et dénaturé sous Photoshop. Il fallait lui garder un aspect très "brut".



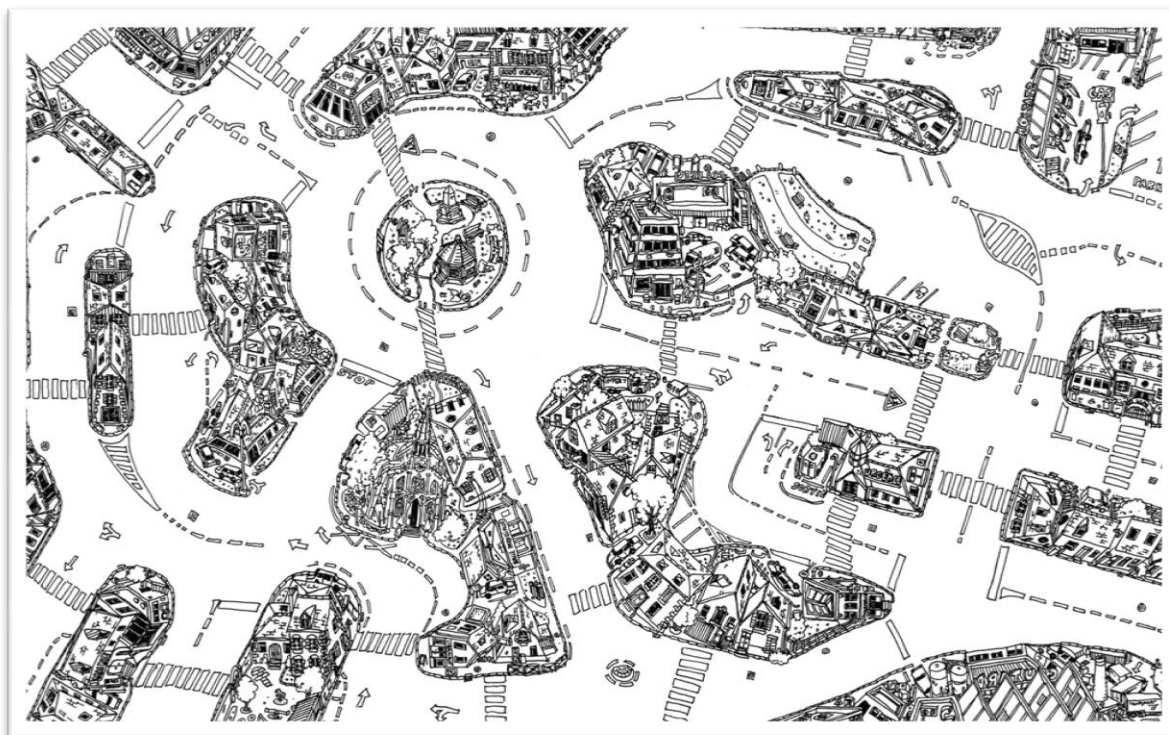
3.1.3 ILLUSTRATION DE LA CARTE



Quitte à être radical dans le choix de la ligne graphique, nous avons jugé bon de ne garder qu'un tracé noir sur fond blanc. Une ligne claire mais très dense, apportant une somme de détails importants sur l'environnement du jeu, pour peu que le joueur s'y attarde. Le dessin nous a permis de nous affranchir de quelque chose de très récurrent dans le jeu vidéo, à savoir la perspective. Il était impossible d'utiliser une perspective conique étant donné qu'il ne s'agissait pas de 3D temps réel. Quand aux ersatz de 3D via l'isométrie (losange ou vue redressée), cela aurait eu pour effet de restreindre les possibilités offertes par le dessin. Cette liberté nous a permis d'éclater la perspective, de

multiplier les points de vue. L'ancrage dans une représentation se fait ici par l'accumulation d'éléments graphiques, dans une optique plus Flamande qu'Italienne par analogie avec la peinture. A l'instar de travaux comme les gravures de Paul Velly ou les cartes anamorphosées de Saul Steinberg, on obtient quelque chose de très dense, une impression de grouillement quasiment organique. Cet effet, totalement assumé étant donné qu'il connote fortement avec le propos du jeu, est appuyé par la multitude de petits personnages déambulant aux travers des artères de circulations laissées libres.

Lors de la réalisation des différents éléments graphiques, il a tout de même fallu tenir compte de plusieurs contraintes techniques. Tout d'abord, pour que le déplacement des personnages soit cohérent, nous n'avons pas pu faire de rues trop étroites, ni d'angles trop aigus. Avant de passer sous *Game Maker*, nous avons effectué des tests sous *Flash*. A l'aide d'un algorithme de *pathfinding* codé en action script, nous pouvions utiliser des chemins plus complexes. Mais transposer ce système sous *Game Maker* posait des problèmes, aussi avons-nous opté pour un système de collisions dynamiques. La taille a aussi été restreinte pour éviter les ralentissements lors de l'exécution. La taille de la carte est de 1800 par 1109 pixel, et elle est visualisée sur une fenêtre en 640*480.



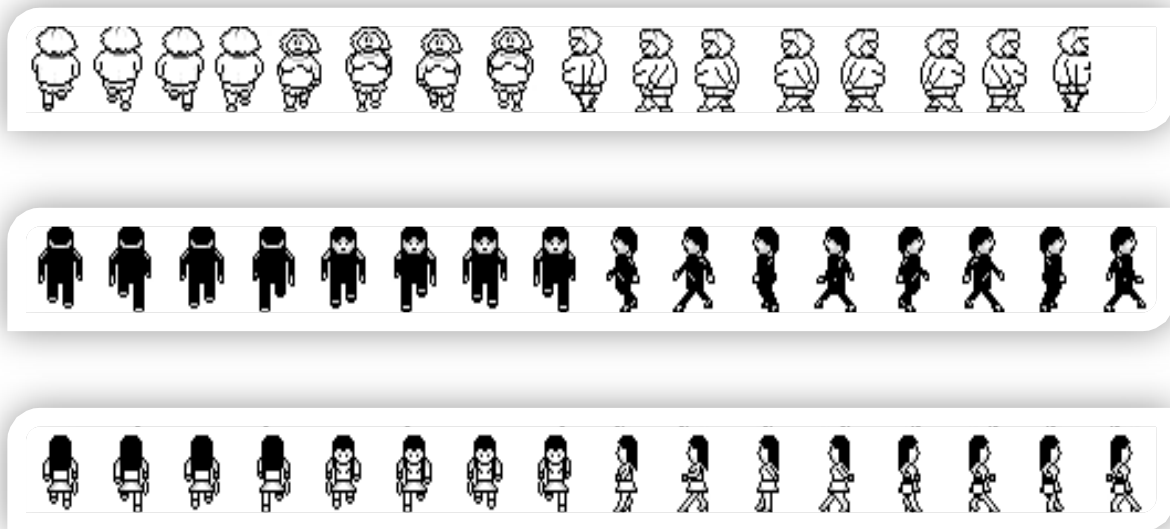
Croquis préparatoire

3.1.4 REALISATION DES ANIMATIONS

Le premier point qui a pesé dans le développement des animations pour les personnages a été de définir les limites techniques de *Game Maker*. C'est-à-dire trouver quelle résolution était la plus adaptée pour un compromis entre nombre de personnages affichés à l'écran et qualité des modèles. Les tests effectués, nous avons convenu que les animations devaient être contenues dans un carré de trente-deux pixels de côté. De même pour le nombre d'étapes d'animation, la vue zénithale obligeait à créer des

mouvements vers le haut, le bas, la gauche et la droite pour chaque personnage, limitant ainsi à 4 images par mouvement.

Une fois ces conditions techniques clairement établies, nous avons pu passer à la réalisation proprement dite, les personnages évoquent beaucoup les jeux sur consoles huit bits et leurs carences en affichage. Ainsi les modèles ont des proportions fantaisistes, d'immenses têtes qui restaient le seul moyen de différenciation entre les personnes dans le jeu. Nous avons porté une attention particulière sur le « héros » puisqu'il fallait se poser la question de sa représentation dans le jeu. Le problème étant que si son modèle était trop proche des autres personnages, le joueur aurait pu le perdre dans la foule de *sprites*, nous ne pouvions pas non plus l'agrandir car cela aurait été à l'encontre du propos du jeu (dans la deuxième phase, le protagoniste est plutôt diminué physiquement et moralement), alors nous avons opté pour un modèle presque entièrement noir mis à part le visage qui le différencie suffisamment de ses congénères sans pour autant lui apporter une dimension plus forte que le reste de la population.



Strips d'étapes d'animation des personnages

3.1.5 L'INTERFACE

L'interface avait comme priorité de rester sobre, les bandes-noires cinématographiques servant à faire le lien entre les vidéos et les séquences de jeu et à intégrer les éléments d'interface. La jauge doit indiquer la progression du joueur dans les deux phases de jeu, elle a un traitement identique pour garder un fil narratif entre les deux parties mais le mot change, « rumeur » puis « dépression », ce dernier permet déjà de se faire une idée fataliste du destin du protagoniste. Le choix de la typographie utilisée a été défini pour rappeler celle des cinématiques. Le remplissage s'effectue en vingt états successifs dans les deux parties.

3.1.6 LES CINEMATIQUES

Concernant les vidéos d'introduction, de transition, et de fin, il fallait garder la cohérence avec le cœur du jeu : à savoir du noir et blanc, et un dessin purement au trait. Pour

dynamiser les cinématiques, nous avons opté pour un système de *papercut* – des dessins découpés sous *Photoshop* puis placés dans l'espace sous *3Dsmax* et *After Effect* dans lequel la caméra peut se déplacer. Nous avons aussi choisi de donner du volume aux dessins en utilisant des hachures, un peu comme une gravure sur cuivre (dans l'esprit des dessins de Robert Crumb).

4 CONCEPTION SONORE

4.1 UTILISATION DE FMOD

Pour des raisons d'optimisation et de fonctionnalités non disponibles dans le moteur audio de Game Maker, nous avons décidé d'utiliser le moteur Audio Fmod.

Compilé dans une DLL externe, nous pouvons faire appel aux différentes fonctions depuis Game Maker.

Voici le fonctionnement global :

```
LoadFMOD(); //Load de la dll fmod

FMODinit(100); //initialisation du nombre de canaux disponibles

global.bkMusic = FMODSoundAdd("bkmusic.mid"); //Définition d'une instance
en affectant un son

FMODSoundLoop("bkmusic.mid"); //lecture du son en boucle

FMODSoundSetGroup(global.bkMusic,3); //Permet de grouper des sons

FMODSoundSetMaxVolume(global.bkMusic,.8); //Défini le niveau maximum d'un
son.

FMODUpdate(); //Rafraichissement du moteur audio (à mettre dans le step)

FMODInstanceFadeVolume(instance, curvolume, targetvolume,numframes,
updatecode) //Permet de gérer un fade d'entrée sorti sur un son
```

4.2 PREMIERE PARTIE

4.2.1 LE FOND SONORE

Ambiance_city.mp3 : son de ville pour immerger le joueur et donner plus de réalisme au monde créé

Mumure_01.mp3 : son de personnes murmurant (tonalité assez grave, avec l'impossibilité de discerner ce qui se raconte).

Mumure_02.mp3 : son de personnes murmurant (tonalité médium, avec l'impossibilité de discerner ce qui se raconte).

Mumure_03.mp3 : son de personnes murmurant (tonalité aigu, avec l'impossibilité de discerner ce qui se raconte).

Mumure_04.mp3 : son de personnes murmurant (tonalité médium, avec l'impossibilité de discerner ce qui se raconte).

La progression des murmures se fait en fonction du nombre de personne atteint par la rumeur (fade in volume par tranche de murmure)

4.2.2 LES BRUITAGES

What_man : son d'exclamation d'homme lorsqu'on click sur une personne pour reprendre la rumeur

What_man_01 : son d'exclamation d'homme lorsqu'on click sur une personne pour reprendre la rumeur

What_man_02 : son d'exclamation d'homme lorsqu'on click sur une personne pour reprendre la rumeur

What_woman : son d'exclamation de femme lorsqu'on click sur une personne pour reprendre la rumeur

What_woman_01 : son d'exclamation de femme lorsqu'on click sur une personne pour reprendre la rumeur

What_woman_02 : son d'exclamation de femme lorsqu'on click sur une personne pour reprendre la rumeur

What_old_man : son d'exclamation de personne âgée lorsqu'on click sur une personne pour reprendre la rumeur

What_child : son d'exclamation d'enfant lorsqu'on click sur une personne pour reprendre la rumeur

4.3 SECONDE PARTIE

4.3.1 LE FOND SONORE

Theme_musique_2eme_gameplay.mp3 : musique d'ambiance qui fade progressivement en fonction du nombre de personnes poursuivant le joueur.

4.3.2 LES BRUITAGES

Mumure_01.mp3 : son de personnes murmurant (tonalité assez grave, avec l'impossibilité de discerner ce qui se raconte).

Mumure_02.mp3 : son de personnes murmurant (tonalité médium, avec l'impossibilité de discerner ce qui se raconte).

Mumure_03.mp3 : son de personnes murmurant (tonalité aigu, avec l'impossibilité de discerner ce qui se raconte).

Mumure_04.mp3 : son de personnes murmurant (tonalité médium, avec l'impossibilité de discerner ce qui se raconte).

La progression des murmures se fait en fonction du nombre de personne poursuivant le joueur. (Fade in volume par tranche de murmure avec diminution de l'ambiance musicale via un fade out progressif).

4.4 HABILLAGE SONORE DES « CINEMATIQUES »

4.4.1 L'INTRODUCTION

Theme_introduction.mp3 : son compose d'une ambiance musicale ainsi que des bruitages pour accompagner la cinématique d'introduction (murmure, sirène de police, talki-walkie, son de fermeture de menottes, murmure réverbéré).

4.4.2 LA TRANSITION ENTRE LES DEUX PARTIES

Theme_introduction.mp3 : reprise du thème musical sans bruitages

4.4.3 LA CLOTURE

Theme_introduction.mp3 : reprise du thème musical sans bruitages

5 PROGRAMMATION

5.1 GENERALITES

Le projet a été réalisé avec la version 7 pro de Game Maker.

Fichier : murmur.gmk

Fichiers externes à intégrer dans le répertoire de l'exécutable :

Les deux DLL nécessaires au bon fonctionnement de la librairie FMOD (fmodex.dll et GMFMODSimple.dll)

Le dossier sons/ contenant les fichiers sonores au format mp3

Le dossier videos/ contenant les fichiers vidéo au format .avi

5.2 DECOMPOSITION DU JEU

Murmur se décompose en deux parties correspondant à deux « rooms » (**Ville** et **Siège**)

A la création de la room Ville, une cinématique d'introduction se lance via la fonction *splash_show_video(name_file : String, loop : boolean)*.

De la même manière, une vidéo de transition se lance au début de la room Siège et à la fin du jeu.

5.3 GESTION DU SON

Les sons sont gérés dans le jeu via la DLL GMFMODSimple (FMOD Sound System, copyright© Firelight Technologies Pty, Ltd., 1994-2007). Cette DLL charge ses fonctions dans le projet (dossier Scripts GMFMODSimple) et nécessite l'utilisation d'un objet **FMODFaderObject** (dans Objects/objets_sons).

Par la suite, un objet dans chaque room (Objets_sonores/**o_son** et **o_son2**) fait office de gestionnaire du son pour la room.

5.4 GESTION DES COLLISIONS AVEC LES BATIMENTS DE LA MAP

Le game design de Murmur imposait une map représentant une ville et des routes sur lesquelles les personnages pouvaient marcher. Les formes des zones « bâtiments » étant courbes, il était nécessaire de gérer des collisions fines autour de ces zones.

La technique employée a donc été de créer des petits objets non visibles de collisions avec sprites (objet Objets_collisions/**o_coll_inst** avec le sprite Collisions/**sprite_coll_inst**) le long de chemins prédéfinis dans Paths/path_collide.

Ces objets de collisions sont initialisés au début de chaque room via la création de l'objet sans sprite Objets_collision/**o_init_coll**.

Cette technique a été trouvée en parcourant les tutoriaux du site www.gamemaker.fr.

5.5 GESTION DES JAUGES

D'après le game design, il fallait que qu'une jauge (soit de Rumeur, soit de Dépression) apparaisse dans les deux parties. Il fallait également que deux bandes noires apparaissent en permanence en haut et en bas de l'écran.

Il a donc été décidé, pour plus de facilité, d'inclure les sprites correspondant à chaque palier de la jauge sur une image transparente avec des bandes noires.

Pour que le joueur ait l'illusion d'avoir un cache et une jauge s'affichant sur l'écran de jeu, on affiche le sprite de la jauge à un instant t à la position correspondante de la vue sur la room (View0 dans les deux rooms). Ainsi, le cache de la jauge « suit » la vue à tout moment.

5.6 NIVEAU 1 : LA ROOM VILLE

Dans la première partie, le joueur dirige le curseur de la souris et doit cliquer sur des personnages non joueurs évoluant de manière aléatoire sur la map de façon à propager la rumeur. La seule manière de propager cette rumeur est de cliquer à l'endroit et au moment

où deux PNJ (au moins) se croisent. A chaque fois qu'un personnage est atteint par la rumeur, une jauge apparaissant à l'écran augmente. Dans l'état actuel du Game Design, la room se termine quand la jauge est complètement remplie.

Il est à noter que l'on a paramétré une vue (View0) sur cette map afin d'effectuer un scrolling qui suit le curseur dirigé par l'écran.

5.6.1 VARIABLES GLOBALES

global.persos_atteints : nombre de personnages atteints par la rumeur

5.6.2 OBJETS

o_perso : objet qui ne fait rien et qui n'est pas créé. Tous les objets des PNJ dérivent de cet objet. Il ne sert qu'à centraliser les actions qui concernent les PNJ

o_perso_pm, ... gm, ... gmo, etc ... : Ce sont les différents PNJ de la map. À chacun de ces objets correspond une série de sprites (ex pour o_perso_pm : les sprites sprite_pm_down, sprite_pm_right, etc ... représentant chacun les directions possibles de l'objet considéré). Ces PNJ changent de direction aléatoirement à chaque fois que Alarm0 sonne. La vitesse de chacun des PNJ est constante.

Event Create : initialisation de la variable bulle (qui indique si l'instance est contaminée par la rumeur), déplacement dans une direction aléatoire puis affichage du sprite correspondant à la bonne direction, paramétrage de Alarm0

Event Alarm0 : Changement de direction et réinitialisation d'Alarm0.

Event Step : gestion des collisions avec les bords de l'écran. En effet, l'événement Intersect Boundary (défini par GM) n'a pas été de tout succès lors des tests. Il a donc été nécessaire de faire autrement et de rendre les collisions avec les bords de l'écran moins sensibles. Il est important de remarquer que selon le bord de l'écran que le PNJ touche, un événement particulier est renvoyé à l'instance (ev_user0, 1, 2, ou 3) pour lui dire de changer de direction à l'opposé du bord.

Event Collision with o_coll_inst : à la collision avec un élément de décor, le personnage change de direction aléatoirement et réinitialise l'Alarm0.

Event User 4 : événement envoyé par le curseur lorsque le joueur clique dans une zone où deux PNJ se croisent. L'action que réalise alors le PNJ est, s'il n'est pas déjà atteint par la rumeur, de créer un objet **o_bulle** au-dessus de sa tête. Est également envoyé un Event User 8 au gestionnaire de son pour jouer un son de feedback.

Remarque : il a été nécessaire de décentraliser le comportement des PNJ à chaque PNJ différents parce que les sprites n'étaient pas les mêmes.

o_bulle : c'est l'objet représenté par le sprite « gribouillis » (sprite_bulle) qui s'affiche au dessus du PNJ quand celui-ci est atteint par la rumeur. Cet objet à une profondeur moindre que les PNJ, de sorte qu'il « passe par dessus » les éléments de décors à l'affichage.

Event Create : on paramètre la vitesse d'animation du sprite et on appelle l'ID de l'instance de `o_perso` qui a créé l'objet.

Event Step : on set les coordonnées de l'objet bulle au dessus de la tête du PNJ qui la porte (de manière à ce que la bulle « suive » bien le personnage)

o_curseur : C'est l'objet représentant le curseur que dirige le joueur. Il correspond au sprite `sprite_curseur`.

Event Create : on ajuste la position du curseur à celle de la souris

Event Step : on ajuste la position du curseur à celle de la souris

Event Left Released : quand le joueur a cliqué, il faut faire deux actions. Tout d'abord, on « trace » un disque autour du point de clic dans lequel on compte le nombre d'instances de `o_perso` qui s'y trouvent. Si on en trouve au moins 2, alors on envoie un événement Event User 4 à toutes les instances de `o_perso` qu'on a trouvé.

C'est donc le curseur qui gère la condition de « propagation » de la rumeur. Le curseur a évidemment une profondeur minimale pour pouvoir apparaître au-dessus de tout à l'affichage.

o_jauge : objet représentant la jauge apparaissant de manière permanente. Sprites correspondants : la série `sprite_jauge`.

Event User 8 : message indiquant la fin du niveau. On définit alors la transition entre les rooms, on effectue cette transition et on détruit les instances de la room. Un Event User 14 est également envoyé au gestionnaire de son pour lui signifier d'arrêter les sons.

Event Draw : affichage du « bon » palier de la jauge (voir commentaires dans le code).

Si la jauge est complètement remplie, on envoie un message de fin de niveau à l'instance self.

5.7 NIVEAU 2 : LA ROOM SIEGE

Dans cette partie, le joueur dirige M. Mur avec les touches directionnelles du clavier. A l'entrée du niveau, on voit six PNJ « s'enfuir » très vite. Par la suite, lorsque le personnage se déplace dans la map, il rencontre des PNJ immobiles qui sont atteints par la rumeur. Lorsqu'il s'approche près d'eux, ces PNJ se mettent à le suivre. Plus de personnages le suivent, plus la jauge de dépression augmente et plus les sons de murmures sont forts. Les issues éventuelles sont également barrées par des sprites immobiles de PNJ. Le niveau se termine lorsque la jauge est complète.

5.7.1 VARIABLES GLOBALES

`global.poursuivants` : le nombre de PNJ qui sont à la poursuite de l'avatar.

5.7.2 OBJETS

o_(perso)2 : On redéfinit les personnages du premier niveau en leur donnant un comportement différent. Ils ont les mêmes sprites.

Event Create : on initialise la variable « rapproche » à false. Cette variable indique si le PNJ poursuit ou non l'avatar. On crée également une instance de o_bulle au-dessus de la tête du PNJ.

Event Step : on vérifie la valeur de rapproche. Si elle est à « true », on effectue une action « move_towards » qui indique au personnage de se diriger vers l'avatar. On indique également quel sprite doit s'afficher en fonction de la direction du PNJ.

User Event 6 : événement correspondant au passage proche de l'avatar. On teste la valeur de « rapproche ». Si elle est à false, on la définit à « true » et on incrémente la variable globale global.poursuivants.

o_(perso)_scripte : ce sont les objets représentant les six personnages qui semblent s'enfuir en début de room. Ils ont chacun un path associé (défini dans path/path_perso). Ils ont les mêmes sprites que les objets PNJ.

Event Create : on associe un path au perso

Event End of Path : on arrête l'objet et on le détruit.

o_perso_col : Objet destiné à représenter les personnages « barrières » sur les issues éventuelles du niveau. Il n'y a rien de particulier à signaler.

o_avatar : C'est le personnage manipulé par le joueur. Il est associé au sprite sprite_heros.

Event Create : on définit la vitesse d'animation du sprite à 0 pour qu'il apparaisse immobile à l'écran.

Event Step : on vérifie si l'avatar est proche d'un PNJ. Si c'est le cas, on envoie un Event User 6 au PNJ.

Event collision avec o_coll_inst : si on rentre en collision avec un élément de décor, on arrête le personnage.

Event collision avec o_perso_coll : idem

Event Intersect Boundary : idem

Event Up/Down/Left/Right : on augmente la coordonnée correspondante à la touche pressée et on affiche le sprite correspondant à la bonne direction.

Event Animation End : on arrête le personnage et l'animation de son sprite.

o_jauge2 : objet correspondant à la jauge de dépression. Il est associé au sprite sprite_jauge2. C'est grosso modo le même objet que la jauge du premier niveau. Elle est directement associée au nombre de poursuivants de l'avatar.

5.8 PROBLEMES RENCONTRES

La principale difficulté a été la gestion du temps. En effet, apprendre à utiliser un outil, bien que simple, et l'utiliser pour un projet ambitieux en peu de temps enrayera fortement la fluidité du travail. C'est toutefois une réalité à laquelle on finit toujours par faire face.

L'autre grande difficulté a été de gérer la gestion des instances Game Maker, et la façon dont celles-ci communiquent. Par ailleurs, tout est objet dans GML, ce qui ne facilite pas la tâche. Un autre souci lié aux instances : aucun contrôle sur l'ordre de leur création et de leur destruction. Si bien que les débuts/fins et transitions de rooms ont été difficiles à gérer.

Un problème qu'il n'a pas vraiment été possible de résoudre est celui-ci du ralentissement général du jeu (dû, sûrement, à un nombre trop important de ressources et animations à gérer en même temps).

5.9 AMELIORATIONS A APPORTER

Elles concernent bien sûr l'optimisation. Peu de temps a pu y être consacré. Par ailleurs, deux bugs persistent : les PNJ dans le deuxième niveau ne parviennent pas à suivre correctement l'avatar. Cela est dû probablement à une mauvaise utilisation de la fonction `move_towards()`.

L'utilisation des événements est peut-être trop massive et donc ralentit peut-être le jeu dans sa globalité.

Pas ou peu d'utilisation de l'héritage → peut être un plus.

Meilleure gestion du moteur FMOD pour le son.

6 ERGONOMIE

6.1 ERGONOMIE VIS-A-VIS DE CE PROJET

Un aspect particulièrement intéressant de la création de ce jeu du point de vue l'ergonomie est que l'*utilité* du jeu n'est pas ici (contrairement à la majorité des jeux) de provoquer le plaisir du joueur mais de lui faire passer un message. Cela dit, il a semblé plus pertinent de garder comme *utilité* au premier niveau le plaisir du joueur. L'*utilité* générale du jeu n'est visée alors qu'au début du deuxième niveau pour donner de la force au message en surprenant le joueur.

L'essentiel du travail s'est déroulé en deux phases. Une série de recommandation est énoncée lors de la conception du jeu. Puis lors, des béta-test, une liste d'ajustement est produite accompagnée d'éventuelles pistes à explorer.

6.2 RECOMMANDATION EN PHASE DE CONCEPTION

6.2.1 PREMIER NIVEAU

Les règles du jeu doivent être explicite, lorsque la phase de jeu commence le joueur sait immédiatement quoi faire.

Respecter les codes, la dialectique du jeu vidéo, lors du premier niveau. Le joueur doit se sentir en terrain connu. Il éprouve un plaisir basé sur la transgression relative à la propagation de la rumeur. On réutilise plusieurs éléments dans l'interface qui lui ramène

à des expériences ultérieures basées sur la performance (jauge, timer) pour lui faire croire qu'elle sera mesurée et que ses choix ont un impact sur le déroulement du jeu.

Le scrolling et le grouillement visuel sont des éléments qui vont volontairement faire baisser la lisibilité de l'action mais lui conférer du dynamisme.

Chaque action du joueur est soutenue par un feedback sonore et visuel explicite. Lorsqu'il convertit un personnage à la rumeur, une bulle apparaît au dessus de sa tête et un effet sonore se fait entendre pour signifier sa réussite.

6.2.2 DEUXIEME NIVEAU

Les nouvelles règles de gameplay doivent être introduites pendant la séquence de transition entre les deux niveaux. Lors du deuxième niveau, tous les artifices connotés « jeu vidéo » doivent avoir disparus.

La jauge est conservée car elle va permettre d'établir un lien entre les deux niveaux. Malgré le changement de gameplay radical, le joueur ne doit pas les considérer comme deux aventures différentes. Il doit établir un lien de cause à effet entre les deux niveaux.

On doit lui faire croire qu'il a une chance de remporter la partie. Mais il doit comprendre au final qu'il n'en avait en fait aucune, que ce n'est pas une erreur de sa part ou une erreur de paramétrage de la difficulté. Le message sera effectivement passé lorsque le joueur se sera rendu compte qu'il n'a aucune chance de gagner.

En ce qui concerne les feedback visuels et sonores, ils sont moins appuyés dans ce deuxième niveau car le personnage a moins d'influence sur ce qui l'entoure. Les feedbacks sonores se font par exemple, beaucoup moins « conquérant » et possèdent une connotation pathétique.

Le zoom est plus fort dans cette partie. Ceci afin de bien illustrer que l'on est sur un plan moins omniscient que dans la première partie. Il achève de faire comprendre que l'influence du joueur est réduite et rend explicite, par le gameplay, l'essentiel du message que l'on veut faire passer : on ne peut pas être plus fort que la rumeur, on ne peut pas se dérober à elle.

6.3 RECOMMANDATIONS APRES BETA-TEST

Après un premier test du jeu partiellement développé, on peut établir une liste d'éléments soulevant une interrogation sur la qualité effective des aspects analysés. A chacune de ces interrogations s'attache une ou plusieurs propositions d'ajustements. Ces nouvelles pistes, compte tenu des délais qui nous étaient impartis, seront explorées d'ici la présentation orale du 12 Décembre.

6.3.1 INTRO ANIME

Questions : Les termes tutoriaux sont ils suffisamment explicites ? Les informations nécessaires à la compréhension totale du gameplay sont elle exposées ?

- Parler de la jauge qui progresse.
- Parler du cercle d'influence du curseur.

Questions : Les messages narratifs et tutoriaux apparaissent ils suffisamment longtemps ?

- Ralentir les passages vidéo concernés pour faciliter la lecture.

6.3.2 PREMIER NIVEAU

Questions : Le joueur comprend-il la notion de « cercle d'influence » autour du curseur, arrive t'il à apprécier le diamètre de ce cercle ?

- Agrandir le viseur, le mettre de la taille du cercle d'influence.

Questions : Motive t'on le joueur à établir une performance ?

- Insérer un Timer.
- Proposer un nombre de clics limités.

Questions : Le joueur comprend il que la jauge se remplit à mesure que la rumeur se propage ?

- Rajouter un feedback sonore se rapportant à la jauge.
- Rajouter un feedback visuel (clignotement, mouvement).
- Présenter en début de partie la jauge de manière à ce que l'on comprenne qu'elle a un rôle à jouer (Iris, zoom, etc).
- Modifier le design de la jauge en rajoutant un indicateur de niveau.

Questions : La partie dure t'elle trop longtemps ? Pas assez ?

- Réduire/Augmenter le nombre de personnages cliquables.
- Ralentir/Accélérer le déplacement des personnages.

Questions : Le niveau ne se termine t'il pas trop brusquement ? Le joueur comprend il pourquoi est ce que le jeu s'arrête ?

- Faire apparaître un message dans l'écran de jeu.
- Eventuellement, accompagner le message d'un élément sonore
- Mettre en évidence la jauge pleine

6.3.3 SEQUENCE DE TRANSITION

Questions : Les nouvelles règles sont elle introduites explicitement ? Le joueur a-t-il conscience de ses nouvelles marges de manœuvres ?

- Modifier le texte en développant en détail les nouveau gameplay.
- Utiliser des images pour illustrer la transition de la rumeur vers l'avatar.
- Faire durer plus longtemps à l'écran les textes tutoriaux.

Questions : Le développement de l'intrigue est il compris ?

- Exposer plus longuement les étapes narratives.
- Présenter de nouvelles images.

6.3.4 DEUXIEME NIVEAU

Questions : Le joueur est il incommodé par le changement de gameplay ?

- Etablir un déplacement de l'avatar à la souris.

Questions : L'interaction entre les personnages non joueurs et l'avatar est elle explicite ? Le joueur comprend il ce qui est en train d'arriver à son avatar ?

- Etablir plus de feedback visuels et auditifs :
 - Sons des personnages plus agressifs.
 - Attitude des personnages moins passive.

Questions : Le joueur comprend il que la jauge se remplit à mesure qu'il rencontre des personnages?

- Rajouter un feedback sonore se rapportant à la jauge.
- Rajouter un feedback visuel (clignotement, mouvement).
- Présenter en début de partie la jauge de manière à ce que l'on comprenne quelle a un rôle à jouer (Iris, zoom, etc).
- Modifier le design de la jauge en rajoutant un indicateur de niveau.

Questions : Le niveau ne se termine t'il pas trop brusquement ? Le joueur comprend il pourquoi est ce que le jeu s'arrête ?

- Accompagner la fin de niveau d'un élément sonore.
- Mettre en évidence la jauge pleine.

6.3.5 SEQUENCE DE FIN

Questions : La séquence dure t'elle suffisamment longtemps ? Explique elle bien ce qui est arrivé à l'avatar ? Le joueur a-t-il le temps de comprendre le message que l'on veut lui faire passer ?

- Allonger la vidéo.
- Insérer de nouvelles images plus explicites.

7 CONCLUSION

Le résultat intermédiaire est cohérent. Charte sonore, charte graphique et game design se soutiennent les uns les autres. La version définitive devrait affiner le gameplay et renforcer l'ergonomie.

Il a été particulièrement intéressant de travailler sur un projet, aussi modeste soit il, qui met en avant un propos par son gameplay. C'est quelque chose qui ne se reproduira pas souvent dans la suite de nos carrières.