

Halo Fleet Battle : Digital Edition

Sommaire :

Chapitre 1/ : Généralités

- Introduction, avant-propos
- Halo, Fleet Battle
- Mise sous forme digitale.....

Chapitre 2/ : Le code

- Diagramme de classes
- Structure du code
- Fonctionnement du jeu
- Ergonomie et immersion

Chapitre 3/ : Conclusion et améliorations.....

- Figures imposées et ajoutées
- Evolutions possibles et limites
- Tutoriel de lancement de la démo

CHAPITRE 1

Introduction, avant-propos

Ce projet est un sujet libre que nous avons trouvé par nous-même. Sa complexité est supérieure aux autres sujets usuellement proposés. Néanmoins, c'est par intérêt personnel pour les jeux-vidéo et les jeux de stratégie que nous avons choisi ce projet.

La version rendue n'est pas une version complète du jeu, ce qui avait été annoncé dès le choix du sujet. Sur les quatre phases de jeu, deux sont jouables à ce jour. Nous sommes cependant quasiment certains que cette version, même partiellement fonctionnelle et non finale, répond aux critères de notation du projet.

Halo, Fleet Battle

Halo, Fleet Battle (HFB) est un **jeu de plateau** du type WarHammer. Les joueurs contrôlent des unités militaires qu'ils doivent déplacer sur un terrain et envoyer au combat contre l'adversaire afin d'anéantir ses forces ou pour remplir un objectif quelconque. Edité par Spartan Games, le jeu est basé sur l'univers de science-fiction Halo, reprenant ses vaisseaux, ses codes et son univers. Il n'est aujourd'hui plus produit et aucune version digitalisée n'existe.



Ce jeu de plateau se prête bien à la programmation orientée objet, car de nombreux jetons et pions sont utilisés dans les règles, et peuvent être modélisés par des classes et itérations de classe. De plus, les règles sont expliquées dans le manuel de façon itérative et proches du pseudo code, ce qui aidera lors de la création de la logique de jeu.

Mise sous forme digitale

Sur le principe, le jeu final serait un jeu au tour par tour, en vue de dessus (donc en 2D), ou les deux joueurs pourraient, chacun leur tour, manipuler leurs vaisseaux sur un même écran.

Les principales difficultés ont été les suivantes :

- Le grand nombre de classes à créer (au moment où ces lignes sont écrites, les fichiers contenant les classes cumulent environ 400 lignes de codes)
- La complexité du jeu, qui propose de nombreuses mécaniques, et qu'il faut retranscrire correctement
- La conception d'une IHM claire et efficace afin d'avoir une interface compréhensible mais complète du point de vue du joueur.
- L'intégration d'animations, la gestion de hitbox et des vecteurs.



Figure 2/ Groupe de vaisseaux



Figure 1/ Jeton d'escadron de bombardier

La modélisation choisie est exactement celle du jeu réel. Les deux seules simplifications admises à ce jour sont la suppression des mécaniques d'abordages de vaisseaux, compliquées et assez peu utilisées selon l'expérience que nous avons du jeu, et la suppression des commandants de flottes, car leurs effets sont difficiles à intégrer dans le déroulement d'un tour car ils peuvent intervenir, selon l'envie du joueur, à plein de moments différents.

Un tour typique de jeu se déroule selon 5 phases principales :

1/ Mouvement des escadrons : les joueurs déplacent leurs chasseurs et bombardiers sur le plateau, à tour de rôle

2/ Combats des escadrons : résolution automatique des combats entre escadrons.

3/ Mouvements de groupes de vaisseaux : les joueurs déplacent à tour de rôle leurs vaisseaux

4/ Combat entre vaisseaux : les joueurs déclarent leurs solutions de tir, qui sont résolues automatiquement une fois déclarées.

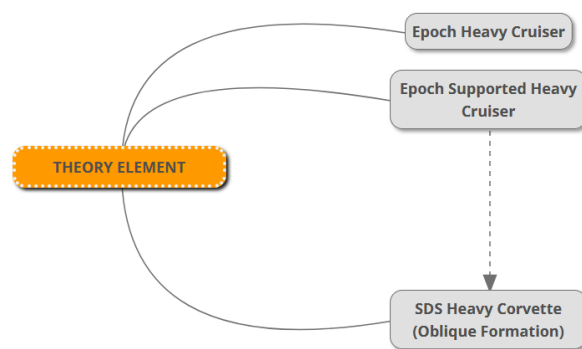
5 / Etapes de fin : gestion des dommages, réparation des vaisseaux et autres

Ces phases dicteront directement la structure même du code. Seule les phases de 1/ et 2/ sont aujourd'hui codées et jouables. Le squelette, les attributs et les fonctions utiles aux autres sont en grande partie implémentées.

CHAPITRE 2

Diagramme de classe

Le diagramme de classe du jeu n'est pas particulièrement complexe, le nombre d'étages étant très limité. En revanche, le nombre de ramifications est important. Par exemple



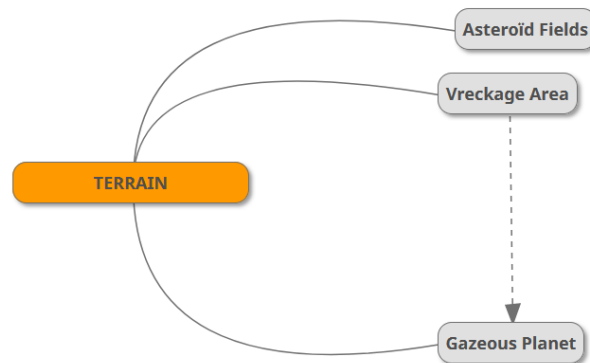
La classe TheoryElement est la superclasse sur laquelle se basent toutes les classes de vaisseaux. Elle contient toutes les fonctions utiles aux sous-classes. Les classes de vaisseaux filles ne contiennent aucune fonction supplémentaire : seules les caractéristiques propres au vaisseau sont précisées, et ses armes (ici la flèche en pointillé remplace la vingtaine de sous classes existantes à ce jour).

Il n'y aura donc plus qu'à instancier les classes de vaisseaux pour créer des pions.



Une classe, « ORS Supported Heavy Cruiser » fille de « Theory Element » représentera cet élément et regroupera toutes ses caractéristiques utiles au joueur et au déroulement du jeu

Les éléments du terrains (autre que le fond de carte) seront organisés de la même façon, avec une superclasse et une classe fille par élément de terrain différent. Le diagramme des terrains est donc similaire. Les terrains



Le reste des classes utilise assez peu le principe d'hérité. Les « loadouts » ou « équipements » sont un ensemble d'objet qui changeront le comportement du jeu si un vaisseau les possède. Chacun est représenté par une classe, souvent quasiment vide.

De manière globale, les autres classes présentes sont : Weapon, Player, Group, Aircraft, Tokens...

Structure du code

Les fichiers GameLogic et main contiennent le déroulement du jeu et l'ensemble des fonctions nécessaires au déroulement de la partie au fil du temps.

Les fichiers units, weapons, loadouts contiennent l'ensemble des classes d'objets que le joueur peut manipuler ou dont le jeu a besoin comme les vaisseaux, les armes et les équipements de vaisseaux.

Le fichier misc contient toutes les fonctions utiles aux calculs de distances et de jets de dés, et le fichier config des variables globales concernant en général l'échelle des objets à afficher.

Fonctionnement du jeu

La librairie Panda3D a été utilisée pour son moteur *render2D* et *render*, qui est un moteur 3D. Le plateau de jeu est en 3D, mais projeté sur un plan 2D pour la

caméra. Celle-ci se déplace dans un plan différent du plateau. Le HUD est ajouté comme un filtre en 2D par-dessus le rendu de *render*.

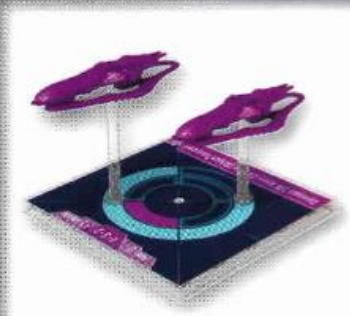
Le jeu se présente en plusieurs phases gérées par la GUI. Chaque action de l'utilisateur peu provoquer un évènement automatique, selon les règles du jeu. En fonction de l'action faite par le joueur, des évènements automatiques se déclenchent, tels que le lancement d'une animation, le passage à la phase suivante, ou la résolution d'un combat.

Les joueurs sont limités dans les actions qu'ils peuvent entreprendre, selon si c'est leur tour de jouer ou non par exemple.

Parmi les mécaniques notoires du code permettant de surmonter la complexité des règles, voici quelques exemples.

- Le problème des loadouts :

Les loadouts modélisent des équipements. Chaque vaisseau, arme ou escadron possède une certaine liste de loadouts.

Covenant SDV Heavy Corvette (Oblique Formation)					Small Shins (40 PTS)				
					Systems Loadouts				
					Cloaking System				
					Defence Array (2)				
					Elusive				
					Glide (5")				
					Point Defence (3)				
Movement		9"							
Damage Track		4•3•3							
Build Rating		1							
Hangars		1							
Boarding Craft		0							
Security Detail		2							
Primary Weapon		Range	Weapon Loadouts		Arc	Dice			
Plasma Cannon Arrays		10/20"	Plasma		F/P/S	4			
Secondary Weapon		Range	Weapon Loadouts		Arc	Dice			
Plasma Torpedoes		12/24"	Missiles/Plasma		F	3			

Le problème étant que les loadouts ont des effets qui interviennent à différents stades du jeu. Certains pendant le déplacement des vaisseaux, d'autres pendant le combat, d'autre avant de subir des dégats etc etc...

Pour pallier à ce problème et éviter d'implémenter une classe différente pour chaque type de loadout, nous avons implémenté un certain nombre de fonctions dans la superclasse loadout.

A chaque fois que nous avons besoin qu'une fonction spécifique à un loadout intervienne dans une phase, nous rajoutions dans la superclasse cette même fonction, mais vide ou renvoyant zéro. On peut alors, à chaque fois qu'un loadout est censé intervenir, parcourir l'ensemble des loadouts du vaisseau et appeler la fonction correspondante. Elle ne modifie le déroulement du jeu que si la fonction a été overload spécifiquement.

- Affichage d'informations détaillées :

Lorsque le joueur veut avoir plus d'informations sur un vaisseau, il clique dessus. Il faut alors afficher sur l'HUD les informations voulues, et donc le modifier. Pour cela, le code crée, au moment où l'utilisateur clique sur le vaisseau, un nouvel élément d'HUD qui contient les informations voulues et qui se superpose au HUD actuel.

Quand le joueur désélectionne l'unité, le nouvel HUD est supprimé du point de vue du jeu. La fonction `del` a été surchargée pour supprimer toutes les occurrences de cet élément d'HUD dans le code.

Ergonomie et immersion

Toutes les informations utiles sont disponibles pour les joueurs. L'ergonomie est classique pour un jeu vidéo du genre : affichage d'informations supplémentaires si unité sélectionnée, annuler la sélection avec un clic droit etc...

Des éléments « d'immersion » ont été ajoutés au jeu, afin de commencer à approcher un vrai jeu vidéo. A savoir, des éléments contingents au fonctionnement du jeu, mais agréable pour l'utilisateur.

D'un point de vue graphique, les boutons du menu sont animés, et les autres devraient l'être sous peu ; à savoir, changement de la couleur du bouton quand la souris passe dessus.

Les jetons utilisés sont des « skins » de vaisseaux de l'univers trouvés sur internet. Des modèles 3D ont été screenshot en vue de dessus, puis détournés pour obtenir un jeton 2D. Chaque jeton correspond donc au vaisseau associé dans l'univers canonique (ce qui est, en tant que fan, très important).

Des animations simples d'attaque ont été implémentées, ainsi que des explosions. Les explosions ne surviennent que sur l'élément détruit lors du

combat. En effets, les animations se déclenchent une fois tous les calculs de combats effectués.

Un début de sound-design a été implémenté. Des bruitages sont lancés lors de passages de souris sur les boutons, et deux musiques différentes se lancent : une dans le menu (Le thème principal de la saga Halo) et l'autre dans le jeu (un mix d'une heure de musique des jeux de l'univers). Les animations sont aussi accompagnées de bruitages, et lorsque le joueur actif sélectionne l'un de ses vaisseaux, l'un des 8 doublages associé à sa faction (UNSC ou Covenant) se lance aléatoirement. Ces doublages ont été enregistrés puis modifiés par nos soins.

L'ensemble des éléments d'ergonomie est disponible dans le fichier « Assets »

CHAPITRE 3

7/ Figures imposées réalisées et ajouts supplémentaires

Figures imposées réalisées :

- Hérité de classes, avec les classes abstraite SpaceCraft, TheoryElement et Loadouts
- Gestion de vecteurs avec la gestion de la caméra, la projection d'animations et le calcul d'angles de tir.
- Lecture/Ecriture de fichiers : Avec l'utilisation de fichier audio, vidéo, et img pour l'immersion, ainsi que l'écriture d'un fichier de sauvegarde
- Tests unitaires : tests du bon déroulement de l'instanciation des classes.

Figures supplémentaires :

- Intégration d'animations évolutives en fonction de la situation, les directions des tirs s'adaptant au positionnement actuel des vaisseaux.
- Gestion de caméra dynamique, avec le zoom et déplacement.
- Mouse tracking

8/ Evolutions possibles et limites

Le temps imparti ne nous a bien sûr pas permis de finir totalement le jeu, mais nous comptons le faire petit à petit. Nous avons pris un réel plaisir à coder ce jeu, malgré les difficultés.

Nous avons été, et serons à l'avenir, limité dans notre choix de jetons, d'images et d'animations, car nous n'avons payé aucun fichier pour nos assets. Cela est notamment très limitant pour les animations, qui sont très difficiles à trouver gratuitement. Il faudra peut-être à l'avenir les faire soi-même.

Enfin l'ergonomie doit être améliorée, par exemple la police d'écriture n'est pas assez lisible actuellement.

9/ Tutoriel

1/ Exécutez le fichier « main ». Dans la fenêtre, ouverte, prenez le temps d'écouter la magnifique musique d'Halo.

2/ Ecoutez encore un peu :D

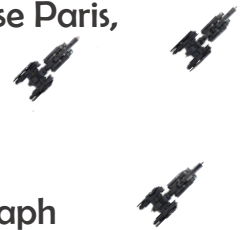
3 / Cliquez sur « New Game »

4/ Vous voilà en partie. En haut, au milieu un peu à droite s'affiche le joueur actif. « UNSC » est la faction humaine. Leurs vaisseaux sont gris. Les Covenantes sont la secte alien. Leurs vaisseaux sont violets. Dans cette démo, vous ne pourrez contrôler que les escadrons (les vaisseaux les plus petits). Vous allez jouer, pour ce tutoriel, les deux factions. Un peu comme si vous jouiez les blancs et les noirs aux échecs. Les UNSC commencent.

5/


UNSC :

Bonne nouvelle, vous possédez deux formations de Frégates de classe Paris, mais aussi un Croiseur lourd de classe Epoch que vous devriez repérer assez vite sur votre écran.



La mauvaise nouvelle, c'est que trois escadrons de Bombardiers Séraph semblent beaucoup trop proche de vos frégates... Pas de panique. Votre croiseur lourd va pouvoir envoyer des chasseurs les intercepter, et une patrouille est déjà sur place.

Sélectionnez votre Epoch en cliquant dessus. Les informations sur le vaisseau s'affichent. Un menu apparaît à côté. Cliquez sur les deux flèches vers le haut.


Dans le menu qui s'affiche, sélectionnez votre escadron d'intercepteurs Broadsword . Un cercle vert et un cercle rouge s'affichent. Le cercle vert indique la distance maximale que vos Broadsword peuvent parcourir. Le rouge est la distance d'engagement. Si un ennemi se trouve dans le cercle rouge, votre escadron s'engagera dans un combat avec lui.

Déployez vos Broadwords pour intercepter le 3^e Séraph  le plus haut. L'ennemi doit se trouver dans la zone rouge de votre vaisseau. Vos chasseurs auront largement l'avantage contre des bombardiers et devraient s'en sortir.

Covenant :

Votre flotte se compose de deux Destroyer de classe CCS, dont l'un est supporté par une Corvette Lourde SDV.



Votre but est d'aller réduire en débris l'une de ces frégates. Le Prophète ne tolérera pas qu'elles reparte entière ! Déplacez un bombardier Séraph  pour attaquer une frégate. Attention ! L'unité interceptée par les chasseurs ne peut plus bouger, prenez en une autre ! De plus, le centre du triangle de formation des frégates doit se trouver dans votre cercle d'engagement.

UNSC :

L'un des bombardiers a atteint sa cible, mais pas le temps de s'apitoyer. Interceptez l'autre avec le Broadsword encore en patrouille !

Covenant :

Cette Epoch est une véritable hérésie, envoyez-y votre bombardier le plus en bas !

UNSC :

C'est l'heure de la contre-attaque ! Déployez le bombardier Longsword depuis votre Epoch !

Covenant :

Bombardier en approche de votre Destroyer ! Envoyez un escadron d'intercepteurs Banshee  pour leur régler leur compte !

Cliquez ensuite sur « Next Phase » deux fois, les animations de combat devraient se lancer. En fonction des résultats des jets de dés (pas visibles encore par le joueur...) certains unités devraient subir des dégâts, voire être détruites.