Modélisation du jeu de GO

Un jeu ancestral de stratégie dans l'ére numérique

Rapport de projet

Rédigé par : *Raphael CAMIZULI*

Rédigé le *01/03/15*

Réf. du document : rapport programmation Go CAMIZULI Raphael

SOMMAIRE

1 RÉSUME DU PROJET

Le sujet du projet est la réalisation d'un jeu originaire de Chine que l'on appelle « GO » , qu'il a fallu réaliser en langage de programmation C.

pour la réalisation de ce projet nous avons eu comme consigne d'utiliser des pointeurs , de décomposer notre programme sous forme de plusieurs fonction avec un nom qui lui est propre, chaque fonction devait avoir un unique tâche.

Le projet devait se faire en binôme sur une période d'environ deux semaines et doit être accompagné d'un rapport détaillant le projet et son fonctionnement ainsi que les différents problèmes rencontrés

Pour réaliser ce projet nous avons eu le choix de faire de la SDL, programme de programmation servant à faire de petites animations en 2D.

2 HISTORIQUE DU JEU DU GO

2.1 HISTOIRE

Le jeu de go est originaire de Chine. Il s'agit du plus ancien jeu de stratégie combinatoire connu. Malgré son ancienneté, le jeu de go continue à jouir d'une grande popularité en Chine, en Corée et au Japon. Dans le reste du monde sa découverte est récente et sa notoriété croissante. Son succès tient autant à la simplicité de ses règles qu'à sa grande richesse combinatoire et sa profondeur stratégique.

Bien que le jeu de go soit un jeu chinois, la terminologie utilisée en Occident est principalement d'origine japonaise.

2.2 PRINCIPE ET REGLE DU JEU

Le but de ce jeu est de capturer les pions de son adversaire. C'est un jeu qui se joue uniquement à 2. Pour jouer à se jeu il faut damier de 9x9 ou 19x19 appelé le goban, sur lequel est tracé une grille de 19 lignes horizontales par 19 lignes verticales, qui déterminent 361 intersections, mais ce nombre de lignes est parfois réduit (souvent ramené à 13 × 13 ou 9 × 9 lignes) pour jouer des parties rapides ou pour faciliter l'apprentissage des règles du jeu.

Chaque joueur tente de contrôler le plan de jeu en y construisant des «territoires» qui se comptent en points. Les pions d'un joueur ne peuvent que être place sur les intersections de ligne et des colonnes.

Pour capturer les pions adverses le joueur doit encercle un ou les pions adversaire en tout en étant en contact avec eux et forme une boucle afin de capture tout les pions adversaire a l'intérieur de celle-ci.

À la fin de la partie, il reste des pierres qui sont impossibles à capturer et qui ainsi délimitent des territoires. On compte un point par intersection libre dans chaque territoire et un point pour chaque prisonnier (pierre prise ou morte) capturé . Le vainqueur est celui qui obtient le plus de points. Dans notre cas nous conterons les pions capture.

3 ANALYSE FONCTIONNELLE GÉNÉRALE

3.1 PRÉSENTATION DU JEU

3.1.1Matériel

Un plateau constitué de 9 lignes et 9 colonnes

Deux lots de pions (ou pierres), en général des noirs et des blancs

Une partie du programme sera dédié à la définition du plateau. Dans le jeu sur plateau de bois, les pions sont posés sur les intersections des lignes. Dans le programme, ces intersections seront remplacées par des cases (81 cases dans le plateau choisi).

Une autre partie du programme garera les pions en terme de couleur, puis de changement de couleur en cas de capture.

Si dans le jeu réel, le nombre de pions est fini (mais largement supérieur au besoin), dans la version virtuelle, ce paramètre ne sera pas pris en compte.

3.1.2But du jeu

Dans cette partie, le programme gérera les participants (2 joueurs, ou 1 joueur contre la machine).

L'essentiel du jeu est de positionner de manière stratégique des pions le plateau. Un ensemble de pions de même couleur constituent un territoire. Si un pion se retrouve encadré par des pions adverses, il est capturé et par conséquent doit changer de couleur.

Une partie du programme devra donc gérer le positionnement de chaque pion. Une série de test devra valider si le pion doit être comme prisonniers ou non. Si le pion est prisonnier, il devra changer de couleur.

DÉBUT DE PARTIE

Au départ, le damier est vide et le joueur qui possède les pions noir commence.

COURS DU JEU

Chacun leur tour, les joueurs posent un pion où ils le souhaitent sur la surface de jeu. Par contre il ne peuvent ni l'enlever ni le déplacer losrqu'il est déjà en place.il n'est pas possible non plus de poser un pion sur une place occupée.

Le programme doit s'articuler sur une bascule (passage d'un joueur à un autre), tout en validant le coup joué. La validation prendra surtout en compte l'occupation de la case. Si le pion posé est sur une case occupe, un message indiquera que le coup est invalide. Le programme invitera le joueur à refaire une proposition.

CAPTURE DES PIONS

Pour capturer un ou des pions, il faut que ceux ci soient encerclés par les pions adverses. Les pions doivent tous être en contact direct pour la capture (aucune case vide dans la limite du territoire pour les assaillants, ni dans le domaine à défendre pour les prisonniers.

FIN DE PARTIE

La partie se termine suivant un des deux critères :

il n'y a plus d'espace pour jouer (poser un pion)

un des joueurs passe deux fois son tour (cela veut dire qu'il ne peut vraiment plus jouer)

AUTRE BESOIN

Certaine des ces particularités seront proposée et effectuées en debut de programme.

Ce jeu se joue avec uniquement deux joueurs. Cependant la technologie nous permet de pouvoir jouer contre une machine (IA; Intelligence Artificielle). Nous conserverons cette possibilité en proposant l'option en début de programme.

En terme de besoin et de convivialité un jeu, nous proposerons aussi aux joueurs de choisir la couleur des pions. Cette option déterminera le joueur qui débutera la partie (dans ce jeu, ce sera le joueur qui possède les pions noirs.

Il sera aussi possible de proposer à l'un des joueurs un handicap. Cela consistera à placer sur la table un certain nombre de pions pour des joueurs. Il en choisira lui-même le nombre ainsi que la position sur la table. La fin de cette option sera commandée par le joueur lui-même par une commande à définir.

D'une manière générale, ce type de jeu demande énormément de temps pour une partie. Il est possible que le où les joueurs que ne puissent terminer leur session. En termes d'option, il y aura donc aussi la possibilité de sauvegarder une partie qui n'a pu aller à son terme, ainsi que la possibilité de la recharger afin de pouvoir la terminer. Les paramètres de la partie seront stockés dans un fichier qui sera rappelé le cas échéant.

Dans ce programme, à chaque fois qu'une saisie sera nécessaire, l'utilisateur se verra proposer les réponses attendues. Les saisies seront comparées aux propositions. La question sera reposée à l'utilisateur jusqu'à ce qu'il entre une réponse conforme aux attentes. L'e1ncadrement de la saisie permet d'éviter des problèmes dans le déroulement du programme.

4 ANALYSE DÉTAILLÉE

4.1 STRUCTURATION DU PROGRAMME

Présentation des blocs de programme :

Définition de la table de jeu	Gestion des pierres retirées
Gestion de la sauvegarde et des parties sauvegardées	Calcul du score
Gestion des parties sauvegardées	Gestion du handicap
Calcul des bonus	Gestion du jeu (position du pion et gestion des degrés de liberté
Test de validité du coup joué	Gestion générale du jeu

Chaque block a une fonction spécifique pour le programme. Le lien essentiel entre ce blocks se fera par la partie « gestion générale du jeu »

4.2 DÉTAIL PAR BOCK

4.2.1Définition de la table de jeu

Définition de la table de jeu de 9x9 cases. Les numéros des lignes et colonnes seront affichés pour faciliter le repérage visuel. Ces affectations serviront aussi pour les différents calculs.

4.2.2Gestion de pierre retirées

Ce bloc permet aussi l'évolution du jeu. C'est la partie où l'on retire les pierres dites « mortes »

Après une validation de la demande de retrait des pierres, le joueur devra choisir la pierre a retirer. Pour cela il saisira le numéro de la case qu'i désire jouer.

Le programme vérifiera si la saisie de la case est correcte.Le joueur donc pourra arrêter de mettre fin aux retrait des pierres par saisie de la commande proposée.

A noter, que pour chaque saisies, la proposition de phase de jeu sera affichée ainsi que les propositions de réponses. Les entrées seront testées, afin de valider l'action en cours. Cette procédure sera valable que tout au long du programme.

4.2.3Gestion du handicap

Conformément à la règle, la gestion du handicap le consiste à placer entre une et neuf pierre sur le plateau. Un compteur sera mis en place avec pour 9 pierres à poser. Le joueur pourra aussi volontairement ne pas utiliser le nombre maximum le handicap par une simple commande. Comme précisé dans la règle, la pose de ces pierres est destinée à favoriser le joueur qui commence (donc celui qui possède les pions noirs).

A la fin de cette séquence, et après le choix du nombre de joueurs, la table s'affiche avec un les pions est posé pour le handicap

4.2.4Calcul du score

Dans la règle du jeu, il est établi que le joueur en possédant les pions blancs, aura un bonus de points. A l'inverse, l'adversaire un score vierge. Au début de chaque partie, les variables seront formatées en conséquence.

4.2.5Gestion des parties sauvegardées

Le programme fera appel à un fichier formaté en conséquence (fichier texte). Un certain nombre de partie seront sauvegardées. Il suffira de rappeler la partie par son numéro de sauvegarde.

Dans le cas où la partie n'existe pas ou tout autre anomalie, le programme affichera un message d'erreur. Le programme reprendra son cours en proposant l'option du nombre de joueur.

4.2.6Gestion de la sauvegarde et des parties sauvegardées

La structure de cette partie est sensiblement la même que dans le block de programme précèdent.

Le nom de la variable de stockage change.

4.2.7Calcul des bonus

Cette phase s'articule en deux parties ;

La table des bonus est paramétrée dans le programme. Il y a 4 types de bonus. La définition de la variable prend en compte la condition d'affectation.

Par exemple ; int Bonus0Liberte = 200, correspond à la situation où il n'y a plus de degré de liberté. Le programme testera s'il reste des possibilité de mouvement et affectera les points en conséquence.

4.2.8Gestion du jeu et des positions des pions(avancement des pions et des prisonniers)

Ce block est un des plus longs du programme.

Cette partie est aussi l'essentiel du programme. Il est géré :

- la validité du coup joué
- l'identité du joueur en lice et la pile de pion sélectionne
- la longueur du territoire
- le nombre de case à occupe
- un signal en cas de coup non valide
- le changement de joueur dès que le coup es validé et joué
-

Tout d'abord, il faut définir des variables que l'on peut dire internes, dans la mesure où aucunes ne seront affichées. Elle ne serviront que pour les calculs et vérification.

La suite de jeu au travers du programme se déroulera par étape :

Étape 1

Le programme est conçu pour le vérifier si un joueur peut placer un pion dans une case. Au fil des coups joués, le programme vérifiera si le joueur place un pion sur une case occupée.

Combiné à divers tests logiques, le programme gère les territoires de chaque joueur à ce que la continuité des territoires

Étape 2

A cette étape, la programmation valide que le joueur en lice arrive bien sur ;

- une case vide
- une case occupée par l'adversaire
-

La structure de test est similaire à l'étape précédente. La séquence est plus courte car certaines données sont issues des tests précédents.

Étape 3

Dans la mesure où le choix du joueur respecte les règles, le coup sera joué (validé) et la partie continuera

Quand un joueur capture un pion, celui –ci est sorti du jeu.

Le programme comptabilise les points pour chaque coup joué et les attribue à chaque joueur.

4.2.9Validité des coups joués

Cette partie du programme gère la validité du coup joué. Il est aussi évalué dans cette partie du programme la valeur du coup joué.

La validité du coup joué est conditionnée par une série de tests logiques.

Dans cette partie, le programme mets à l'affichage du tableau

4.2.10 Gestion générale du jeu

Cette partie de programme gère la représentation et l'affichage du jeu.

Dans un premier temps le programme proposera de continuer une partie sauvegardée. Si l'utilisateur ne souhaite pas que reprendre une partie déjà entamée, on recommence un processus.

Dans un premier temps le programme proposera l'option de jeu ;

Deux joueurs

Jeu contre l'ordinateur

Dans un second temps, on joueurs choisira ou non s'il a un handicap. Puis il placera un maximum de neuf pions pour le handicap. Comme nous l'avons précisé dans un paragraphe précédent,

Les pions de handicap seront noirs et par conséquent le joueur possédant les noirs, commencera.

La question suivante portera sur le choix de la couleur des pions par l'un des joueurs.

Outre l'affichage du plateau de jeu, cette partie de programme gèrera les autres d'affichage. Par exemple, il est possible de citer le joueur en lice, l'avertissement d'une réponse erronée.

Cette partie soigne l'interface homme machine en facilitant le mode de lecture à l'écran, la saisie simplifiée et orientée (les réponses sont suggérées et comparées).

5 SDL

5.1 QU'EST CE QUE LA SDL (SIMPLE DIRECTMEDIA LAYER)

Simple DirectMedia Layer est une bibliothèque multiplate-forme de développement conçu pour fournir un accès au son, clavier, souris, joystick, et des graphiques matériel via OpenGL et Direct3D.

5.2 UTILISATION DE LA SDL

Pour ce projet nous avions la possibilité d'utiliser la SDL pour réaliser ce projet, même si nous aurions pu faire sans.

Pour des raisons pratiques et de lisibilité nous avons tenue à utiliser la SDL.En utilisant cet outil, notre objectif est de rendre le programme plus convivial. En effet, il est possible de faire apparaître une fenêtre avec la table, est de réaliser l'acquisition par l'intermédiaire d'une souris plutôt que le clavier.

Le principal problème était le manque de maîtrise de cet outil. L'apprentissage de l'utilisation de la SDL s'est fait grâce à divers tutoriels.

Pour maîtriser la SDL, il est nécessaire d'avoir une connaissance approfondie du langage C

5.3 FONCTION SDL

Il est donc possible de pouvoir gérer via cette interface des fenêtres, des sons, des acquisitions.

Dans notre programme il existe plusieurs fonctions en rapport avec le type SDL.

Par exemple il a une fonction « SDL_.. » qui va servir à à l'affichage d'une fenêtre. Le complément SDL permettra de gérer la taille de cette fenêtre.

En utilisant cette fonction, nous comptons ouvrir une fenêtre montrant la table de jeu. Dans la mesure du possible nous souhaitons utiliser cette fenêtre comme table de jeu. Par conséquent elle servira aussi à la pose des pions de joueurs.

Pour rendre le jeu encore plus convivial, il serait intéressant avec des sons. Par exemple nous pouvons sonoriser une entrée erronée par un son de klaxon (allusion a un jeu radiophonique). De même que chaque action toute être illustré par un son.

5.4 MISE EN PRATIQUE

Malgré une étude approfondie des divers tutoriaux, l'intégration de la SDL et la mise en pratique n'a pas permis de réaliser un programme efficace.

C'est pour cela, que la totalité de la programmation se fera en langage C. Au niveau de la restitution, nous nous contenterons d'une fenêtre noire et d'une table de jeu avec des intersections visibles.

6 CONCLUSION

Nous avons terminé notre programme et vérifié sont bon fonctionnement qui répond à nos attentes excepté la fonction servant a enlever les points automatique que nous avons remplacé par une fonction permettant de les enlever manuellement en les ôtant un par un.

Nous avons renoncé à utiliser la SDL car malgré le fait que cela permette une meilleur représentation du jeu car cela demandait beaucoup de calculs et demandait trop de temps.

La volonté de réaliser le programme en 5 grandes fonctions (architecture du programme) permet de traiter sans interaction les demandes spécifiques du jeu.

Chaque partie est conçus de manière la plus courte possible afin de conserver une vitesse d'exécution optimale. Dans ce programme « simple », la différence ne sera pas vraiment perceptible.

Le découpage permets aussi, lors de test d'identifier les manquement au fonctionnement du jeu voire au règle du jeu.

La volonté d'avoir un programme ne permet probablement pas de gérer de manière exhaustive tous les cas possible d'incident.

7 ANNEXE

Règle du jeu §3 source Wikipédia

http://fr.wikipedia.org/wiki/Puluc

Site du zéro