Projet long de Technologie Objet

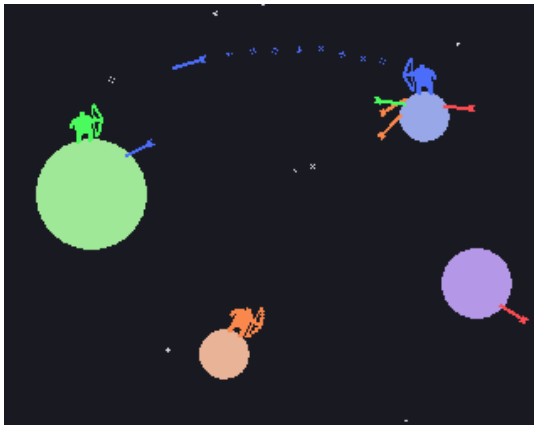
Liste des sujets envisagés

Fainsin Laurent
Guillemin Johan
Guillotin Damien
Heurtebise Tom
Jourdan Pierre-eliot
Kirupananthan Nadesan

Département Sciences du Numérique
Première année
2020 — 2021

Table des matières

1 Sagittarius



Nous envisageons de créer un programme permettant de jouer à Sagittarius [1], un jeu tour par tour donc l'unique but est de tirer avec un arc et des flêches sur les autres joueurs et où le mécanisme principal est la gravité !

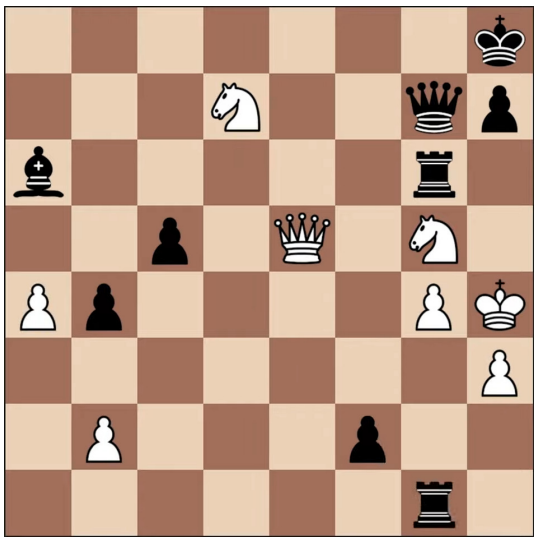
1.1 Objectifs et intérêts

L'objectif principal est d'obtenir un jeu jouable et qui donne envie aux utilisateur de jouer. Comme la mécanique de ce jeu s'appuie fortement sur la gravité, il nous faudra trouver un moyen de simuler celle-ci. Pour calculer la trajectoire d'une flêche dans un système à N corps [2], nous avons le choix de le faire en temps réel (i.e. à chaque nouvelle image [3]), ou bien par calcul la trajectoire entière de la flêche (via RK4 [4] par exemple). L'intérêt de choisir un jeu est que cela nécéssite généralement la bonne compréhension et utilisation de la technologie des objets. Il est assez facile de voir que nous aurons besoin d'objets pour les joueurs, les planètes, les flèches, les particules...

1.2 Difficultés

L'obstacle majeur à la réalisation de ce jeu est l'outil graphique que nous allons devoir utiliser. En effet, nous apprendrons à utiliser Swing lors du dernier cours de Technologie Objet, cependant il est peu probable que cet environnement graphique soit adapté à ce projet. Il nous faudra alors trouver une alternative, cela implique qu'une partie du dévelopement soit alloué à l'apprentissage (basique) d'une librairie.

2 Le jeu des échecs



Ce projet vise à créer un programme de jeu d'échecs classique. Deux joueurs s'affrontent selon deux types de parties possibles : les parties dites "rapides" (où un temps sera fixé au début de la partie et le premier joueur le dépassant perdra) ou des parties dites "longues" (où il n'y aura pas de limite de temps global mais où on pourra définir une limite de temps entre chaque coup). Le joueur qui commence est celui qui possède les pièces blanches, puis c'est au tour du joueur possédant les pièces noires etc...

2.1 But du projet et intérêts

Il sera d'abord nécessaire de définir la manière la plus efficace et la moins coûteuse avec laquelle nous représenterons la table de jeu (à savoir une grille carrée de 8 cases de côté soit 64 cases). Il est possible d'utiliser un tableau, des listes chaînées...

L'utilisation de la technologie objet sera utile pour manipuler les différentes pièces du jeu (pions(8 x2), tours(2 x2), fous(2 x2), cavaliers(2 x2), roi (x2), reine (x2) ...) en définissant par exemple des classes pour chaque catégorie de pièces et même des héritages (on peut imaginer que la classe Reine hérite de la classe Roi).

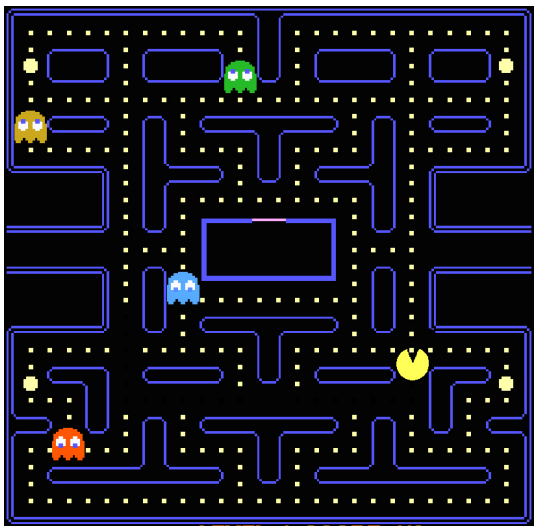
Il nous faudra aussi définir les différents déplacements et coups existants (roque...) ainsi que les issues possibles (echec et mat, pat...).

Les possibilités de mouvement de chaque joueur devront être calculées à chaque tour en fonction de la pièce considéré et de sa position sur le plateau, c'est pourquoi il est nécessaire de mettre à jour la position de chaque pion à chaque étape du jeu en lui affectant, par exemple, des coordonnées, dans un plan qu'on définirait par défaut. Seront aussi à définir par défaut le type de jeu et la durée. Enfin l'attribution de la couleur des pièces (blanc ou noir) sera déterminée arbitrairement avec un tirage au sort au début du programme.

2.2 Difficultés

Le risque dans la programmation d'un jeu d'échecs est le coût de chaque opération (calcul de la trajectoire et des coups) qui risque de rendre un peu long l'exécution du programme et pouvant donner une complexité très grande si elles ne sont pas optimisées.
Enfin il y a toujours une difficulté concernant l'interface graphique que nous devrons utiliser pour modéliser le jeu (avec éventuellement, si le temps le permet, la création d'un chat ou autre interface interactive pour que les deux joueurs puissent communiquer entre eux).

3 Pac-Man



Pac-Man est un jeu d'arcade ou le but du joueur est de récupérer tous les points situés sur la carte avant de se faire attraper par les fantômes. Le premier objectif de ce projet serait de créer le jeu avec ses principales fonctionnalités. Un second objectif pourra être de permettre à un utilisateur de créer un bot qui remplacera les contrôles l'utilisateur.

3.1 Règles

Pac-Man peut se déplacer comme bon lui semble sur la carte sans sortir des chemins. Les quatre fantômes peuvent également se déplacer sur toute la carte mais ils n'ont pas le droit de faire de demi-tours. Ils ont chacun leurs propres méthodes d'attaques.

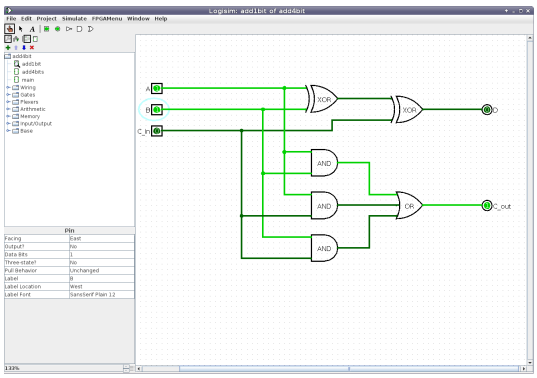
Blinky (rouge) est le fantôme qui poursuit simplement Pac-Man. Pinky (rose) cible quatre unités de distance devant Pac-Man. Inky (cyan) cible la localisation opposée à Blinky par rapport à celle de Pac-Man. Le dernier, Clyde (orange) cible Pac-Man mais lorsqu'il est trop près de ce dernier, il fuit vers son coin.

Sur la carte, y sont disposé quatre super Pac-Gomme. Lorsque Pac-Man en mange une, les fantômes passent en mode panique. Ce mode panique les fait prendre des directions aléatoires pendant quelques secondes ou jusqu'à ce qu'ils se fassent manger par Pac-Man. Dans ce cas, il retourne à leur zone d'apparition pour finalement réattaquer Pac-Man.

3.2 Intérêts

Dans ce projet, il va falloir générer le plateau de jeu et veiller à ce que les différentes entités respect bien les la forme du plateau. Il faudra créer deux types d'entités mais aussi des sous-types pour les fantômes. La création d'un algorithme de path-finding sera nécessaire pour générer les déplacements des fantômes.

4 Simulateur graphique de circuits logiques



Ce projet vise à créer un programme permettant de simuler un circuit logique dont la construction se ferait graphiquement. Dans le style de Logisim [5] l'utilisateur aurait à sa disposition une boite à outil pour poser des portes logiques et un outil pour tracer les liaisons.

4.1 Objectifs et intérêts

L'utilisation de la technologie objet se prête bien à ce projet pour définir et utiliser les différentes portes logiques. Par exemple, chaque porte aura plusieurs entrée et sorties, celles-ci pourrons être représentées en code par des attributs associés à des objets "liaison".

4.2 Difficultés

La première difficulté provient de l'interface graphique, puisque l'utilisateur doit avoir la possibilité de placer sur un plan de travail n'importe quel type de portes logiques (drag and drop). De même il devra pouvoir piloter des entrées manuelement, mais aussi pouvoir paramétrer des horloges, de la RAM/ROM...

5 Outil d'aide à la prise de décision dans le cadre de l'épidémie de COVID-19

Le but d'un tel outil serait de générer, à partir de données actuelles sur l'épidémie de coronavirus (extraites sous la forme d'un fichier json), un rapport concis synthétisant ces données et étudiant plusieurs scénarios possibles. Il pourrait alors être intéressant d'envisager la formulation d'un conseil sur les mesures à prendre parmi un panel d'options (couvre-feu, aucune mesure ou encore confinement strict) .

5.1 Jalons du projet

Il est à noter qu'un tel projet nécessite des connaissances importantes en modélisation ainsi qu'en épidémiologie et peut paraître très ambitieux. Une première phase sera alors dédiée à la recherche. Une deuxième phase sera la définition du modèle à employer avec une première version utilisant des hypothèses simplificatrices. Ainsi nous envisagerons par exemple le fait que deux individus se rencontrent de façon aléatoire (alors qu'un individu choisit évidemment ses fréquentations), le fait que les distanciations sociales sont parfaitement respectées ou encore la non prise en compte des variants du virus existants. D'autres hypothèses pourront être formulées par la suite. Une troisième phase du projet sera de supprimer les hypothèses formulées pour avoir un modèle plus proche de la réalité.

5.2 Description du modèle

Afin que notre modèle soit le plus représentatif possible de la réalité nous pourrons adopter des modèles déjà existants comme celui du SIR (Suspectible Infectious Recovered) dans lequel la population est divisée en 3 catégories. La première peut être contaminée, la deuxième est infectée et la troisième est guérie. Ce modèle prend aussi en compte le taux de propagation du virus appelé Radius, plus celui-ci est élevé et plus un individu a de chances de transmettre le virus à une autre personne. Pour un modèle plus complexe il faudra considérer le cas des personnes asymptomatiques pour faire une quatrième catégorie au sein de la population et prendre en compte d'autres paramètres comme le non respect de la distanciation par exemple.

5.3 Scénario a étudier

Une fonctionnalité à implémenter sera la mise en quarantaine des individus (contaminés ou non) pour pouvoir étudier les effets de différentes mesures déjà évoquées. Cette fonctionnalité pourra être développée de façon incrémentale en envisageant des périodes de plus en plus éloignées (si possible) tel que J+1, J+7 et J+ 10 par exemple.

5.4 Réserves sur le projet

Ce projet bien que très intéressant nous paraît assez ambitieux même avec une équipe de 6 dans la mesure ou nous manquons de connaissances en modélisation de population et de phénomènes.

6 Références

[1] Itch.io, Sagittarius, George Prosser
https://gprosser.itch.io/sagittarius

[2] Wikipedia, n-body problem
https://en.wikipedia.org/wiki/N-body_problem
https://en.wikipedia.org/wiki/Three-body_problem

[3] Wikipedia, Verlet integration
https://en.wikipedia.org/wiki/Verlet_integration
https://gamedev.stackexchange.com/a/41917

[4] Wikipedia, Runge Kutta methods
https://en.wikipedia.org/wiki/Runge-Kutta_methods

[5] Logisim
http://www.cburch.com/logisim/