Lenglemetz Zoé
Duchez Rémi
Casandjian Ewan

Dossier projet : ISN

Jeu de la vie

Année 2017-2018

Au début, l'enjeu principal de notre projet était de réussir à allier l'art et l'informatique mais des aspects techniques nous ont contraints à revoir cette idée. En effet, notre idée de base était de crée un cube de LED au format 8*8*8. Mais, le coté électronique, et plus particulièrement l'alimentation du cube et de ses différents étages, à représenter un problème majeur. Ainsi voyant que nous n'arrivions pas à résoudre ce problème, nous avons décidé de changer de projet.

Nous avons donc choisis de plutôt allier un aspect mathématique, ici la logique, avec l'informatique. Dans un désir de rester dans une approche ludique, nous avons entamé des recherches pour trouver des jeux correspondant à nos critères.

Dans un premier temps, nous avons d'abord pensé à des jeux tels que les échecs, les dames et le jeu de la vie.

Mais en prenant un peu de recul, nous avons compris que l'intelligence artificielle nécessaire pour créer un jeu de dame ou d'échecs était très complexe.

Notre choix s'est donc porter sur le jeu de la vie. En effet, ce jeu répondait à notre désir d'allier les mathématiques à l'informatique, tout en ayant la possibilité de coder en python.

Le jeu de la vie est un automate cellulaire pensé vers 1970 par un certain John Horton Conway. Depuis ce jour-là et encore aujourd'hui, le jeu de la vie reste l'un des automates cellulaire les plus connus.

Mais qu'est-ce qu'un automate cellulaire ?

Un automate cellulaire peut être considéré comme un objet mathématique et informatique qui évolue étapes par étapes selon des règles définie. Cette évolution est, dans une certaine mesure, une imitation des capacités autoreproductrices que possèdent les cellules constituant les êtres vivants.

Les règles définissant un automate cellulaire peuvent être diverses et variés et selon ces dernières différents dessins apparaissent de par la coloration des cellules en fonction de leur état.

Ainsi, on peut donc dire que le jeu de la vie, n'est en réalité, pas considéré comme un véritable jeu car il ne nécessite pas la présence d'un joueur pour fonctionner. Le principe est simple, l'automate est constitué d'une grille dont les cases sont appelées « cellules », ces dernières peuvent prendre deux états différents, «mort» ou «vivant».

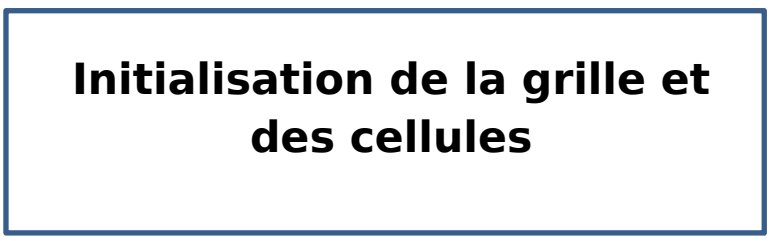
La règle que nous avons décidé d'appliquer était que chaque cellule vivante ne possédant pas au moins deux voisins devait mourir tandis qu'une cellule morte présentant au moins deux voisins vivant devenait vivante à son tour. Mais le changement d'état des cellules étant automatique, l'interaction avec la personne était très faible voir quasi-inexistante.

Cependant, nous voulions faire en sorte qu'il y ait un minimum d'interaction pour rendre la chose plus attrayante. Pour cela nous avons décidé de laisser le choix des cellules de départ à l'utilisateur et non d'utiliser des cellules prédéfinies. De plus, ce dernier peut à tout moment réinitialiser la grille et les cellules.

On peut ainsi considérer que le principe des automates cellulaires constitue le squelette de notre projet. Cependant il fallait maintenant le travailler et le mettre en place pour en arriver petit à petit au jeu de la vie.

Pour coder ce projet, nous avons donc utilisé le langage python. Ce dernier étant relativement simple, possédant une très grande bibliothèque et pouvant être utilisé par de nombreuses applications, il convenait parfaitement à notre projet.

Schéma du jeu de la vie :



Lancement de l’automate cellulaire

L’état des cellules dépend des règles prédéfini. Ici, une cellule morte possédant exactement trois voisines vivantes devient vivante tandis qu’une cellule vivante possédant au moins deux voisines vivantes

La boucle sur laquelle est basé l’algorithme se répète tant que cela est possible

Répartition des taches :

Fonction objet	Tkinter	Algorithme	Incrémentation	Rédaction du dossier

Zoé	Zoé	Zoé		
	Rémi	Rémi		Rémi
			Ewan	Ewan

Démarche collaborative :

Pour créer notre projet nous avons utilisé différentes plateformes collaboratives dont Github et Google doc.

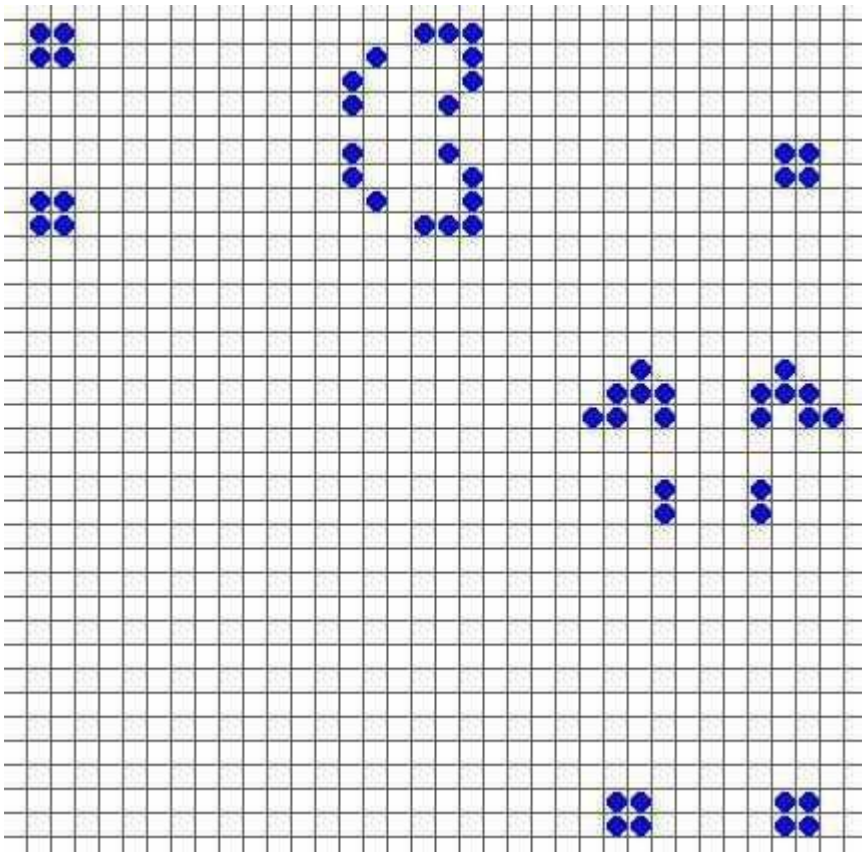
La plateforme Github nous a permis de remplir un cahier de bord et de nous partager nos recherches, les besoins que nous avions pour compléter notre code et enfin de consulter l'avancement du projet et de suivre l'évolution du code.

La plateforme Google doc quant à elle nous a servi pour la rédaction du dossier et le coté algorithmique du projet

Ainsi nous pouvions suivre l'avancement du projet et rajouter petit à petit des choses au code ou corriger des erreurs qui empêchaient le fonctionnement de ce dernier. Nous avons de plus, profité des cours d'ISN pour nous retrouver et en discuter.

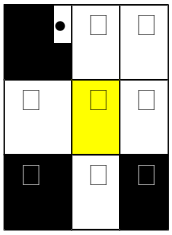
Compte rendu personnel : Ewan

Contrairement à la majeure partie des autres jeux, « le jeu de la vie » possède une certaine valeur artistique.

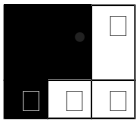


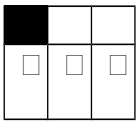
En effet, l'observation de l'évolution des formes crées par ce jeu est fascinant. Malgré cette apparente complexité, le jeu ne repose que sur trois règles fondamentales :

Si une cellule (une case de la grille) a exactement trois voisines vivantes (case de la grille coloriée en noir), elle est vivante à l'étape suivante. Dans cet exemple, la case jaune sera vivante à l'étape prochaine



Si une cellule a exactement deux voisines vivantes, elle reste dans son état actuel à l'étape suivante. Ces trois cellules noires resteront identiques à
L'étape prochaine.





Si une cellule a strictement moins de deux ou strictement plus de trois voisines vivantes, elle est morte à l'étape suivante. La cellule verte va mourir à l'étape prochaine d'asphyxie.

Afin de pouvoir créer ces grilles, nous avons utilisés la bibliothèque graphique « tkinter »

Mon rôle était de coder l'interaction des cellules entre elles.
Pour cela, nous nous sommes orienté vers « la programmation orienté objet ».
La programmation orienté objet a pour intérêt de permettre aux objets, (aux cellules dans notre cas) d'interagir entre eux.

En premier temps, afin de pouvoir analyser correctement la grille, il fallait que nous puissions observer l'état individuel de chaque cellule, afin de faire cela, nous avons créé une boucle dans une boucle :
En premier lieu nous avons défini les axes x et y en fonction du nombre de cellules dans le grillage. Par la suite, nous exécutons à la commande :

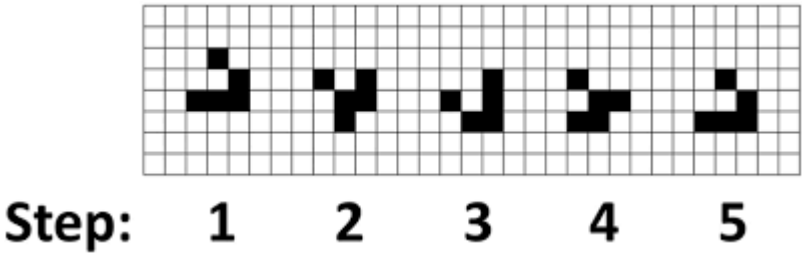
```
for i in range(1, self.size_y + 1):

    for j in range(1, self.size_x + 1):
```

Cette boucle va tourner autant de fois qu'il y a de cellules à analyser.
Nous pouvons par la suite identifier de quelle cellule il s'agit grâce à combien de fois les valeurs de « i » et « j » ont été incrémentés.
Par la suite nous analysons le nombre de voisins par cellule ainsi que sa couleur afin de déterminer son état au prochain tour, nous utilisons par exemple la commande :

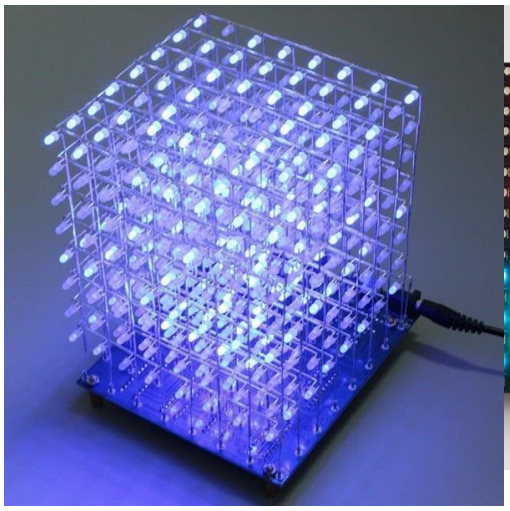
```
elif self.cell_buttons[i][j]['bg'] == "black" and self.neighbor_count(i, j) != 3
and self.neighbor_count(i, j) != 2:
```

ici, nous cherchons a déterminer si la cellule est vivante (noir) et si elle possède aussi en dessous de 2 cellule vivante voisine ou au-dessus de 3 cellules vivante voisine. Dans ce cas-là, la cellule sera tuée (transformé en blanc) au prochain tour.

Une fois le codage terminé, nous devions vérifier que notre jeu suivait correctement les trois règles fondamentales du jeu de la vie mentionné plus tôt, en effet dans le cas contraire notre projet final n'aurait pas été un vrai jeu de la vie. Afin de vérifier que notre version du jeu était fidèle nous avons lancé notre version du jeu avec la première étape de l'image ci-dessous, nous avons ensuite pus constaté que notre version du jeu suivait étape par étape les formes attendus. En effet, les cellules apparaissaient, disparaissaient ou restaient de manière conforme aux trois règles fondamentales. Nous pouvions en déduire que le code était fonctionnel



Ainsi, ce travail m'a été très enrichissant, en effet, j'ai pu prendre connaissance de nouveau types de programmations comme la POO (programmation orienté objet) ainsi que de bibliothèques graphiques comme tkinter.
Cependant j'ai aussi pu apprendre de certaines erreurs. En effet, notre projet original se présentait sous la forme d'un cube composé de 8X8X8 leds. Cependant, j'ai commis l'erreur de ne pas avoir fait assez de recherches avant d'adopter le projet. Bien que mes connaissances en électronique soient très limitées, je me suis lancé dans un projet quasiment exclusivement basé sur l'électronique. Ainsi, nous avons passé un certain temps a « patauger » sans réellement avancé avant de décidé de changer de projet afin d'en commencer un plus réaliste et plus axé sur l'informatique.
Je pense que si nous avions eu plus de temps pour ce projet, il aurait encore pus aller dans des directions différentes. En effet, par exemple, en premier lieu nous avions l'intention d'encoder le jeu de la vie en 3 dimensions sur notre cube led cependant, nous n'avions ni le temps, ni les compétences requises afin d'effectuer ce projet.
De plus, l'image générale des métiers de l'informatique est souvent celle d'un programmeur travaillant tout seul. Il est intéressant de constater que travailler en équipe est possible quel que soit le domaine, informatique ou non grâce à des plateformes en ligne tel que « github »



Premier projet abandonné

second projet abandonné

Lien Github

https://github.com/lasource2018/plus_qu-un_simple_cube.../tree/code