



ÉCOLE SUP GALILÉE

RAPPORT

Etude de cas en Matlab

Élèves :
A ABDELMOULA

Enseignant :
John CHAUSSARD

27 octobre 2017

Table des matières

1	But du projet	2
2	Le jeu de la vie	2
2.1	Le jeu de la vie : la règle	2
2.2	Exemple de structures	2
3	Implémentation du jeu de la vie	3
3.1	Méthode	3
3.2	Utilisation du programme	4
4	Références	9

1 But du projet

Écrire un code Matlab qui, à partir d'une situation de départ, produit les générations successives selon les règles du jeu de la vie.

2 Le jeu de la vie

Le jeu de la vie (*ou Game of Life*) a été inventé en 1970 par le mathématicien *John Horton Conway* en 1970. Il s'agit d'un automate cellulaire à deux états sur une grille infinie. Les éléments de la grille sont appelées des cellules et chaque cellule possède deux états : morte ou vivante. Initialement, un nombre fini de cellules sont vivantes. Toutes les cellules changent simultanément d'état, selon un temps discret. Le nouvel état d'une cellule est déterminé par une règle simple.

2.1 Le jeu de la vie : la règle

La règle est la suivante : Chaque cellule se trouve dans deux états possibles : *vivante* ou *morte*. A chaque génération, et pour chacune des cellules, on compte le nombre de cellules vivantes de son voisinage (les huit cellules en contact direct). Deux cas se présentent :

Une cellule *morte* entourée d'exactly trois cellules vivantes devient vivante sinon, elle reste morte.



Une cellule *vivante* entourée de deux ou trois cellules vivantes reste vivante sinon, elle meurt de solitude ou d'étouffement.

Par exemple, la configuration  devient .




Bien que les règles soient simples, ce jeu permet le développement de motifs très complexes. Durant l'exécution des règles, plusieurs structures particulières peuvent apparaître, notamment des structures stables, des oscillateurs, ou encore des vaisseaux.

2.2 Exemple de structures

Structures stables : ces structures ne changent jamais d'un instant à l'autre, à moins qu'un parasite arrive à proximité de la structure. Par exemple, le bloc de quatre cellules :

 ou encore le bateau .

Oscillateurs : ce sont des motifs qui se transforment de manière cyclique avant de retrouver leur état initial. Le premier exemple donné est un oscillateur sur deux périodes :

la configuration  devient , qui redevient , et ainsi de suite.

Vaisseaux : ce sont des structures pouvant se décaler en gardant la même forme après un certain nombre de générations. Par exemple, ce motif, nommé *glider*, retrouve son état

initial, mais translaté, après quatre périodes :  →  →  →  → .

Autres : il en existe bien d'autres, comme les « puffeurs » (vaisseaux laissant des débris), les canons (oscillateurs qui émettent des vaisseaux), le mathusalem qui explose en de nombreux objets stables etc. Des animations sont disponibles sur internet pour voir ce que cela donne.

Les canons : [https://fr.wikipedia.org/wiki/Canon_\(automate_cellulaire\)](https://fr.wikipedia.org/wiki/Canon_(automate_cellulaire)).

Le Puffeur : <https://fr.wikipedia.org/wiki/Puffeur>.

3 Implémentation du jeu de la vie

3.1 Méthode

Pour commencer, nous allons créer une matrice M de N par N , dont les valeurs des cellules (mortes / vivantes) sont générées aléatoirement avec $randi([0,1],N,N)$. Cette matrice est donc la situation initiale de l'automate. Il s'agit maintenant de créer deux fonctions permettant de faire les tests nécessaires (état de la cellule testée et celle des cellules voisines) afin de déterminer si une cellule à une case donnée sera vivante (état 1) ou morte (état 0) à la génération suivante.

On suppose que la cellule morte est égal à 1 et la cellule vivante est égal à 0.

Notons qu'une cellule (vivante ou morte) qui se trouve sur l'un des 4 coins de M c'est-à-dire en $M(1,1)$, $M(1,N)$, $M(N,1)$ et $M(N,N)$ ont exactement 3 voisins et une cellule (vivante ou morte) qui se trouve sur la première colonne ou sur la première ligne ou sur la dernière ligne ou sur la dernière colonne excepter les 4 coins de M ont 5 voisins. Et les autres cellules ont 8 voisins.

Commençons par la première fonction **test_voisin_cell_morte**(M, i, j) qui teste l'état des cellules qui entoure la cellule morte (i,j) , si la cellule morte (i,j) admet exactement 3 voisins vivants, alors la cellule (i,j) devient vivante, la fonction retourne 1 si la cellule devient vivante, 0 sinon. Cette fonction teste l'état des voisins de la cellule morte (i,j) selon sa position dans M .

La deuxième fonction **test_voisin_cell_vivante**(M, i, j) teste l'état des cellules qui entoure la cellule vivante (i,j) , si la cellule vivante (i,j) est entourée de deux ou trois cellules vivantes reste vivante sinon, elle meurt de solitude ou d'étouffement, la fonction retourne 1 si la cellule reste vivante, 0 sinon. Cette fonction teste les l'état des voisins de la cellule vivante (i,j) selon sa position dans M .

Ensuite, il faudra créer deux boucles pour parcourir et tester l'état de l'ensemble des cellules (i,j) de la matrice M .

Si $M(i,j) == 1$ (cellule morte) alors on appelle la fonction **test_voisin_cell_morte**(M, i, j) qui va nous dire si la cellule $M(i,j)$ va devenir vivante ou elle va rester morte, si la fonction **test_voisin_cell_morte**(M, i, j) renvoie 1, alors $M(i,j) \leftarrow 0$. La cellule devient vivante.

Si $M(i,j) == 0$ (cellule vivante) alors on appelle la fonction **test_voisin_cell_vivante**(M, i, j) qui va nous dire si la cellule $M(i,j)$ va rester vivante ou elle va mourir, si la fonction **test_voisin_cell_morte**(M, i, j) renvoie 0, alors $M(i,j) \leftarrow 1$. La cellule meurt.

3.2 Utilisation du programme

1- Le menu :

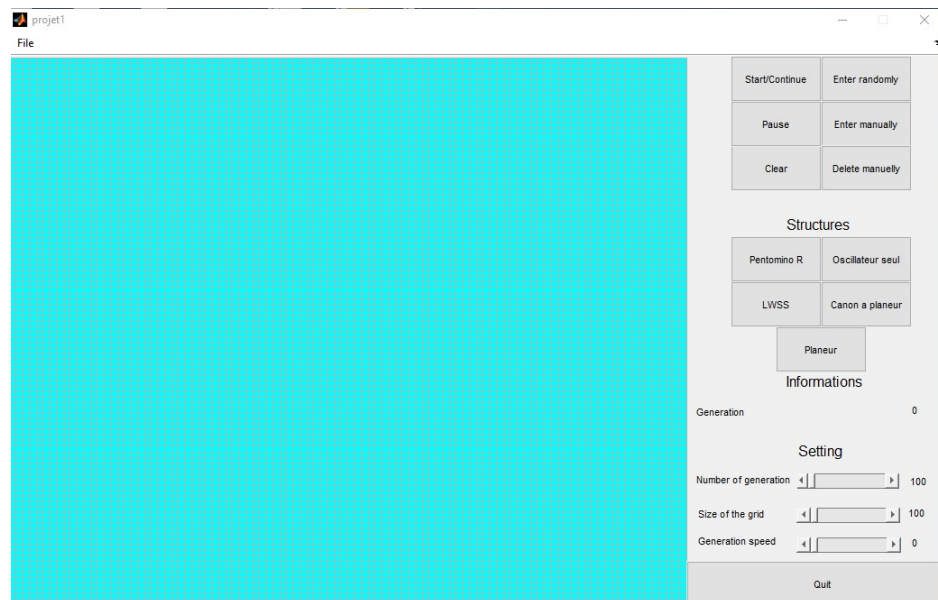
Affichage d'un menu avec les différentes options du jeu, dont une pour lancer le jeu avec les règles de base, une permettant de quitter le programme, et une pour mettre en pause la simulation pour observer plus en détail les évolutions etc...

Pour démarrer la simulation, il faut cliquer sur le bouton *Start/Continue*.

Pour créer une grille *aléatoire*, il faut cliquer sur le bouton *Enter randomly*.

Pour mettre la simulation en *pause*, il faut cliquer sur le bouton *Pause*.

Nb : Si on veut ajouter des cellules ou des structures dans un schéma en cours de simulation, il faut avant cliquer sur le bouton pause.



2- Les paramètres :

La taille de la grille : il faut que l'utilisateur donne la taille de la grille (l'univers) avant le début de la simulation, si l'utilisateur tente de changer la taille de la grille en cours de la simulation, un message d'erreur s'affichera. L'utilisateur peut modifier la taille de la grille grâce au bouton « Size of grid » de type *Slider*.

Nombre de génération : L'utilisateur peut faire varier le nombre de génération de 100 à 10100 grâce au bouton « Number of generation » de type *Slider*.

La Vitesse de simulation : L'utilisateur peut faire varier la vitesse de simulation grâce au bouton « Generation speed » de type *Slider*.

Setting

Number of generation 100

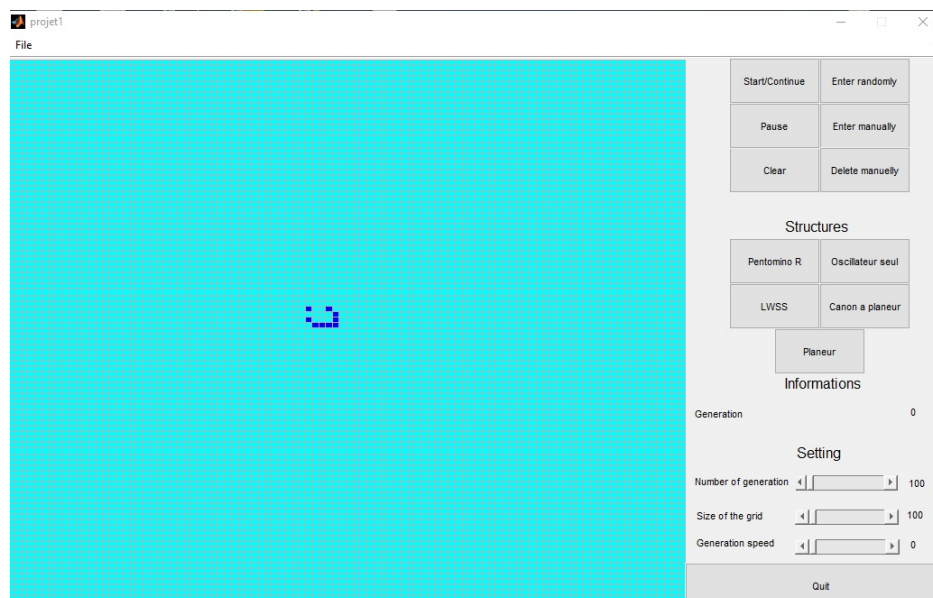
Size of the grid 100

Generation speed 0

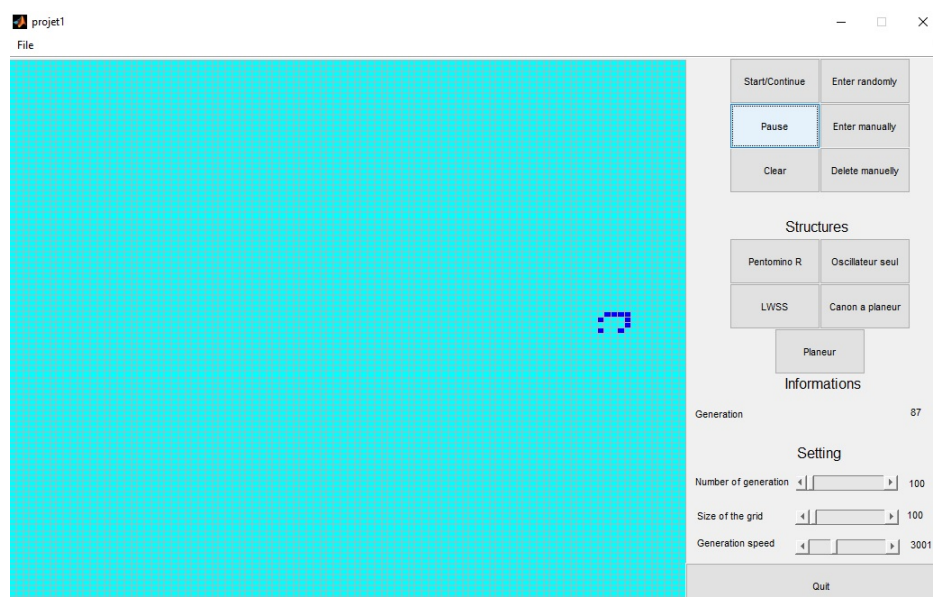
3- Quelques structures :

a- Chasseur LWSS : Trois vaisseaux qui se déplacent horizontalement (ou verticalement) de deux cases toutes les quatre générations. Ils portent en anglais le nom de Light, Medium et Heavy Weight Spaceships (littéralement « vaisseaux de poids léger, moyen et lourd »), généralement abrégés en LWSS.

Génération 0

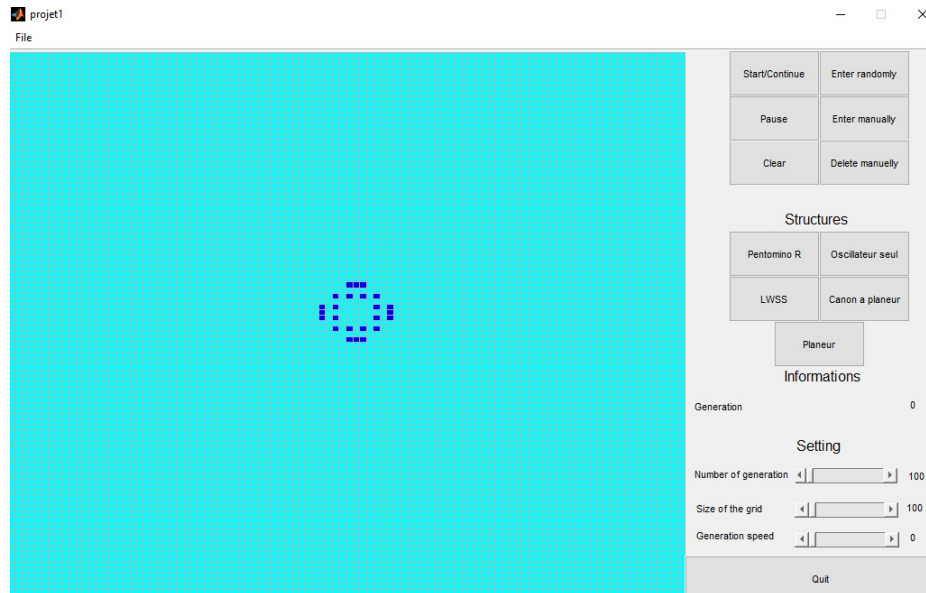


Génération 87

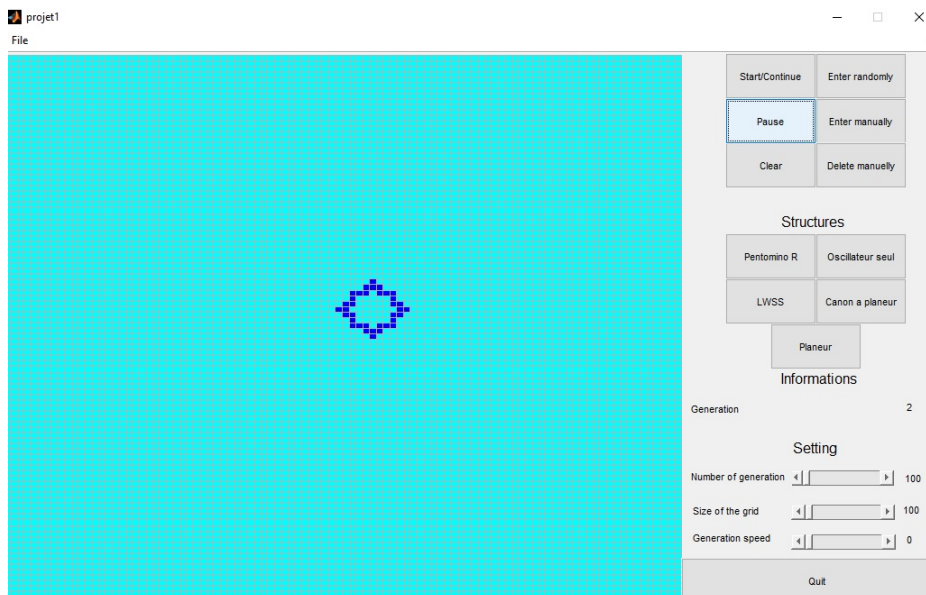


b- *Oscillateur seul* : C'est une structure périodique, qui reprend sa forme initiale après 2 itérations.

Génération 0

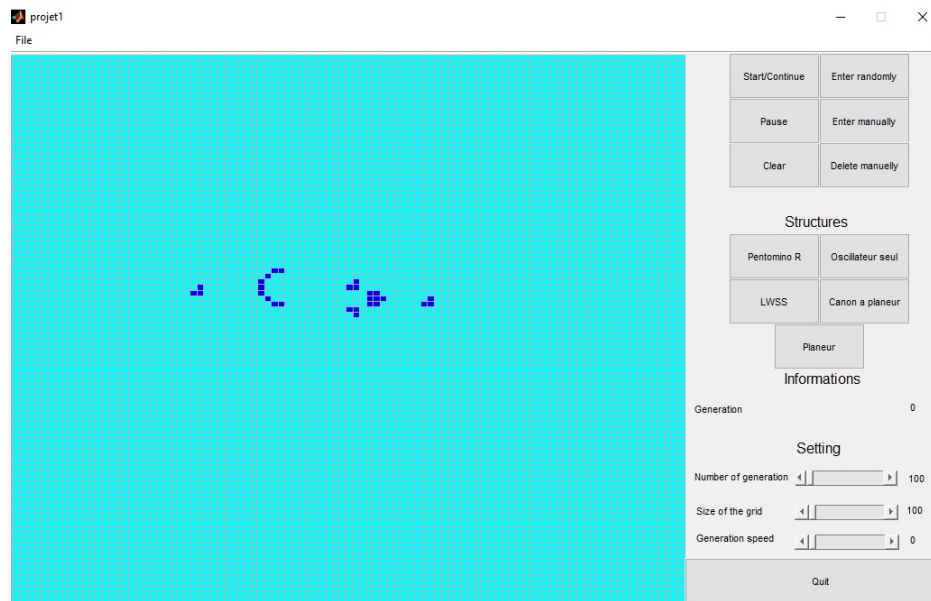


Génération 2

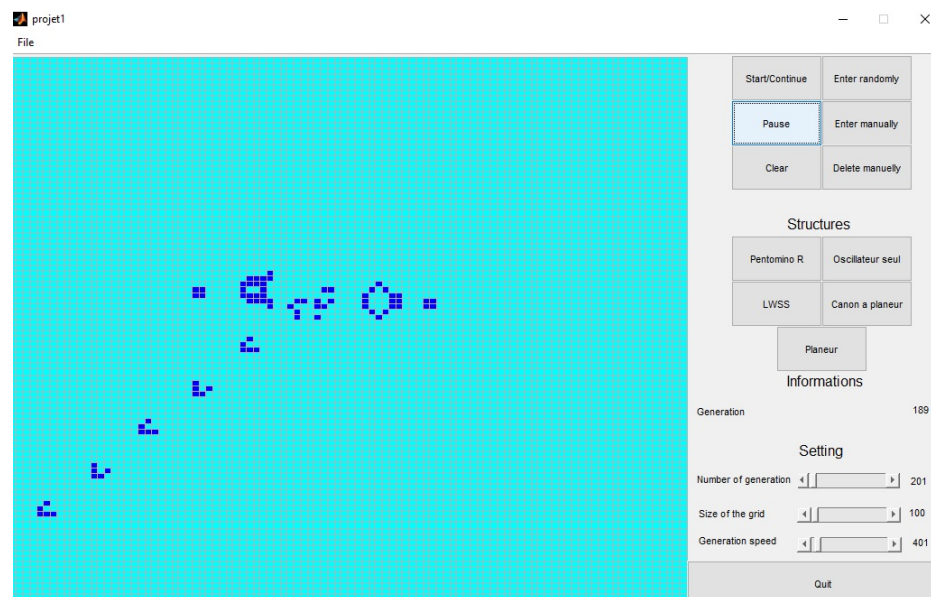


c- *Canon a planeur* : Les canons sont constitués de 2 navettes centrales qui oscillent entre 2 blocs. Le canon produit alors un planeur toutes les 30 itérations. Les canons sont de véritables "créateurs" de matière.

Génération 0

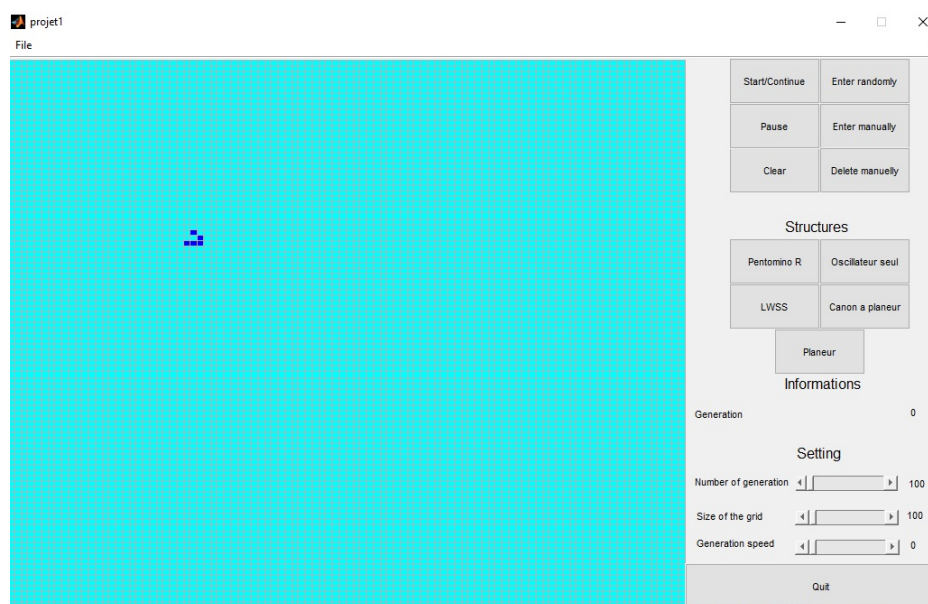


Génération 189

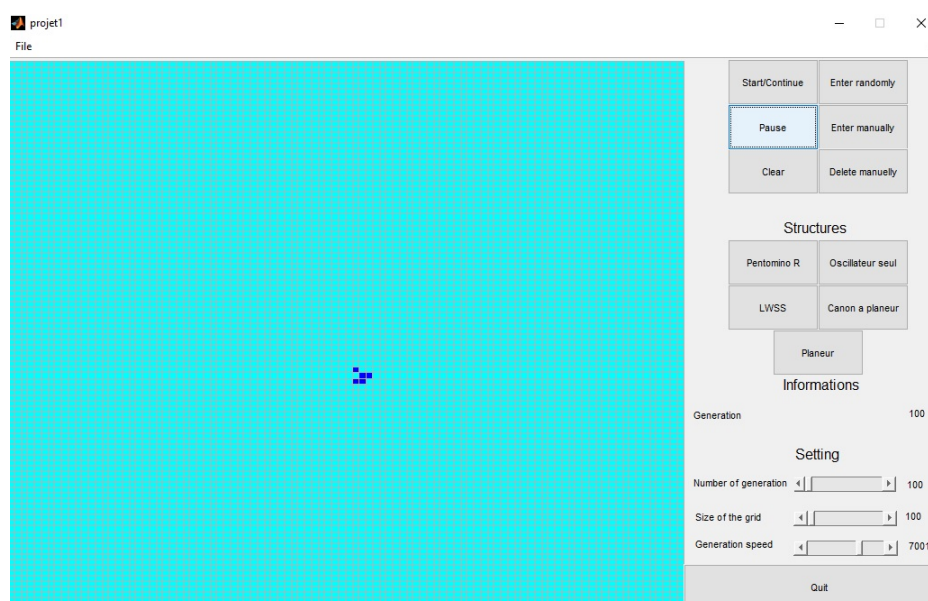


d- Planeur : Les planeurs adoptent une forme particulière de vaisseau. Elle est simplement composée de 5 cellules dans chacun des 4 stades de son évolution. C'est une structure animée qui se déplace en diagonale dans l'univers du jeu, donnant la preuve qu'une forme élémentaire peut naître, mourir... mais aussi se mouvoir.

Génération 0

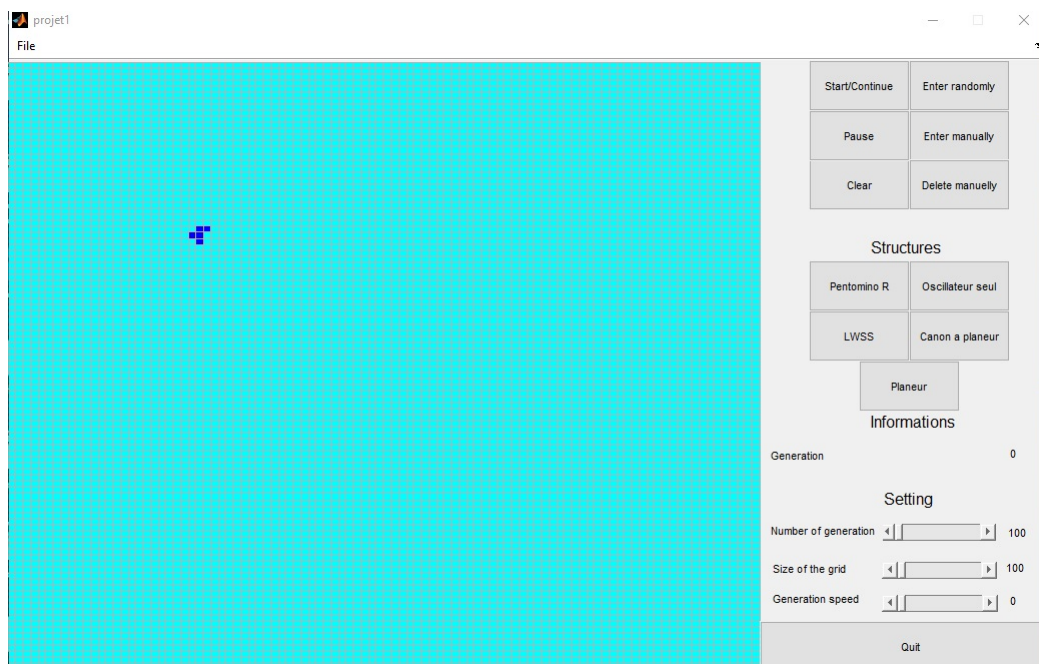


Génération 100

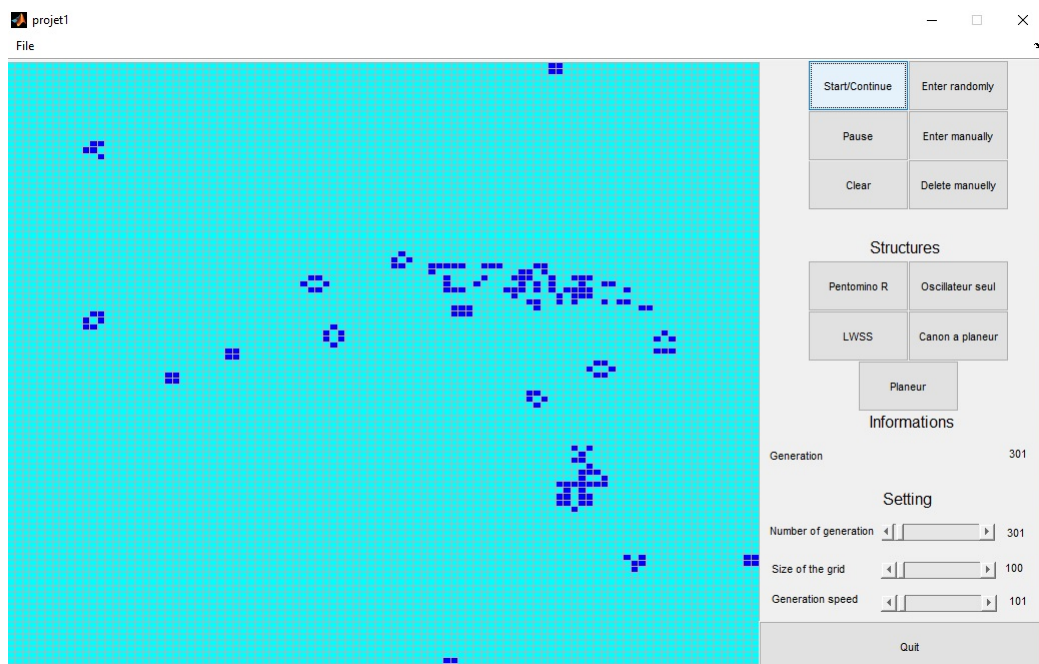


e- Pentomino R

Génération 0



Génération 301



4 Références

[1] Wikipédia, articles Automate cellulaire et Jeu de la vie. https://fr.wikipedia.org/wiki/Jeu_de_la_vie.

[2] Wikibooks, Automate cellulaire, Jeu de la vie et structure stable. https://fr.wikibooks.org/wiki/Automate_cellulaire/Jeu_de_la_vie/Structure_stable.

[3] Wikipédia, articles Jeu de la vie(Planeur). [https://fr.wikipedia.org/wiki/Planeur_\(jeu_de_la_vie\)](https://fr.wikipedia.org/wiki/Planeur_(jeu_de_la_vie)).