BRUCE ROSE 3700694
DANIELA BAKA 3703218

Projet de Vie Artificielle et Jeux Systémiques

Rapport de Bruce Rose et Daniela Baka

Table des matières

Introduction

Dans le cadre de l'unité d'enseignement *Projet de Vie Artificielle et Jeux Systémiques* (2I013-P4) de la faculté *Pierre et Marie Curie (Sorbonne Université)*, nous avons été amené à développer la simulation d'un monde artificiel abritant une forme de vie adaptative devant faire face à des défis environnementaux dynamiques. En outre, nous avons dû concevoir un écosystème équilibré et durable. Nous vous présenterons dans un premier temps les fonctionnalités relatives au terrain que nous avons décidé d'implémenter ainsi que la manière dont nous nous y sommes pris. Ensuite, nous ferons de même pour les fonctionnalités relatives aux agents peuplant notre écosystème.

1 Écosystème

1.1 Terrain

La fonction étant en charge de générer le terrain sur lequel notre écosystème s'établira peut utiliser une image en nuances de gris. Plus un pixel est blanc/clair, plus haute s'en retrouve la case du terrain. Au contraire, plus noire/foncée est le pixel, plus bas est la case. Nous avons donc décidé de réaliser un automate cellulaire pour générer une image en nuance de gris. L'idée était d'imiter un bruit de Perlin, mais également de générer un volcan au centre du terrain.

Les cases de l'automate cellulaire utilisées sont des structures nommées LsCell qui possèdent trois attributs :

- Une *blancheur* ;
- Un *mode* ;
- Un entier nommé *paramètre*.

Il existe 5 modes :

- Normal ;
- Volcano_up ;
- Volcano_down ;
- Stop ;

- NormalStop.

Chaque mode a une fonction différente. En mode normal, l'attribut *blancheur* d'une cellule LsCell a une chance de spontanément devenir plus ou moins élevée à condition que le nombre d'itération courant est inférieur au nombre d'itération maximum divisé par 1,5. Autrement dit, les cases peuvent changer d'intensité de blanc pendant les deux premiers tiers du nombre d'itération maximum passé en paramètre. Si la case *Normal* ne voit pas son attribut blancheur changer spontanément d'intensité, celle-ci devient alors la moyenne des blancheurs des cases avoisinantes ainsi que d'elle même (voisinage de Moore + soi-même), formant ainsi une texture de *nuage*.

Par défaut, la case au centre de l'image est une case dont le mode est *Volcano_up* (couleur violette sur les images ci-dessous).

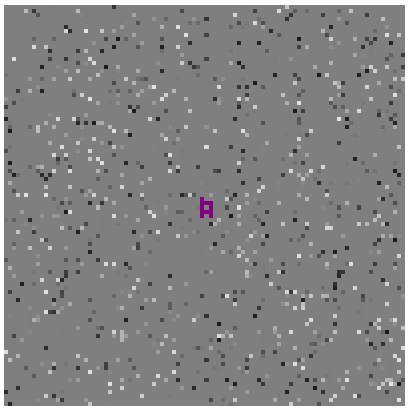


FIGURE 1 Image en nuances de gris générée à l'itération 1/30.

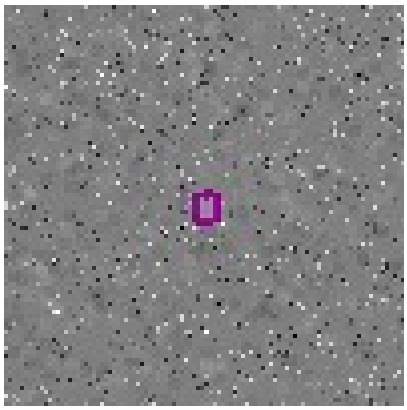


FIGURE 2 Image en nuances de gris générée à l'itération 2/30.

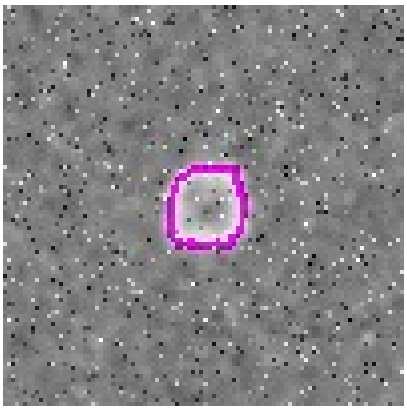


FIGURE 3 Image en nuances de gris générée à l'itération 5/30.

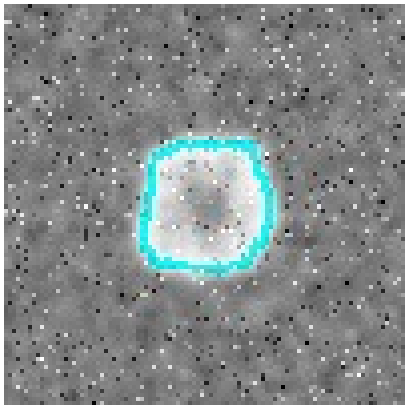


FIGURE 4 Image en nuances de gris générée à l'itération 9/30.

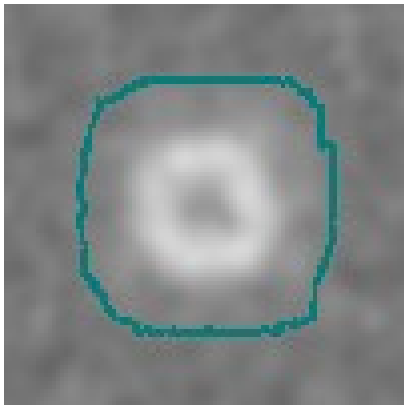


FIGURE 5 Image en nuances de gris générée à l'itération 17/30.

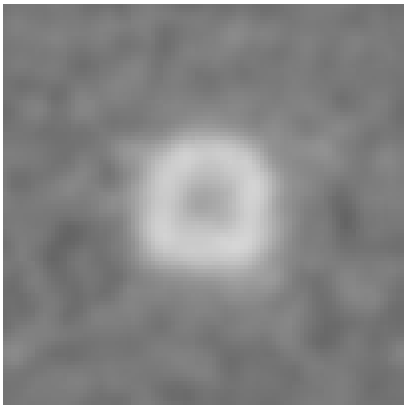


FIGURE 6 Image en nuances de gris générée à l'itération 18/30.

Une cellule dont le mode est *Volcano_up* change d'office celui-ci à *NormalStop*, mais exécute tout de même le comportement d'un *Volcano_up*, pour une itération. Si le *paramètre* d'un *Volcano_up* est inférieur ou égal à 7, ses voisins se voient devenir des *Volcano_up*, à condition qu'ils étaient à *Normal* avant. Leurs *paramètres*, qui étaient initialement égaux à 0, deviennent celui du *Volcano_up* « contaminant » incrémenté de 1. Enfin, les *blancheurs* des cases « contaminées » sont recalculées selon la formule suivante :

$$CaseAdjacente.blancheur = CaseAdjacente.blancheur + \left(CaseAdjacente.paramètre \right)^2 \times 1{,}5 + 1$$

Elle est inspirée de la fonction

$$y = x^2 \times 1{,}5 + 1$$

où *CaseAdjacente.paramètre* fait office de *x*. Cette fonction permet de former un dénivelé harmonieux pour la pente intérieure du volcan.

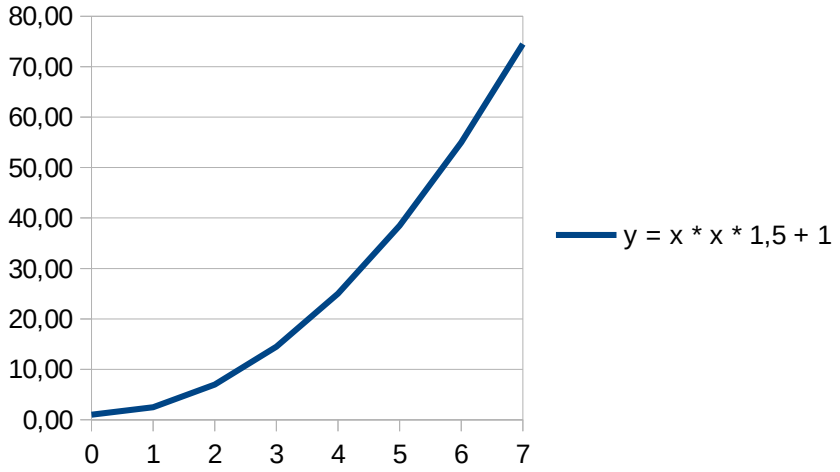


FIGURE 7 Graphe de la fonction de pente intérieure du volcan

Si le paramètre d'un *Volcano_up* est supérieur à 7, il devient un *Volcano_Down* (couleur bleue sur les images ci-dessus).

Un *Volcano_down* devient d'office un *NormalStop*, mais suit tout de même le comportement d'un *Volcano_down* pour une itération. Si son *paramètre* n'est pas inférieur à 0, tous ses voisins qui sont *Normal* deviennent des *Volcano_down*. Les voisins ainsi *Volcano_down*-isés voient leurs *paramètres* être égaux au paramètre du *Volcano_down* « contaminant », décrémenté de 1.

Enfin, les *blancheurs* des cases « contaminées » sont recalculées selon la formule suivante :

$$CaseAdjacente.blancheur = CaseAdjacente.blancheur - (CaseAdjacente.paramètre) \times 1{,}1 + 1$$

Elle est inspirée de la fonction

$$y = constante - x \times 1{,}1$$

où *CaseAdjacente.paramètre* fait office de *x*. Cette fonction permet de former un dénivelé harmonieux pour la pente extérieure du volcan.

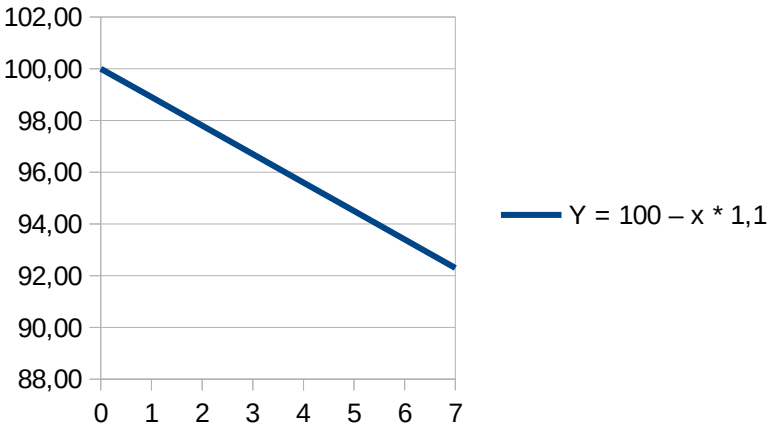


FIGURE 8 Graphe de la fonction de pente extérieure du volcan

Une cellule de type *Stop* est une cellule qui reste statique. Elle ne peut être modifiée.

Une cellule de type *NormalStop* est la moyenne des *blancheurs* des cellules environnantes à la manière d'un *Normal*, sauf qu'elle ne peut être transformée en une cellule de mode différent.

Lors du lancement de la simulation, un nombre aléatoire est tiré entre 20 et 50. Ce nombre est alors utilisé comme nombre d'itération maximum pour l'automate cellulaire s'occupant de générer l'image en nuances de gris à partir de laquelle notre programme pourra générer le monde virtuel. Ceci nous assure de toujours avoir un monde différent à chaque génération. Plus le nombre d'itérations est grand, plus « lissé » paraît le terrain.

1.2 Lave

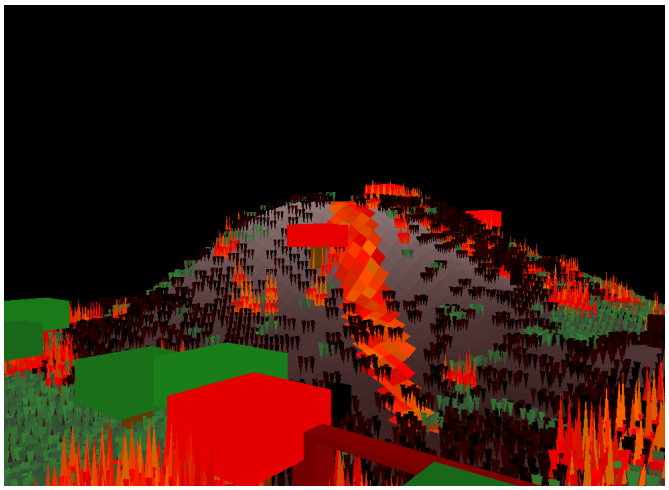


FIGURE 9 Lave coulant le long du volcan

Un volcan n'en serait pas vraiment un sans *lave*. Puisqu'une fois n'est pas coutume, nous avons décidé d'utiliser un automate cellulaire pour gérer la quantité de fluide magmatique sur une case donnée du terrain. Il s'agit simplement d'un nombre (*double*) pour chaque case. Les cases contenant une quantité supérieure à 0 de lave sont alors coloriées en teintes de rouge / jaune scintillant. Les cases qui sont de hauteur égale à 0 sont des cases d'eau et sont initialement peintes en bleu. L'automate cellulaire s'occupant de la lave repeint alors continuellement les cases du terrain qui ne contiennent pas de lave et qui sont à 0 de hauteur en bleu. De même, les cases du terrain émergées mais ne contenant pas de lave ont une couleur définies initialement en fonction de la hauteur du terrain. L'automate cellulaire de la lave se charge de repeindre continuellement ces cases dans leurs couleurs d'origine en tenant compte de leur hauteur.

Pour chaque case contenant un niveau de lave positif non-nul, l'automate cellulaire visite les cases voisines et deux cas de figure se présentent alors :

- la case visitée est de hauteur inférieure ou égale à celle de la hauteur de la case centrale. Dans ce cas, le niveau de lave de cette case voisine est incrémentée de 1 tandis que celui de la case centrale est décrémentée de 1.
- la case visitée est de hauteur supérieure. On ne fait rien.

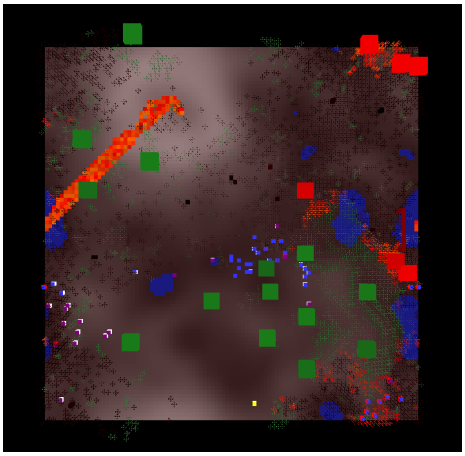


FIGURE 10 *En haut à gauche de l'image, lave coulant le long du volcan, vue de dessus*

2 Agents

L'écosystème que nous avons érigé comporte quatre types d'Agents : les *Herbes*, les *Arbres* (les *Plantes*), les *Proies* et les *Prédateurs* (les *Animaux*). Ces quatre agents sont contenus dans un automate cellulaire sur listes. Les listes nous permettent de placer plusieurs agents sur la même case (c'est cependant faux pour les plantes).

Les deux super-types Plante et Animal ont des caractéristiques communes. Ils possèdent tous deux a un état qui peut prendre quatre valeurs : *ALIVE*, *BURNING*, *BURNT*, et *DEAD*, ainsi que certaines caractéristiques comme une *santé*, et une *énergie courante*.

En plus de ces attributs, la classe Animal est un peu plus complète. Elle possède :

- un *age* ;
- une *lenteur* ;
- un *age de mort* ;
- une *santé maximum* ;
- un *partenaire de reproduction* ;
- un *seuil d'énergie limite* ;
- un *champs de vision* ;
- un *comportement* qui peut prendre divers états en fonction du type de l'animal ;

- un *age de fertilité* ;
- et une *liste de voisins*.

2.1 Végétaux

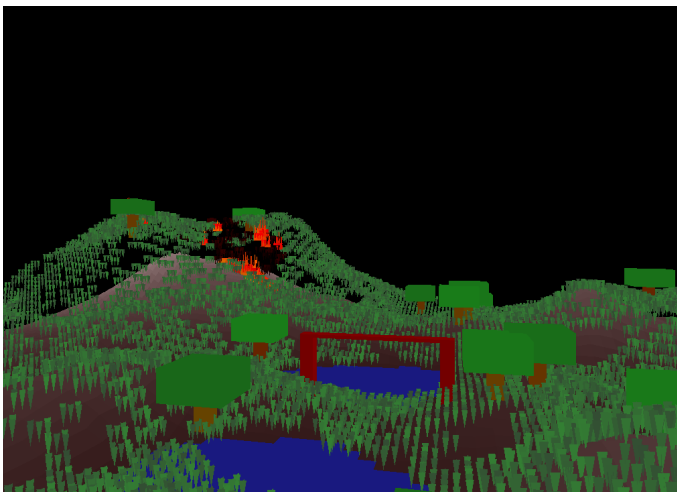


FIGURE 11 *Arbres et herbes se côtoyant dans notre monde artificiel. On aperçoit au loin sur le volcan un feu de forêt.*

Les végétaux se déclinent sous deux formes : les *Herbes* et les *Arbres*.

Les herbes sont représentées par de petits groupes de triangles renversés.

Lorsque l'état d'une herbe est ALIVE, elle vérifie d'abord que la case sur laquelle elle se situe ne contient pas de lave. Dans l'affirmative, son état devient alors BURNING. Sinon, l'herbe a une chance sur cent mille de devenir un arbre. Ensuite, elle visite les cases avoisinantes. Si elle y trouve une plante (arbre ou herbe) en état BURNING, alors cette herbe a également une chance sur quatre cents de devenir BURNING. Enfin, s'il n'y a pas de plante sur une case avoisinante, il y a trois chances sur vingt mille qu'il y apparaisse une herbe. Quand elle est BURNING, une herbe met en BURNING les animaux adjacents et apparaît en rouge. Aussi, sa santé diminue de 1 à chaque itération. Quand la santé d'une herbe atteint 0, elle est fixée à 1 symboliquement et passe à l'état BURNT (Une herbe dont la santé est 0 est d'office supprimée, considérée DEAD). Une herbe à l'état BURNT apparaît en noir. Elle a alors une chance sur mille de devenir DEAD et d'être ainsi supprimée par l'automate cellulaire des agents. Les arbres sont représentés par des pavés verticaux marrons surmontés de cubes verts. Ils fonctionnent exactement de la même manière que les herbes.

Cependant, comme nous le verrons par la suite, les arbres ont un rôle pilier au sein de l'écosystème, dans la mesure où ceux-ci ne peuvent être consommés par les animaux, à la différence des herbes.

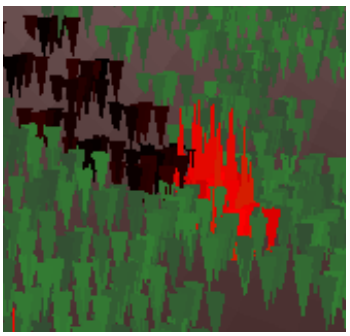


FIGURE 12 *En vert des herbes vivantes. En noir/rouge foncé, des herbes brûlées. Enfin, au centre en rouge vif, des herbes en flammes.*

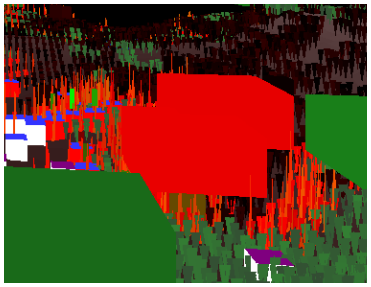


FIGURE 13 *Arbres en feu.*

2.2 Animaux

Les *Proies* et *Prédateurs* possèdent une barre de vie verte visible au dessus d'eux lorsque leurs santés ne sont pas à leurs maximums et qui change de couleur en fonction du niveau de celles-ci, allant du vert (bonne santé) à rouge (mort imminente).

Ils sont représentés dans le monde par des figures cubiques de couleur violette pour les *Prédateurs* et bleue pour les *Proies*.

2.2.1 Proies

En plus de son état vital hérité de Animal (ALIVE, DEAD, etc.) et de divers attributs, une proie possède un attribut *food* ainsi qu'un attribut *threat*. Le comportement d'une proie peut prendre quatre valeurs :

- en *Rut* ;
- *Normal* ;

- en fuite (*flight*) ;
- en quête de nourriture (*Hungry*).

Avant toute chose, si une Proie est vivante, plusieurs conditions doivent être validées pour la laisser exécuter son comportement. Si la case sur laquelle se trouve la Proie contient de la lave, celle-ci se met en mode BURNING. Ensuite, si sa santé est inférieure ou égale à 0, que son âge est supérieur à l'age limite ou que son énergie est égale à 0, l'état de la Proie devient DEAD. En outre, si l'attribut Threat n'est pas vide, alors la Proie adopte le comportement Flight.

En mode Normal, une Proie a tout d'abord une chance d'entrer en Rut. Cela arrive si l'âge de la Proie est supérieure à l'age de fertilité. Un nombre décimal entre 0 et 1 est également tiré au sort. La Proie entre en Rut uniquement si ce nombre est inférieur à un sur deux fois le nombre de Proies déjà existantes. La condition est alors :

```
(age >= birthAge && Math.random() < 1. / (nbIndividusProie * 2))
```

Ces conditions permettent à la population de Proie de s'auto-réguler. Avec ces règles, les Proies vont davantage se reproduire lorsqu'elles seront peu, évitant ainsi une extinction prématurée de l'espèce. A l'inverse, pour empêcher les Proies d'entrer en surpopulation, celles-ci ont moins de chance d'entrer en Rut lorsque leur effectif croît.

Ensuite, si l'énergie courante de la Proie est inférieure au seuil d'énergie limite, la Proie entre en mode Hungry. L'attribut énergie des Proies peut être imaginé comme la quantité de graisse que celles-ci possèdent. Lorsque leurs réserves s'amenuisent, c'est le signal pour se nourrir de nouveau.

La fin du comportement Normal est marquée par un appel à un pas de direction aléatoire.

En mode Flight, une Proie part dans la direction opposée à la position de l'agent contenu par l'attribut Threat. Pour ce faire, on calcule d'abord la différence des x et des y entre les deux agents. Si cette différence est supérieure à deux fois la largeur/hauteur du monde, alors plutôt que de diminuer cet écart, nous l'agrandissons en partant dans la position opposée. On profite ainsi de l'aspect torique du monde, ce qui fait toujours prendre le plus court chemin d'un point A à un point B[FIGURE 14]. Comme on ne veut pas aller vers le danger, nous inversons ces règles.

0, 10											
0, 9											
0, 8											
0, 7							<- Proie1			<- Préda1	
0, 6											
0, 5											
0, 4											
0, 3		Proie2 ->									Preda2 ->
0, 2											
0, 1											
0, 0											
	0, 0	1, 0	2, 0	3, 0	4, 0	5, 0	6, 0	7, 0	8, 0	9, 0	10, 0

FIGURE 14 Schéma explicatif des distances dans un monde torique. La distance pour les x entre Proie1 et Préda1 est de 2, ce qui est inférieur à la moitié de la largeur du tableau qui est de 5,5. Ceci signifie que pour fuir Preda1, la Proie1 doit agrandir la distance entre elle et le Préda. La distance pour les x entre Proie2 et Préda2 est de 8, ce qui est supérieur à 5,5. Pour Proie2, s'approcher de Préda2 (en allant vers la droite) revient à s'en éloigner du fait du monde torique.

Quand une Proie est en Rut, si son attribut Mate n'est affecté à aucune autre Proie alors elle en cherchera une. Il s'agit de parcourir les cases adjacentes dans un rayon du champs de vision de la Proie et de choisir la Proie qui est la plus proche. Quand un partenaire de reproduction est trouvé, la Proie se dirige en sa direction. Quand une Proie atteint son partenaire de reproduction, elle génère une nouvelle Proie d'âge 0. L'énergie du parent se voit alors divisée par trois, et son nouvel état est Normal.

Enfin, quand une Proie est Hungry, elle agit comme pour la reproduction, sauf qu'elle ne cherche pas une autre Proie, mais un agent de type Herbe. Lorsqu'elle atteint l'herbe, la Proie gagne en énergie celle stockée dans Herbe, et cette dernière se voit attribuer 0 à sa santé ce qui entraîne sa mort.

Quand une Proie est BURNING, elle apparaît en rouge et perd des points de vie à chaque tour. En même temps, elle a une certaine probabilité de ne plus être en feu.

2.2.2 Prédateurs

Les différents états des Prédateurs sont *Normal*, *Rut*, *Hunting* et *Returnhome*.

Avant toute chose, si un prédateur est vivant, plusieurs conditions doivent être validées pour le laisser exécuter son comportement. Si la case sur laquelle il se trouve contient de la lave, il se met en mode BURNING. Ensuite, si sa santé est inférieure ou égale à 0, que son âge est supérieur à l'age limite ou que son énergie est égale à 0, son état devient DEAD.

A l'état *Normal*, un Prédateur se déplace de manière aléatoire. Il a alors une certaine probabilité d'entrer en *Rut* s'il atteint l'âge de fertilité. Un nombre décimal entre 0 et 1 est également tiré au sort. Le Prédateur entre en Rut uniquement si ce nombre est inférieur à un certain nombre calculé à partir du nombre de Proies existantes. La condition est alors la suivante :

```
(age >= birthAge && Math.random() < 0.001 * ProieAgent.nbIndividusProie /
(nbIndividusPreda + 1.))
```

Ces conditions permettent à la population de Prédateurs de s'auto-réguler. Avec ces règles, les Prédateurs vont davantage se reproduire lorsque le nombre de Proie sera grand, évitant ainsi de dilapider le monde de sa population de Proies. Pour la même raison, ceux-ci ont moins de chance d'entrer en Rut lorsque leur effectif croît trop.

Un Prédateur a également la possibilité d'entrer en état *Hunting* si son énergie descend en dessous d'un certain seuil fixé. Dès que la chasse commence, la position originale du Prédateur est stockée. Il cherche la Proie la plus proche dans son champs de vision, en fait sa cible puis la chasse. Quand un prédateur mange une Proie il récupère l'énergie qui était la sienne. La chasse se termine quand son énergie est supérieure ou égale au double du seuil fixé : il redevient *Normal*.

Lorsqu'un Prédateur est en mode ReturnHome, celui-ci retourne à la position indiquée par son origine, stockée au début de la chasse. Ceci permet d'observer deux groupes plus distincts d'animaux, les uns carnivores, et les autres, herbivores, et de les voir interagir entre eux sans pour autant créer un gros amas d'agents compactés.

Quand un Prédateur est en Rut, les mêmes mécanismes que pour les Proies sont observés.

Quand un Prédateur est BURNING, celui-ci perd de la vie au fur et à mesure qu'il brûle.

2.2.3 Remarques

Quand il ne reste presque plus d'herbe dans le monde, les proies forment de petites colonies et se regroupent autour des arbres qui génèrent des herbes.

Les deux espèces cohabitent et interagissent tout en maintenant un certain équilibre au niveau de leurs nombres. On obtient alors un ratio qui tend vers un prédateur pour quatre proies.

A Scrum Documents

Product Backlog

User Stories	Tâches	Estimation de la taille	Estimation de la priorité
L'utilisateur veut voir le monde simple	Décider de faire un monde en 3D ou 2D	10	0
Voir des arbres Simples et des brousses simples	Implémenter des agents Arbres et Brousses.	5	1
	Modéliser ces agents	2	
Voir de l'eau	Rien à faire !	0	2
Voir un Volcan	Implémenter un automate cellulaire générant une image en nuances de gris.	20	3
	Écrire les règles de comportement des cellules de l'automate	10	
	Intégrer cette fonctionnalité au projet.	5	
Voir des animaux qui interagissent entre eux	Implémenter des agents « Proies »	10	4
	Implémenter des agents « Prédateurs »	10	

	Implémenter la consommation de végétaux pour les proies	5	
	Créer une fonction de recherche d'agent	6	
	Factoriser cette fonction de recherche à toutes les méthodes l'utilisant	4	
	Implémenter un comportement de fuite pour les proies	2	
	Ajuster l'équilibre des populations	10	
L'utilisateur veut voir des feux de forêt	Implémenter un état de brûlure pour tous les agents	5	5
	Implémenter les comportements appropriés	5	
Voir l'évolution des animaux grâce à l'implémentation d'un ADN qui muterais en fonction des différentes	Déterminer des traits critiques à la survie des animaux (exemples : vitesse de course, force physique)	10	6

reproductions et de la sélection naturelle	Implémenter des contraintes de trait : les animaux ne peuvent pas être super-fort sans que ça aie un coût sur leur nourriture consommée par exemple.	6	
	Implémenter la méiose : un système de probabilité de fréquence et d'amplitude des mutations génétiques lors de la reproduction	20	
	Tester et affiner	15	
L'environnement est complexe, avec différents types de sol, ou bien différents types d'ensoleillement, etc	/	20	8
Voir l'évolution des Plantes selon la même idée que pour les animaux	/	30	7

Enrichir le degré de liberté d'évolution des animaux, avec d'avantage de gênes et donc de traits possibles	/	30	9
Voir l'interaction entre volcan et eau (création de roche volcanique)	/	25	10
Visualiser les nuages, et un ciel	/	30	11
Visualiser la morphologie des animaux (d'après leurs traits physiques)	/	50	12
Visualiser la morphologie des plantes (d'après leurs traits physiques, L-System)	/	20	13
Voir des changements climatiques pouvant influer sur les animaux	/	20	14

Sprint-1

Sprint goal

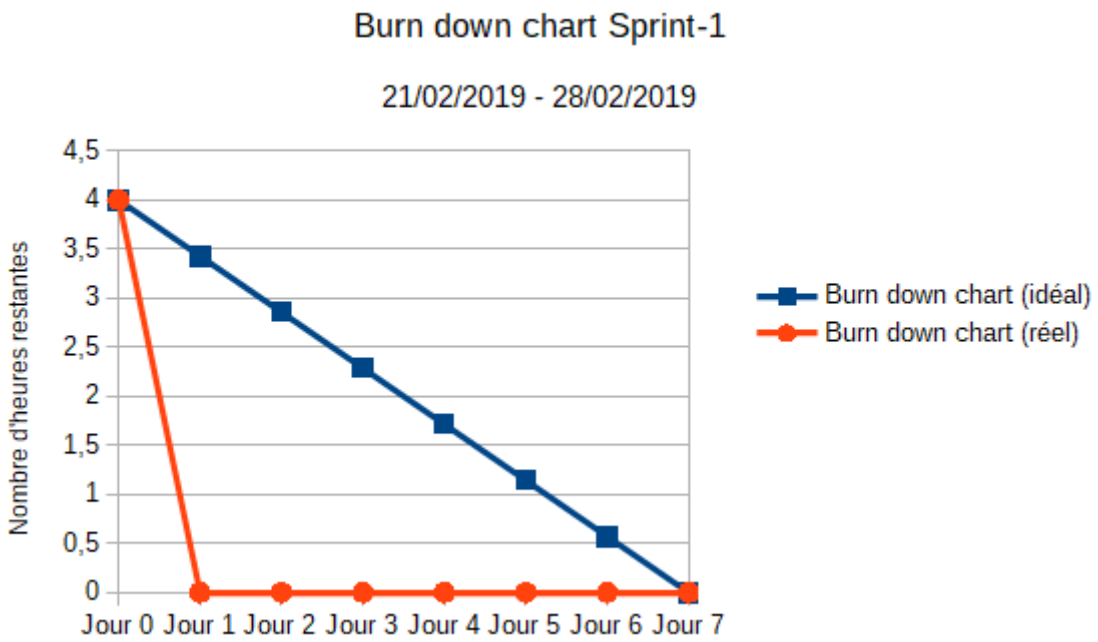
Du 21/02/2019 au 28/02/2019

Prendre en main les ressources données par le professeur. Décider du rendu graphique (3D ou 2D).

Sprint backlog

User stories	Tâches	Jour 1	Jour 2	Jour 3	Jour 4	En attente	En cours	Finie !
L'utilisateur veut voir le monde	Étudier l'option java 3D	2	0	0	0			X
	Étudier la 2D	2	0	0	0			X

Burn-down chart



Sprint review

Nous avons eu de nouvelles idées concernant le product backlog. Redéfinition de l'ordre d'exécution des tâches et de leurs priorités respectives.
Nous allons faire notre projet en 3D

Sprint retrospective

Tout va bien.

Sprint-2

Sprint goal

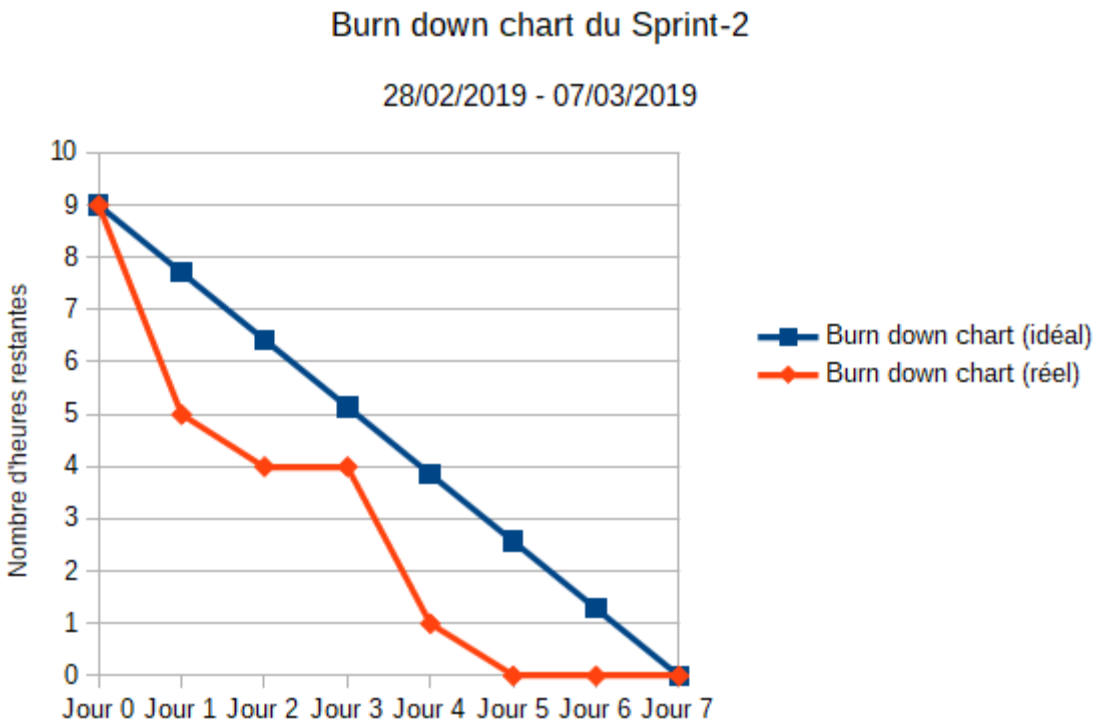
Du 28/02/2019 au 07/03/2019

Proposer un environnement plus riche en se familiarisant avec les outils donnés.

Sprint-backlog

User stories	Tâches	Jour							En attente	En cours	Finie !	Abandon
		1	2	3	4	5	6	7				
Approfondir la compréhension du fonctionnement de la démo	/	2	2	1	0	0	0	0			X	
Voir des arbres Simples en 3d	Implémenter des arbres	2	2	2	0	0	0	0			X	
	Positionner les arbres de manière quasi-random	3	0	0	0	0	0	0			X	
Voir de l'eau	Ajuster la couleur de l'eau	1	0	0	0	0	0	0			X	
Voir un Volcan	Étudier les solutions possibles	1	1	1	1	0	0	0			X	

Burn-down chart



Sprint review

Nous avons eu de nouvelles idées concernant le *product backlog*. Création des arbres. Prise en main des outils.

Nouvelle idée pour la création d'un volcan. Idée d'utilisation d'un automate cellulaire pour la formation du terrain.

Nous avons décidé que simplement utiliser un nombre aléatoire et une certaine probabilité pour placer un arbre ou non suffit.

Sprint retrospective

Impossibilité d'installer l'environnement chez aucun d'entre nous. Obligation de travailler à l'université.

Manque de rigueur sur le détail du *product backlog*.

Sprint-3

Sprint goal

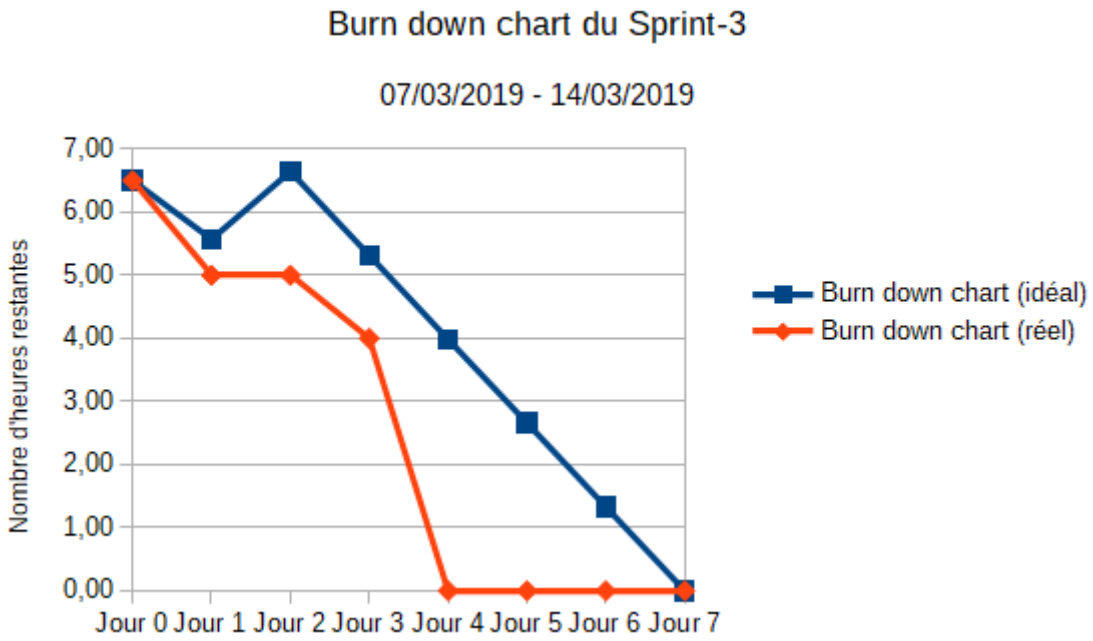
Du 07/03/2019 au 14/03/2019

Mettre en place les acteurs principaux du projet : les agents, le volcan, la terraformation.

Sprint-backlog

User stories	Tâches	Jour							En attente	En cours	Finie !	Abandon
		1	2	3	4	5	6	7				
Voir des arbres Simples en 3d	Changer les arbres en brousse	1	1	0	0	0	0	0			X	
	Implémenter des arbres	2	1	0	0	0	0	0			X	
	Créer des terrains ressemblants à une génération par Perlin	1/2	0	0	0	0	0	0			X	
Voir un Volcan	Créer un automate cellulaire créant un volcan de manière procédurale	2	2	2	1	0	0	0			X	
	Implanter l'automate	1	1	1	1	0	0	0			X	
Voir des animaux qui interagissent entre eux	Réfléchir à la mise en place de ce genre de système	0	0	2	2	0	0	0			X	

Burn-down chart



Sprint review

Nous avons décidé de faire de considérer les arbres comme des brousses, et d'implémenter un réel type d'arbre, dont le modèle sera plus grand.

Notre projet possède les fonctionnalités suivantes :

- Visualisation d'un monde en **3 dimensions ;**
- Ce monde est peuplé d'**herbes** et d'**arbres ;**
- Des **feux de forêt** peuvent changer l'aspect du monde ;
- La **création du terrain se fait aléatoirement** grâce à un automate cellulaire qui donne un résultat similaire à celui du **bruit de Perlin** ;
- Un **volcan** est généré de manière procédurale grâce à un automate cellulaire ;
- Un **système proie-prédateur simple** a été implémenté et attend d'être implanté.

Il reste encore beaucoup de choses à implémenter d'ici le 04 avril 2019.

Sprint retrospective

Du fait de nos incompatibilités horaires, nous sommes obligés de travailler de visu uniquement pendant les heures de cours dédiées au projet.

Après vérification de la méthodologie *scrum*, nous avons détecté un manquement sur la forme et le contenu des documents *scrum*. Une rectification a été nécessaire à l'occasion du rapport intermédiaire. Notamment, nous nous sommes rendu compte que les *sprint backlog* n'étaient pas conformes à la méthodologie *scrum*. Par conséquent, cette erreur d'appréciation nous a amené à sous-évaluer notre capacité de travail ce qui a résulté en une accumulation de retard pour le reste du projet, mettant celui-ci en péril.

Sprint-4

Sprint goal

Du 14/03/2019 au 21/03/2019

Finaliser le volcan en ajoutant la lave.

Ajout au projet d'une vraie valeur ajoutée avec des animaux qui évoluent.

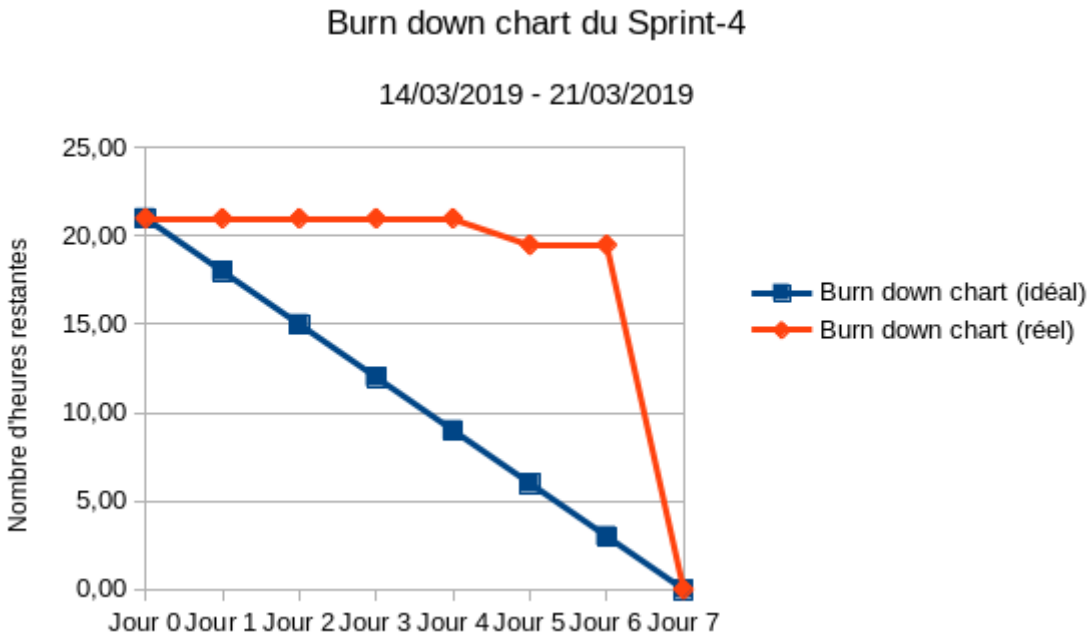
Régler la gestion des éléments supprimés/morts.

Sprint-backlog

User stories	Tâches	Jour							En attente	En cours	Finie !	Abandon
		1	2	3	4	5	6	7				
Voir de la lave	Implanter une nouvelle couche automatique pour la lave	6	6	6	6	6	6	0				X
Voir des animaux qui interagissent entre eux	Implémenter les prédateurs et les proies	3	3	3	3	1,5	1,5	0		X		
Voir les feux de forêt	Rendre le feu de forêt propre avec disparition des agents brûlé, et disparition des cases brûlées	3	3	3	3	3	3	0				X
	Brûler les agents animaux également	2	2	2	2	2	2	0				X

Voir l'évolution des animaux(Système ADN)	Créer le système ADN	1	1	1	1	1	1	0				X
	Implémenter la reproduction sexuée des animaux	2	2	2	2	2	2	0				X
	Implémenter l'analyse de l'environnement	3	3	3	3	3	3	0				X
	Mettre en place une évolution qui les protège du feu	1	1	1	1	1	1	0				X

Burn-down chart



Sprint review

La gestion des agents « animaux » se fait avec un automate cellulaire sur listes d'agents. Il faut modifier la méthode swap pour qu'elle passe correctement les agents d'un buffer à un autre.

Sprint retrospective

Ce sprint était particulier dans la mesure ou nous n'avons pas eu beaucoup de temps libre sur la fac pour travailler sur le projet. Le sprint-5 devrait être beaucoup plus productif.

Sprint-5

Sprint goal

Du 21/03/2019 au 28/03/2019

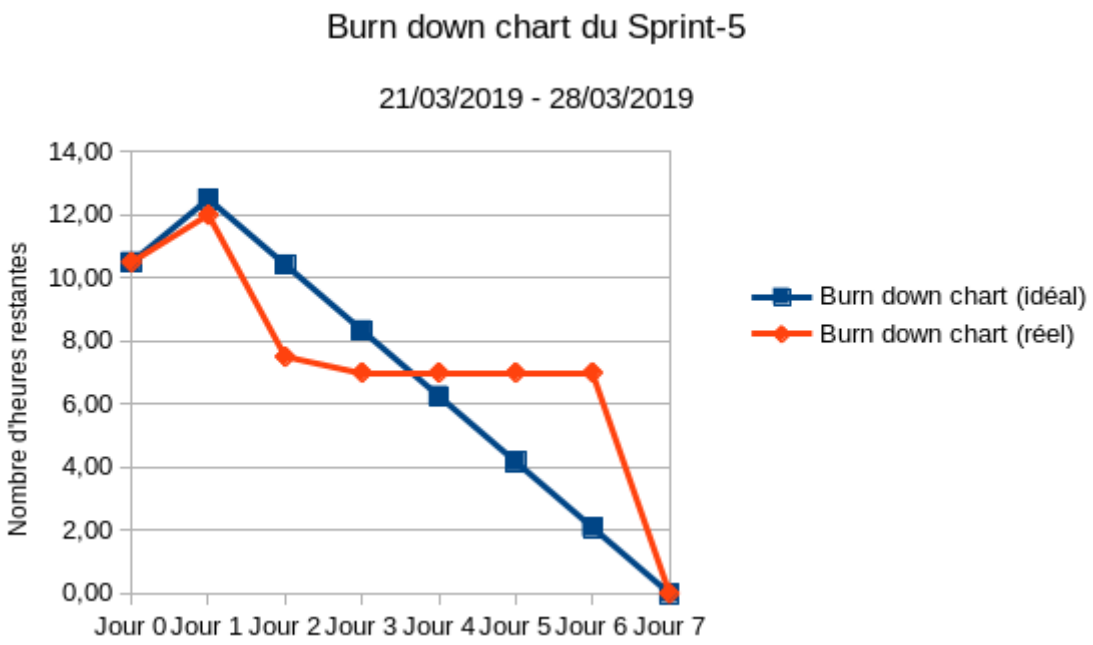
Finir l'implémentation du système proie-prédateur simple. Ajustement de la gestion des agents tués/supprimés.

Sprint-backlog

User stories	Tâches	Jour							En attente	En cours	Finie !	Abandon
		1	2	3	4	5	6	7				
Voir un écoulement de fluide	Implémenter un automate cellulaire de fluide	0	2	0	0	0	0	0			X	
Voir des animaux qui interagissent entre eux	Implémenter les prédateurs et les proies	1,5	1	0,5	0	0	0	0			X	
Voir les feux de forêt	Rendre le feu de forêt propre avec disparition des agents brûlé, et disparition des cases brûlées	3	3	3	3	3	3	0		X		
Voir l'évolution des animaux(Système ADN)	Créer le système ADN	1	1	1	1	1	1	0	X			

	Implémenter la reproduction sexuée des animaux (simple)	2	2	0	0	0	0	0			X	
	Implémenter l'analyse de l'environnement	3	3	3	3	3	3	0	X			

Burn-down chart



Sprint review

Nous avons décidé d'ajouter l'implémentation du fluide au sprint car il s'agit d'un prérequis du projet. Pour implémenter le fluide, nous avons décidé d'utiliser un automate cellulaire de *double*. Chaque case a pour valeur la quantité de fluide présent sur cette case.

Les proies et prédateurs sont gérés par un automate cellulaire. Les prédateurs mangent les proies quand leur énergie est trop basse. Les proies et prédateurs on une probabilité de se mettre en mode « accouplement ». Ils possèdent une variable vieillesse. Au delà d'un certain âge, ils meurent ; ceci nous permet de ne pas inonder le monde de créatures et donc de conserver un nombre de FPS convenable.

Sprint retrospective

RAS.

Sprint-6

Sprint goal

Du 28/03/2019 au 04/04/2019

Développer l'originalité de notre projet : une vie artificielle en perpétuelle ré-adaptation.

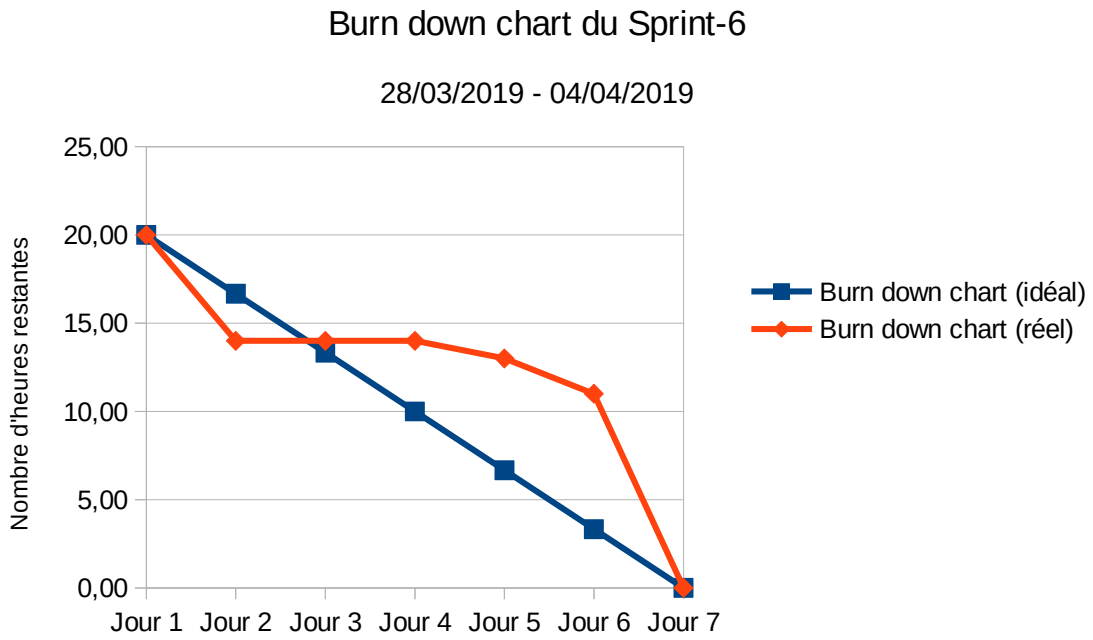
Sprint-backlog

User stories	Tâches	Jour							En attente	En cours	Finie !	Abandon
		1	2	3	4	5	6	7				
Voir les feux de forêt	Rendre le feu de forêt propre avec disparition des agents brûlé, et disparition des cases brûlées	3	2	2	2	2	0	0			X	
	Faire que les végétaux sont contenus et gérés par CellularAutomataAgents	2	2	2	2	2	2	0			X	
Voir des prédateurs et des proies	Implémenter la consommation de végétaux pour les proies	2	2	2	2	1	1	0			X	
	Affiner la fonction de recherche d'agent	2	0	0	0	0	0	0			X	

	Factoriser cette fonction de recherche à toutes les méthodes l'utilisant	2	0	0	0	0	0	0			X	
	Implémenter un comportement de fuite pour les proies	2	0	0	0	0	0	0			X	
Voir l'évolution des animaux(Système ADN)	Déterminer des traits critiques à la survie des animaux (exemples : vitesse de course, force physique)	2	1	1	1	1	1	0				X
	Implémenter des contraintes de trait : les animaux ne peuvent pas être super-fort sans que ça aie un coût sur leur nourriture consommée par exemple.	2	2	2	2	2	2	0				X

	Implémenter la méiose : un système de probabilité de fréquence et d'amplitude des mutations génétiques lors de la reproduction	3	3	3	3	3	3	0				X
	Tester et affiner	2	2	2	2	2	2	0				X

Burn-down chart



Sprint review

Ajout des fonctionnalités suivantes :

- Les proies se nourrissent d'herbes (exclusivement) et entraînent leur disparition ;
- Les végétaux (arbres et herbes) sont gérés à la manière d'un automate cellulaire :
  - Lorsque la case courante contient de la lave, le végétal s'enflamme ;
  - Lorsqu'un végétal brûle, il perd des points de vie, qui lorsqu'ils sont à 0, entraînent la mise du végétal à l'état « brûlé » ;
  - Lorsqu'un végétal brûle, si il y a un animal sur une case adjacente (voisinage de Moore), le met à l'état « brûle » ;
  - Un végétal brûlé apparaît en noir, il a une probabilité de se mettre en mode « mort » ;
  - Un végétal mort est supprimé par la structure d'automate cellulaire le contenant ;
  - Un végétal vivant a une faible probabilité de devenir un arbre ;
  - Un végétal vivant a une probabilité de s'enflammer si un végétal adjacent brûle ;
  - Une herbe vivante a une probabilité de générer une herbe sur une case adjacente.
- Les animaux peuvent brûler ;
- Les animaux perdent des points de vie à chaque itération lorsqu'ils brûlent ;
- Lorsque les prédateurs prennent en chasse une proie, celles-ci le remarquent et fuient.

Sprint retrospective

RAS.

B Manuel d'installation

Pour lancer notre monde artificiel, vous aurez besoin de Eclipse. La version que nous utilisons est : Oxygen.2 Release (4.7.2).

Téléchargez l'ensemble du répertoire github, et placez l'archive où bon vous semble. Décompressez-la.

Supprimez les dossiers suivants :

- /projet-2I013-master/Images,
- /projet-2I013-master/javaImages,
- /projet-2I013-master/applications,
- /projet-2I013-master/cellularautomata,
- /projet-2I013-master/graphics,
- /projet-2I013-master/interfaces,
- /projet-2I013-master/landscapegenerator,
- et enfin /projet-2I013-master/objects.

Téléchargez le backup local de JOGL : http://pages.isir.upmc.fr/~bredeche/Teaching/2i013/2018-2019/JOGL.zip

Décompressez cette archive à la racine du projet, dans un dossier que vous créerez, nommez-le javalib.

Ouvrez eclipse. Suivez le tutoriel Tutorial-javaOpenGL.pdf fourni, mais ignorez la partie nommée "Ajouter JOGL à votre projet".

Suivez plutôt les instructions suivantes :

Importez dans Eclipse le dossier /projet-2I013-master/3D/WorldOfCells en tant que projet. (file->import->projects from folder)

Dans Eclipse, cliquez droit sur le projet (le dossier bleu avec la lettre J) dans le panneau à gauche, et sélectionnez Build Path -> Add Library... . Dans la fenêtre, sélectionnez ‘User Library’ puis cochez ‘jogl-2.0’. Cliquez sur Finish.

JOGL est maintenant lié à votre projet.

TRES IMPORTANT : dans eclipse, supprimez du projet le package landscapegenerator.ca_landscape

Des avertissements s'affichent ça et là : ignorez-les.

Ouvrez le fichier applications.simpleworld.MyEcosystem
et lancez-le.

C Scénario d'utilisation

Lancez la simulation. Vous voyez le monde tourner autour de vous. Stabilisez la caméra en appuyant plusieurs fois sur « q ». « q » et « d » servent à tourner à gauche et à droite. Maintenant, utilisez les flèches directionnelles pour vous déplacer. Vous vous sentirez sûrement mal à l'aise de vous apercevoir que les flèches ne vous font pas avancer comme espéré. Appuyez sur la flèche du haut, et observez la direction dans laquelle vous vous dirigez. Tournez la caméra grâce à q et d de telle sorte que cette direction soit en face de vous. Si vous arrivez à vous y faire, vous pouvez passer l'étape précédente. Vous avez sûrement manqué beaucoup de choses en manipulant les touches. Relancez la simulation. Une fois stabilisé, observez autour de vous. Pouvez-vous voir le volcan ? Approchez-vous. Arrivez-vous à trouver la lave qui en découle ? Une fois fait, regardez attentivement l'écoulement de la lave. En passant, elle a sûrement déjà enflammé plusieurs herbes, voire des arbres ! Suivez la ligne de front du feu, regardez le se propager. Avez-vous remarqué que certaines herbes ne sont pas touchées par les flammes ? Recueillez-vous un instant sur les herbes calcinées.

Il ne vous a sûrement pas échappé que notre monde n'est pas uniquement habité par des végétaux. Avez-vous vu ces petits cubes blancs se balader ? Regardez les prendre feu quand ils passent à proximité d'un feu. Pratiques ces barres de vie qui apparaissent au dessus d'eux quand ils perdent de la vie, non ? Étudions un peu plus ces agents mobiles. Relancez la simulation. Appuyez sur « v » et activez la vue aérienne. De petits carrés violets et bleus bougent. Les violets sont les prédateurs, et les bleus les proies. Observez le va et vient des prédateurs. Remarquez la manière dont tout une zone d'herbe est tondue par les proies. Attendez un peu. Le nombre d'herbes diminue. Quand est-il des proies ? Des prédateurs ? Remarquez la manière dont les proies commencent à se disperser et à se grouper autour des arbres. Les arbres sont générateurs d'herbes. Le nombre de proies a diminué, mais les prédateurs ne les ont pas pour autant surpassés en nombre (ou alors vous êtes malchanceux). À partir de maintenant, le nombre de proies grandira puis diminuera, et ainsi fera celui des prédateurs, indéfiniement.