Un projet basé sur WorldOfCells de
Jarski Paul – Responsable Agents
Sok Chanattan – Responsable Écosystème

Rapport de projet
[LU2IN013]
Wonderland

Aide aux touches :

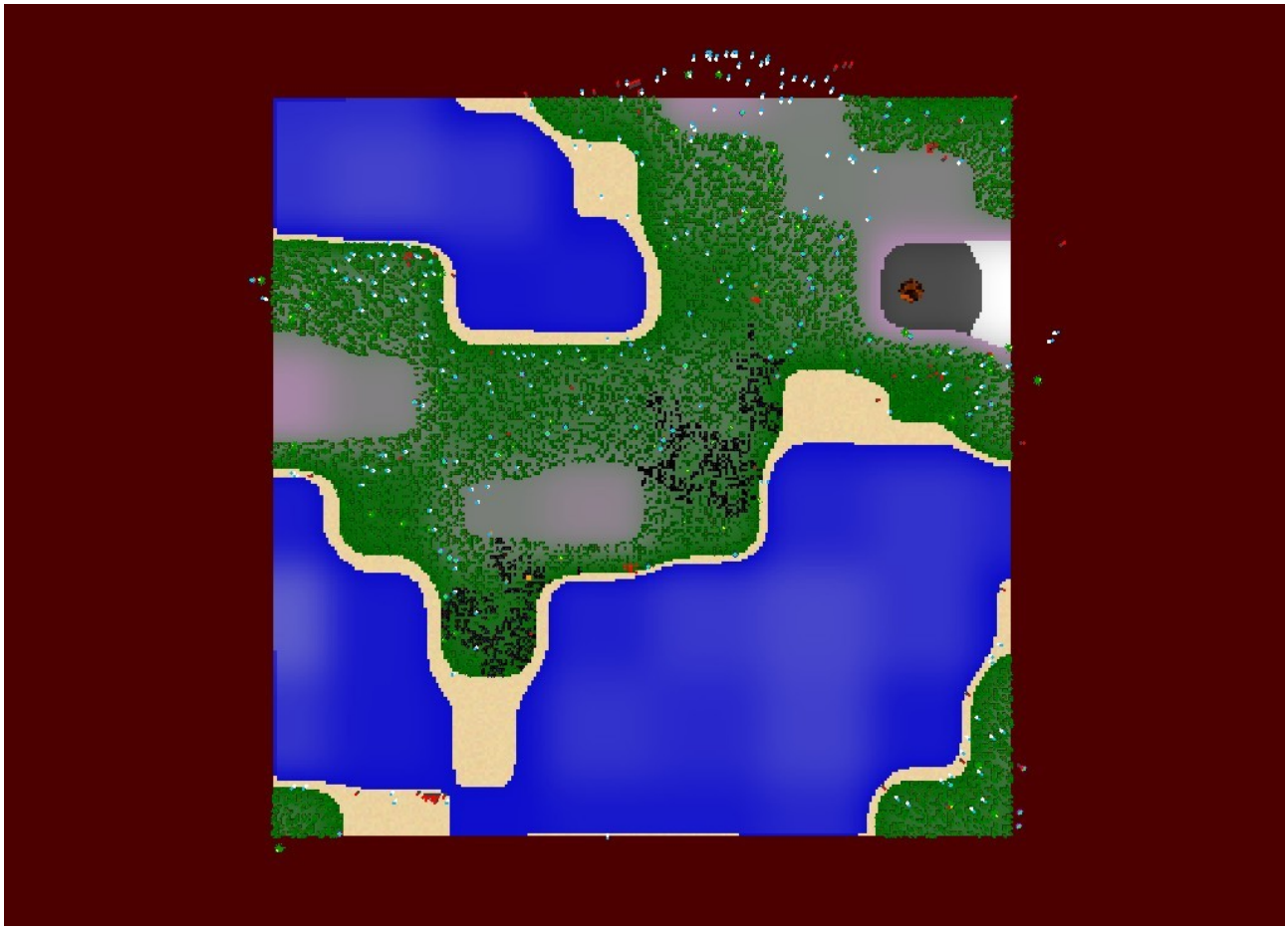
[V] - vue de la carte
[O] – affichage objets
[A/Z] – augmenter/diminuer la hauteur globale du terrain
[L/M] – se déplacer selon l'axe z vers le haut ou vers le bas
flèches directionnelles – se déplacer dans le monde
[Q/D] – rotation gauche ou droite dans le monde
[U/I] – accélérer/décélérer le temps
[B] – afficher les fps
[X] – faire entrer en éruption le volcan
[T] – faire neiger
[Y] – faire pleuvoir
[J] – faire l'ensoleillement
[G] – faire apparaître/disparaître Godzilla
[P] – invoquer un prédateur
[K] – invoquer une proie
[H] – afficher/enlever l'aide sur l'interface graphique
[F] – activer/désactiver le mode PPTI pour les problèmes d'affichage (arbres/feuilles)
[échap] - quitter l'écosystème

Événements écosystème :

L'écosystème se rapporte à des bases très simples : le terrain, le temps, la météo, les catastrophes naturelles, les éléments esthétiques.

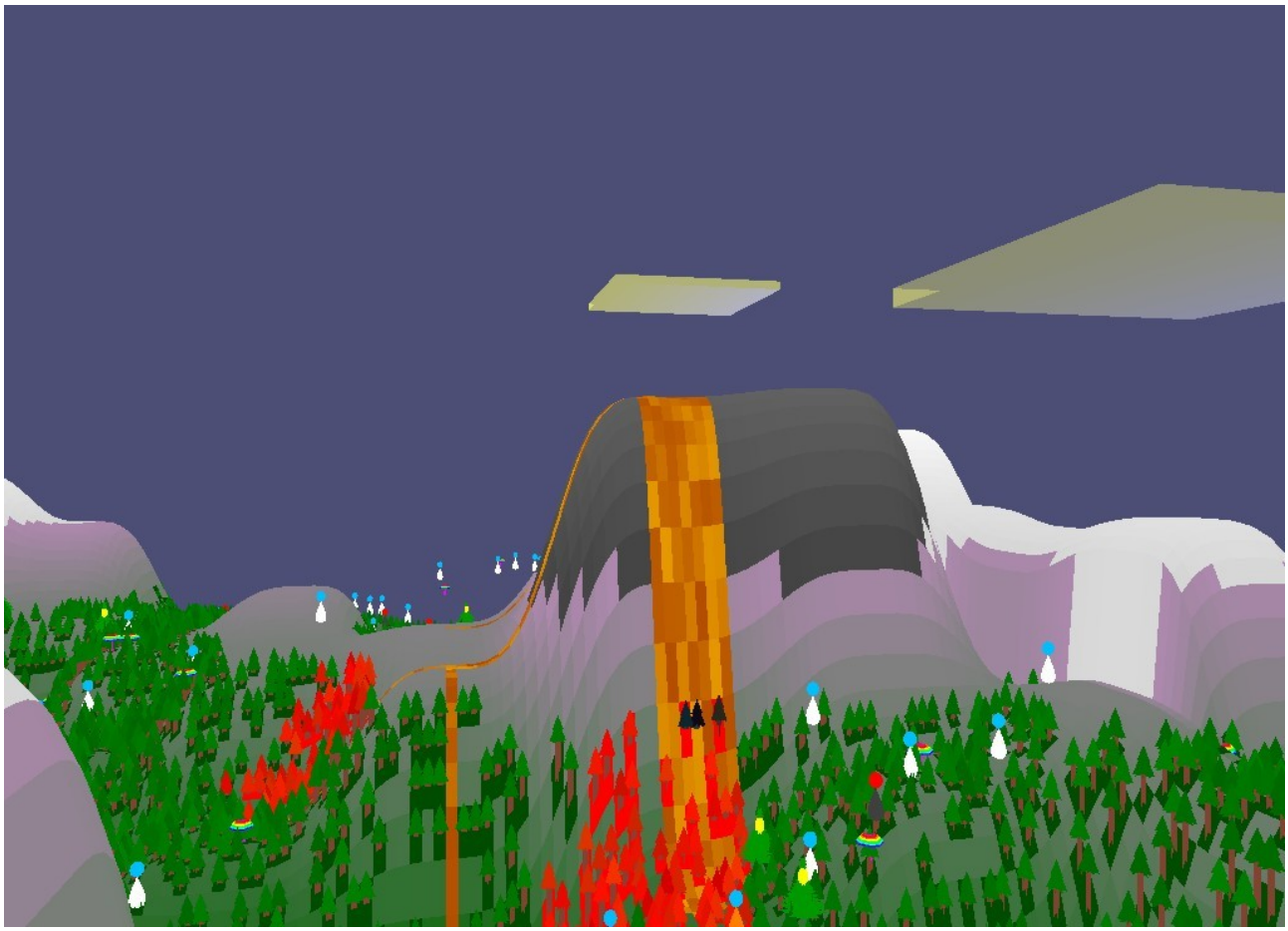
Note : le feu de forêt étant l'implémentation la plus basique du projet elle ne sera pas détaillée ici, de plus les seules modifications faites à la première implémentation sont le rallongement du temps de brûlage et une propagation plus large en regardant les voisins sur le schéma d'une croix de longueur 2.

I) Carte



La carte est générée aléatoirement par le bruit de Perlin, selon les valeurs des hauteurs nous avons implémenté de la neige à plus haute altitude sur les montagnes et de l'eau ainsi que du sable à plus basse altitude, le reste est de la terre, de l'herbe.
Par manque de temps nous n'avons pas pu développer le bruit de Perlin périodique qui oblige les falaises aux extrémités visibles lorsqu'on se déplace dans le monde en 3D.

II) Volcan



Le volcan est en vérité une simple montagne générée par le bruit de Perlin, cette montagne est recherchée dans un intervalle bien défini et ensuite de la lave inanimée va s'installer dans le creux au centre de cette montagne.
Lors de l'éruption la lave va s'écouler sur une certaine distance jusqu'à soit atteindre l'eau soit au bout d'un certain temps pour se changer en roche. La lave brûlera tout sur son passage.
Le volcan a une infime chance de se déclencher seul cependant il est activable par l'utilisateur et peut ré-entrer en éruption à volonté.

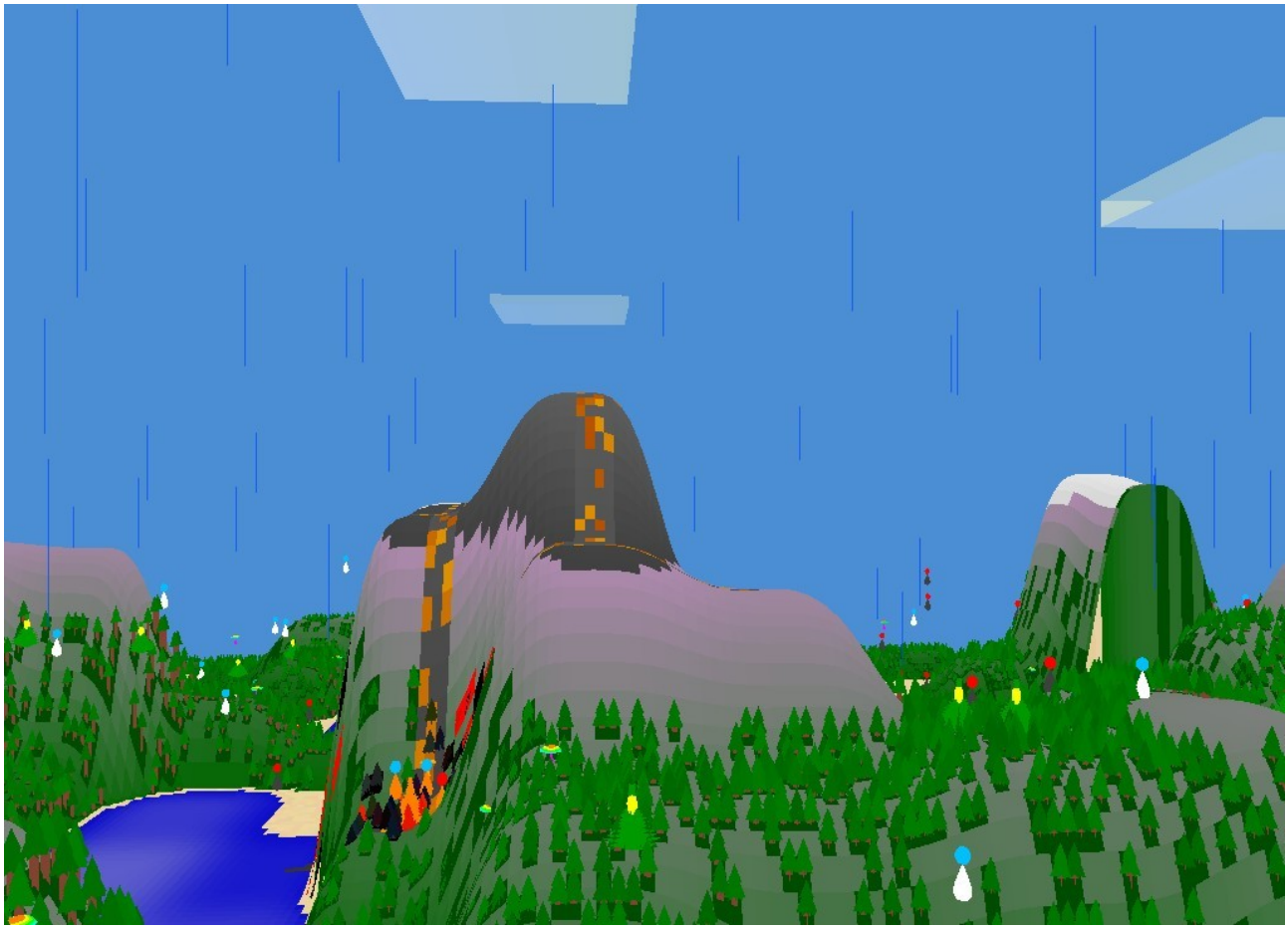
III) Nuages

Les nuages ne sont visibles que dans le monde et non sur la carte par raison de vision.
Ce sont de larges rectangles plus ou moins transparents qui avancent à des vitesses variées et sont à différentes altitudes. Ils se déplacent en boucle dans le monde selon l'axe x.
Leur nombre est prédéfini au début. Le jour ils sont blancs clairs et plus la nuit avance plus ils deviennent jaunes pour simuler le coucher du soleil.

IV) Pluie/Neige/Ensoleillé

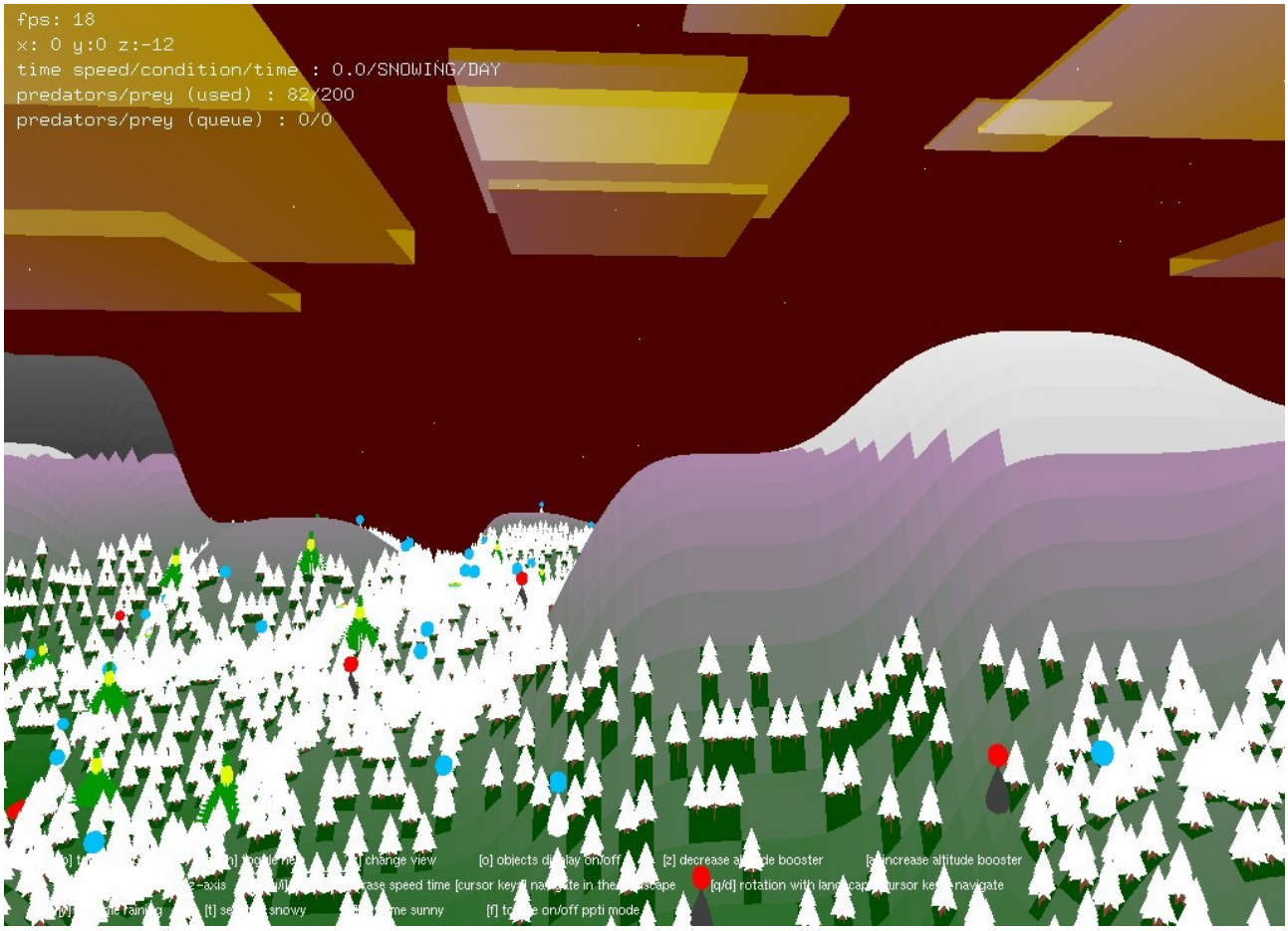
Les conditions météorologiques influent le monde et sont au nombre de 3 :
la pluie permet d'éteindre les incendies et éventuellement d'arrêter la propagation de la lave mais a aussi des effets sur les agents. Il est en va de même pour la neige mais elle ne fait rien à la forêt et à la lave. Le temps ensoleillé est le temps par défaut où rien ne tombe.
Les conditions météorologiques changent aléatoirement toutes les 60 secondes par défaut.

Pluie qui éteint un feu de forêt :



On voit que la lave se transforme en roche progressivement au dépend d'une certaine probabilité et donc au bout d'un certain temps, la même chose arrive aux arbres qui passent à l'état carbonisé plus rapidement.

Neige qui rend les arbres blancs :



La neige est assez fine mais on voit des particules blanches.
Elle rend les arbres blancs et affecte certains agents, cela sera vu dans la section événements agents.

V) Temps/Modification du temps

Le temps est implémenté pour influencer directement les agents dans un cycle jour/nuit.

Le temps est modifiable en ayant la possibilité de l'accélérer ou le ralentir (voir screen précédent en haut à gauche sur l'interface utilisateur : time speed) avec les touches u/i et cela permet de voir défiler certaines choses plus vites (les nuages, le ciel, le comportement de certains agents) mais pas toutes. On peut également changer les conditions météorologiques selon les touches t (neige), y (pluie), j (ensoleillé).

VI) Soleil et ciel

Le soleil est une sphère jaune qui tourne sur un même axe autour du monde au dépend de la vitesse du temps, elle permet de se repérer dans le cycle jour/nuit (voir interface graphique en haut à gauche : time est encore à day) et lorsque le soleil n'est plus visible la nuit commence.
Le ciel tourne progressivement au dépend de la position du soleil (du temps) du bleu au rouge.

Événements agents :

Dans ce monde il y a trois types d'agents; les prédateurs, les proies et Godzilla; et deux types de plantes dynamiques, les champignons[1] et les ananas. Le travail de création de ces formes de vie artificielle avait un aspect dessin et un aspect comportement.

Dessin :

On a exploré des possibilités de créer des objets avec blender, mais l'intégration de ces objets dans notre écosystème était trop compliqué. Heureusement une contrainte peut être une source d'inspiration aussi, et au fur et à mesure on a développé des outils d'affichage qui se trouvent dans la classe DisplayToolbox. Maintenant on peut dessiner des objets lisses et radialement symétriques sans trop de difficulté, et grâce à cela on a pu implémenter un thème échecs : les agents ressemblent à des pions et « Godzilla » ressemble à une dame. Cette facilité de création d'objets ronds explique aussi la présence de champignons et pas de fleurs.

I) Champignons

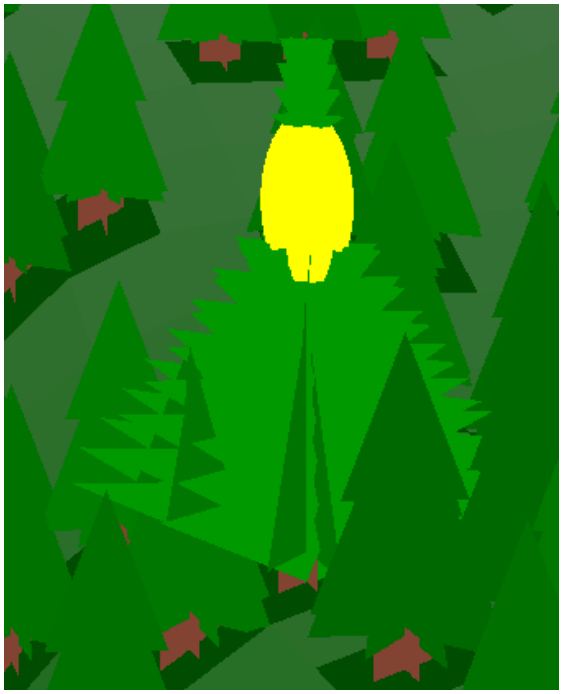
Les champignons croissent de taille 0 jusqu'à taille 11, toujours avec le nombre de bandes colorées égale à la taille. Une fois que tous les 11 bandes sont présentes, les couleurs changent en cycle grâce à une méthode d'incrémentation de couleur qui parcourt une suite circulaire de 12 couleurs. Puisque la taille est un moins que le nombre de couleurs, il y a 12 combinaisons de couleurs possibles et l'effet est assez joli.

Quand une proie mange un champignon, sa taille est diminuée de 1.

1 Bien sûr un champignon n'est pas biologiquement une plante, mais en ce qui concerne l'hiérarchie des classes, il peut en être une.

II) Ananas

Les ananas commencent comme des arbustes, qui font pousser des fruits. La taille d'un ananas varie entre 0 et 10 : sans fruit à taille 0 et avec un fruit grand et jaune à taille 10. Le nombre de feuilles sur le fruit est égale à la partie entière de la taille divisé par deux, moins un. Les fruits deviennent de plus en plus jaune lorsqu'ils grandissent. Quand une proie mange un fruit, la taille recommence à 0.

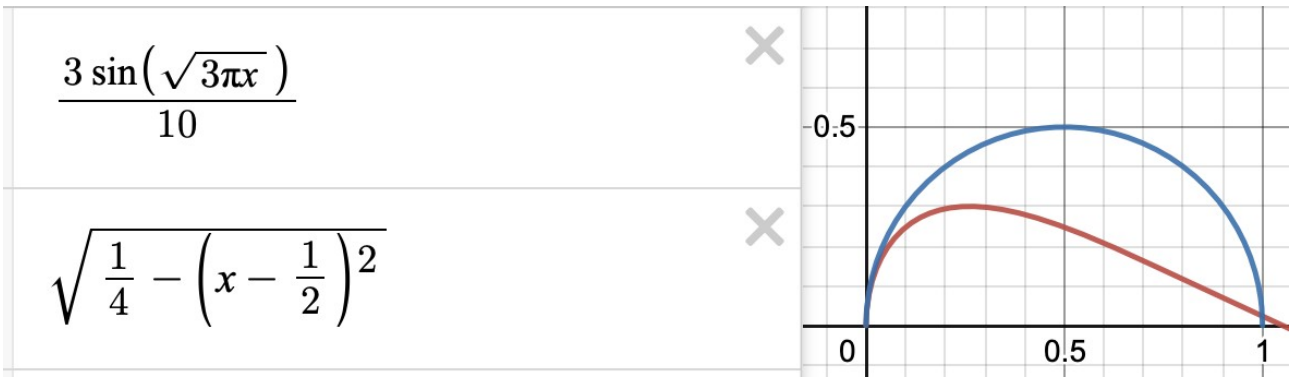


III) Arbres

Les feuilles des arbres et des ananas sont construits avec une méthode qui dessine des feuilles de la même manière que les arbres triangulaires au début du projet, mais avec des fonctions trigonométriques qui permettent de dessiner un nombre pair de feuilles dans un cercle. Malheureusement ils ont les mêmes problèmes d'affichage que les arbres du début. La touche [F] permet de visualiser les arbres dans ce cas.

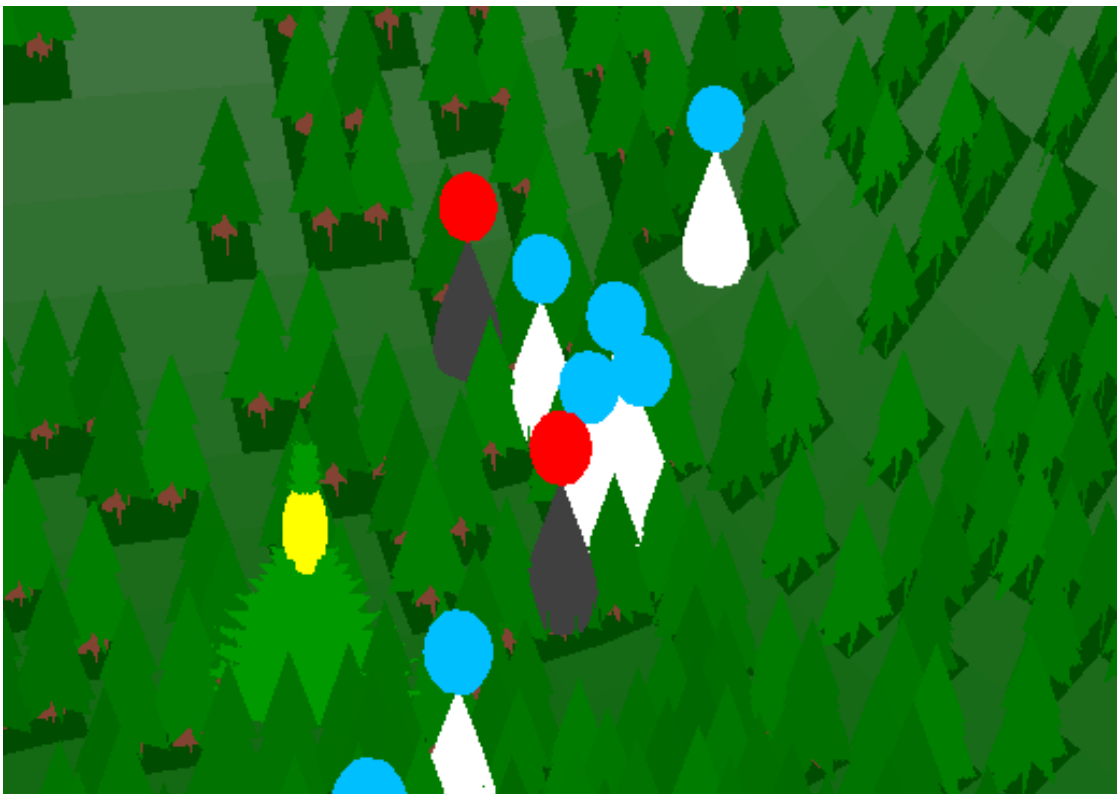
IV) Prédateurs

Les prédateurs et proies sont construits à partir de deux fonctions sur l'intervalle [0,1] et la fonction DisplayToolbox.drawOctagonalPrism() qui permet de dessiner des formes quasi-rondes.



Les prédateurs ont une vision triangulaire orienté devant l'agent avec une pointe sur la case ou l'agent se trouve. Les prédateurs ont la reproduction sexuée et utilisent cette vision pour la recherche de victimes et compagnons, mais elle n'est pas employée à chaque tour. Si le prédateur à déjà repéré un cible, il le poursuit sans devoir rechercher encore. Cette approche permet les mêmes comportements de chasse qu'un algorithme qui fait la recherche à chaque tour, mais c'est beaucoup moins couteux en temps de calcul.

L'arbre de décisions est détaillé et commenté dans la fonction step() de la classe Predator. Cela marche avec un entier qui vaut -1 si le prochain déplacement n'a pas encore été déterminé, -2 si le Predator ne va pas bouger ou un entier entre 0 et 3 selon la direction choisie.

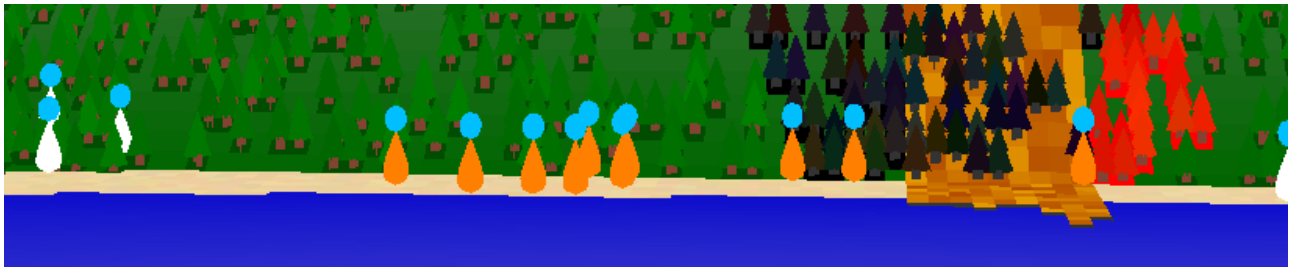


V) Proies

Les proies ont un champs de vision autour d'eux qui est parcouru en spirale pour trouver le prédateur le plus proche. Les proies doivent aussi faire la recherche des prédateurs à chaque tour, car il peut y en avoir plusieurs aux alentours. L'arbre de décision des proies est détaillé et commenté dans la fonction step(). La reproduction des proies est asexuée, donc ils n'ont pas besoin de chercher des compagnons.

VI) Propagation d'incendies

Quand un agent est touché par la lave ou une incendie, il devient orange et sa vitesse augmente. Il peut alors propager l'incendie et aux plantes/arbres et aux autres agents dans le voisinage von Neumann.



Par contre, juste comme un écoulement de lave est ralenti par la pluie ou la neige, les agents ont aussi une chance d'être éteint par la précipitation.

VII) Contraintes sur les populations

Puisque l'enrichissement de l'écosystème est couteux en temps de calcul, il faut borner le nombre maximal d'agents. Pour faire cela, on a des classes Pool qui mettent des bornes supérieures sur les populations, et en plus, ils permettent d'optimiser la création de nouveaux agents. Quand un agent meurt, il est transféré à un ArrayList d'attente. Les nouveaux nés sont tiré de cette liste et créés de nouveau uniquement si cette liste est vide.

L'utilisateur peut aussi gérer les populations avec les touches [P] et [K] et avec Godzilla, qui est invoqué par la touche [G].

VIII) Godzilla

Godzilla est une classe singleton, qui peut être invoqué par l'utilisateur à volonté. Il parcourt le monde de façon aléatoire, avec une préférence pour la direction tout droit et sa haleine radioactive tue des forêts et des agents régulièrement