Rapport de projet
IEVA

Enseignant : Éric MAISEL

Projet réalisé par Pierre LE DEZ

Table des matières

Pierre LE DEZ
p5ledez@enib.fr

I – Architecture Générale

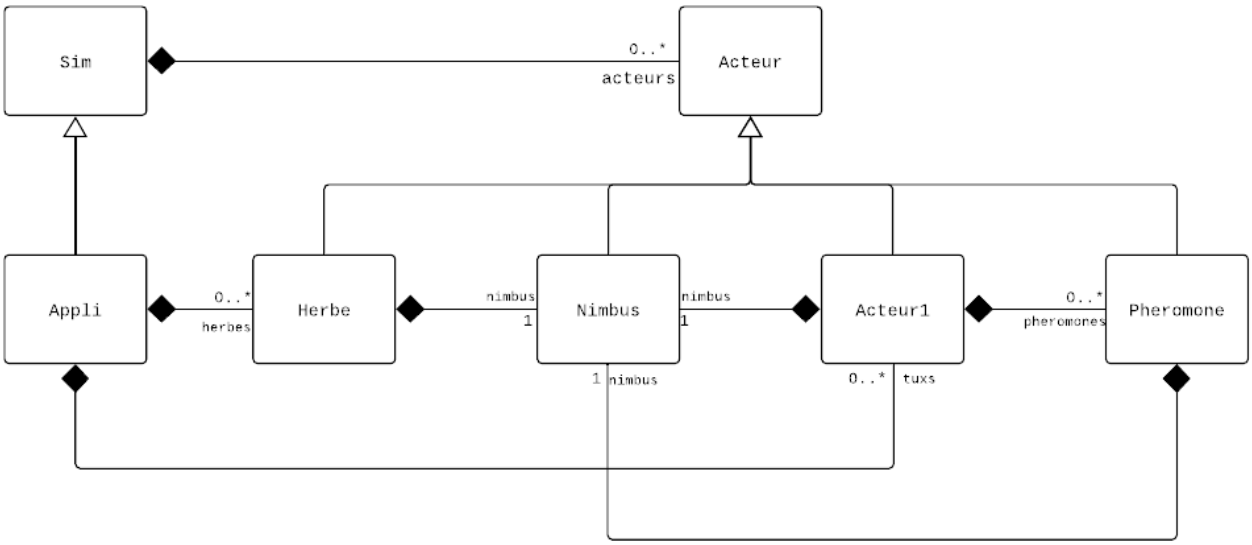

Figure 1 : Architecture générale

La figure ci-dessus montre l'architecture générale du projet. On retrouve les classes présentes dès le début ainsi que les classes ajoutées (Pheromone et Nimbus). La classe Appli peut accéder aux différents acteurs présents car elle hérite de « Sim ». Pour des raisons de facilité, l'application accède aux Herbes et aux Pingouins par le biais de 2 listes distinctes au lieu de la liste des acteurs. Chaque acteur possède un nimbus et les acteurs Pingouins (Acteur1) possèdent également des phéromones.

II – L'environnement

Lorsque l'application est créée dans le navigateur, la première chose à faire avant de créer les différents acteurs consiste à générer l'environnement dans lequel ces derniers évolueront. L'environnement contient un plan horizontal servant de sol aux acteurs et également des touffes d'herbes.

Pierre LE DEZ
p5ledez@enib.fr

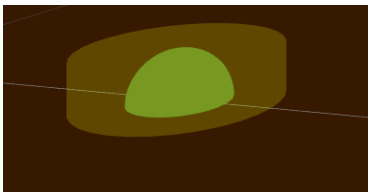

Figure 2: Touffe d'herbe avec son nimbus visible

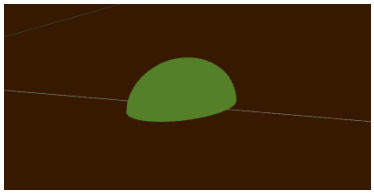

Figure 3: Touffe d'herbe avec son nimbus caché

Les 2 figures ci-dessus illustrent une touffe d'herbe avec (Fig. 2) et sans (Fig. 3) l'affichage du nimbus de l'herbe. Les touffes sont représentées par des sphères vertes à moitié enfoncées dans le sol.

```
//generate grass
max = 50;
min = -50;
nb_grass = 200;
for (let i=0; i < nb_grass; i++) {
    x = Math.floor(Math.random() * (max - min) ) + min;
    z = Math.floor(Math.random() * (max - min) ) + min;
    let temp_h = new Herbe("herbe"+i.toString(), {couleur: 0xaaff55}, this);
    this.addActeur(temp_h);
    temp_h.setPosition(x, 0, z);
    temp_h.nimbus.placeNimbus();
    this.herbes.push(temp_h);
}
```

Figure 4: Génération des touffes d'herbes

L'environnement dans son état initial contient 200 touffes d'herbes dont la position est choisie aléatoirement. **Le code présent dans la Fig. 4 répond à la Q1.**

III – Les acteurs

Dans cette partie, nous allons rentrer plus en détails sur les comportements ajoutés aux différents acteurs et plus particulièrement aux pingouins (classe Acteur1).

Pierre LE DEZ
p5ledez@enib.fr

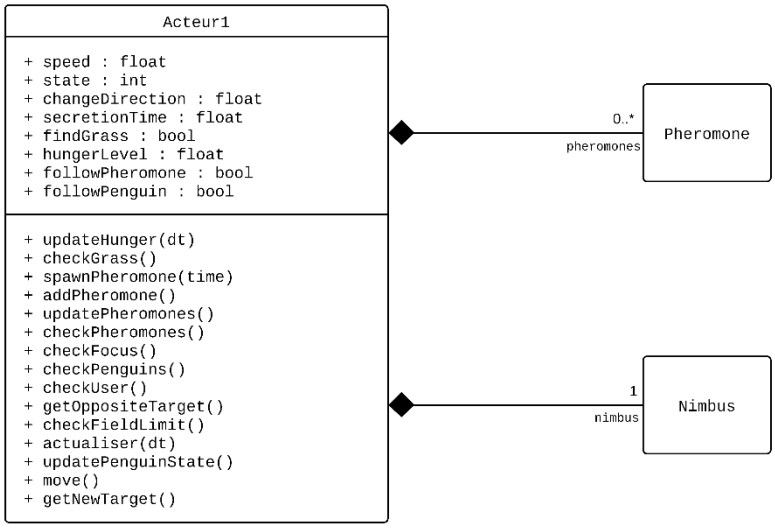

Figure 5: Classe Acteur1

A – Machine à états

La Fig. 5 contient une déclaration plus précise que de la classe Acteur1 que dans la Fig. 1 de la partie I. J'ai choisi de structurer les acteurs « Pingouins » comme une machine à état. Cette dernière comporte 3 états : « Fuir l'utilisateur », « Se Nourrir » et « Se Promener ».

L'état d'un acteur Pingouin (Acteur1) est mis à jour au début de la fonction « actualiser », l'état est ensuite utilisé dans la suite de la fonction pour décider des actions que qu'il va réaliser. La Fig. 6 montre comment l'état est mis à jour.

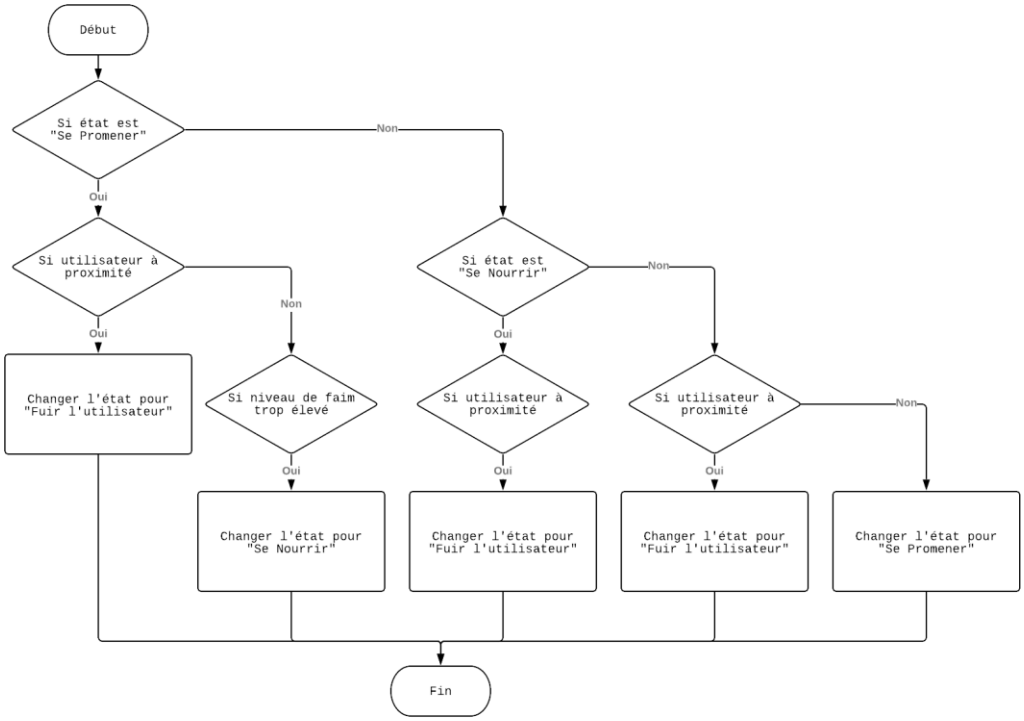

Figure 6: Mise à jour de l'état d'un acteur Pingouin

Pierre LE DEZ
p5ledez@enib.fr

Lors de la mise à jour de l'état d'un Pingouin, un système de priorité est à prendre en compte. L'état « Fuir l'utilisateur » est le plus prioritaire, ensuite c'est « Se Nourrir ». Le dernier état est « Se Promener ».

Dans l'état « Fuir l'utilisateur », le pingouin va changer de direction lorsque son nimbus rentrera en contact avec le nimbus de la caméra de l'utilisateur.

Au début de la méthode « actualiser », le niveau de faim du pingouin est mis à jour, il augmente légèrement en fonction du pas de temps « dt ». Lorsque ce niveau est supérieur à un entier fixée dans la déclaration de la classe Acteur1, le pingouin passe dans l'état « Se Nourrir ».

Dans ce dernier état, un pingouin se promène de manière aléatoire et change de direction de temps en temps (en fonction de l'horloge de la simulation). Si une phéromone rentre en contact avec le nimbus du pingouin, il ne va plus se déplacer aléatoirement mais suivre les phéromones dégagées par un autre pingouin. Ensuite il se peut qu'un autre pingouin traverse le nimbus de l'objet courant, alors il va le prendre pour cible. La dernière étape consiste à regarder si d'autres pingouins se trouvent dans le champ de vision de notre pingouin, si c'est le cas, il en prend un pour cible. Les 2 cas sont différenciés (nimbus et champ de vision = focus) car le champ de vision est plus long que le rayon du nimbus. La Fig. 10 en Annexes montre le déroulement global de la méthode « actualiser ».

B – Déplacement des pingouins

```
Acteur1.prototype.getNewTarget = function(dt) {
    x = Math.floor(Math.random() * 100) - 50;
    z = Math.floor(Math.random() * 100) - 50;
    this.target = {"x": x, "y": 0, "z": z};
    this.objet3d.lookAt(this.target.x, 0, this.target.z);
}
```

Figure 7: Fonction permettant de récupérer une nouvelle cible dans le champ

```
Acteur1.prototype.checkFieldLimit = function(dt){
    let pos = this.getPosition();
    if (pos.x > 50 || pos.x < -50 || pos.z > 50 || pos.z < -50) {
        this.getNewTarget();
        this.objet3d.lookAt(this.target.x, 0, this.target.z);
    }
}
```

Figure 8: Fonction permettant de vérifier les limites du champ

Les 2 fonctions ci-dessus (Fig. 7 et Fig. 8) sont utilisés lorsque l'état d'un pingouin est « Se Promener » et qu'il ne suit pas de phéromone ni un autre pingouin.

Pierre LE DEZ
p5ledez@enib.fr

La première permet de définir une nouvelle cible jusqu'à laquelle se rendre dans le champ et la deuxième permet de vérifier qu'il ne sort pas du champ. **Ces 2 méthodes répondent à la Q2.**

C – Nimbus

Tous les acteurs (Herbe, Pheromone et Acteur1) sont dotés d'un nimbus. Le nimbus des touffes d'herbes et des pingouins est un cylindre de rayon variable (un rayon défini pour l'herbe et un autre pour les pingouins) et de hauteur égale à la hauteur des acteurs. Les phéromones ont un nimbus sphérique comme ce sont elles aussi des sphères. Pour finir la caméra de l'utilisateur a également un nimbus cylindrique.

La classe Acteur1 a une méthode pour chaque type de classe : checkGrass, checkPheromones, checkUser et checkPenguins. Ces différentes méthodes effectuent le même calcul. La liste des acteurs est tout d'abord parcourue (« herbes » pour checkGrass, « pheromones » et « tuxs » pour checkPheromones), ensuite la distance entre le pingouin et un acteur (une touffe d'herbe, un autre pingouin ou une phéromone) est comparé au rayon des 2 nimbus. Si la distance est plus petite, alors il y a croisement des deux nimbus. Pour l'utilisateur, le calcul est le même excepté qu'il n'y a pas de parcours de liste. **Les méthodes « checkGrass », « checkUser », « checkPheromones » et « checkPenguins » répondent à la Q3.**

J'ai choisi 2 types de nimbus, un nimbus cylindrique pour la caméra de l'utilisateur, les touffes d'herbes et les pingouins car ces derniers peuvent être assimilés à une entité touchant le sol. Concernant les phéromones, j'ai choisi un nimbus sphérique qui entoure la phéromone, de ce fait on conserve la forme de particule de la phéromone. **Cela répond à la Q4.**

D – Focus

Comme dit précédemment, un pingouin « A » a 2 manières de détecter un pingouin « B » pour le suivre, la première consiste en un contact du pingouin « B », ou de son nimbus, avec le nimbus du pingouin « A ». La deuxième manière est lorsque le pingouin « A » voit le pingouin « B » dans son champ de vision. La notion que j'appelle champ de vision correspond à la notion de focus présente dans l'énoncé. Les pingouins ont un champ de vision de 180° pour pouvoir facilement détecter leurs congénères. **La méthode « checkFocus » dans le code répond à la Q5.**

Pierre LE DEZ
p5ledez@enib.fr

E – Phéromone

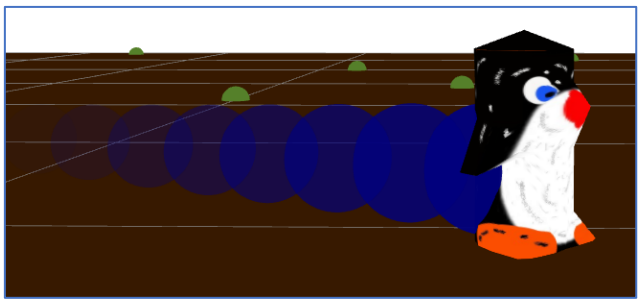

Figure 9: Pingouin avec ses phéromones

Les pingouins sécrètent des phéromones par le biais de la méthode « spawnPheromone », cette méthode est appelé au début de la méthode « actualiser » et fait apparaître une phéromone supplémentaire si le temps écoulé est supérieure à une durée fixée dans la classe et qui correspond à l'intervalle de génération de phéromones. Lorsque la méthode « updatePheromones » est appelée, la liste des phéromones sécrétés par un pingouin est parcourue et la méthode « actualiser » des phéromones est appelée. Cette méthode décrémente l'âge de la phéromone courant en fonction du pas de temps « dt ». Si jamais l'âge est inférieur à 0, ce dernier est détruit autrement la taille et l'opacité diminue. La Fig. 9 montre un pingouin avec ses phéromones, les plus récents sont ceux proches du pingouin et complètement opaque, les plus âgés sont les plus petits. **Cela répond à la Q6.**

Pierre LE DEZ
p5ledez@enib.fr

IV – Annexes

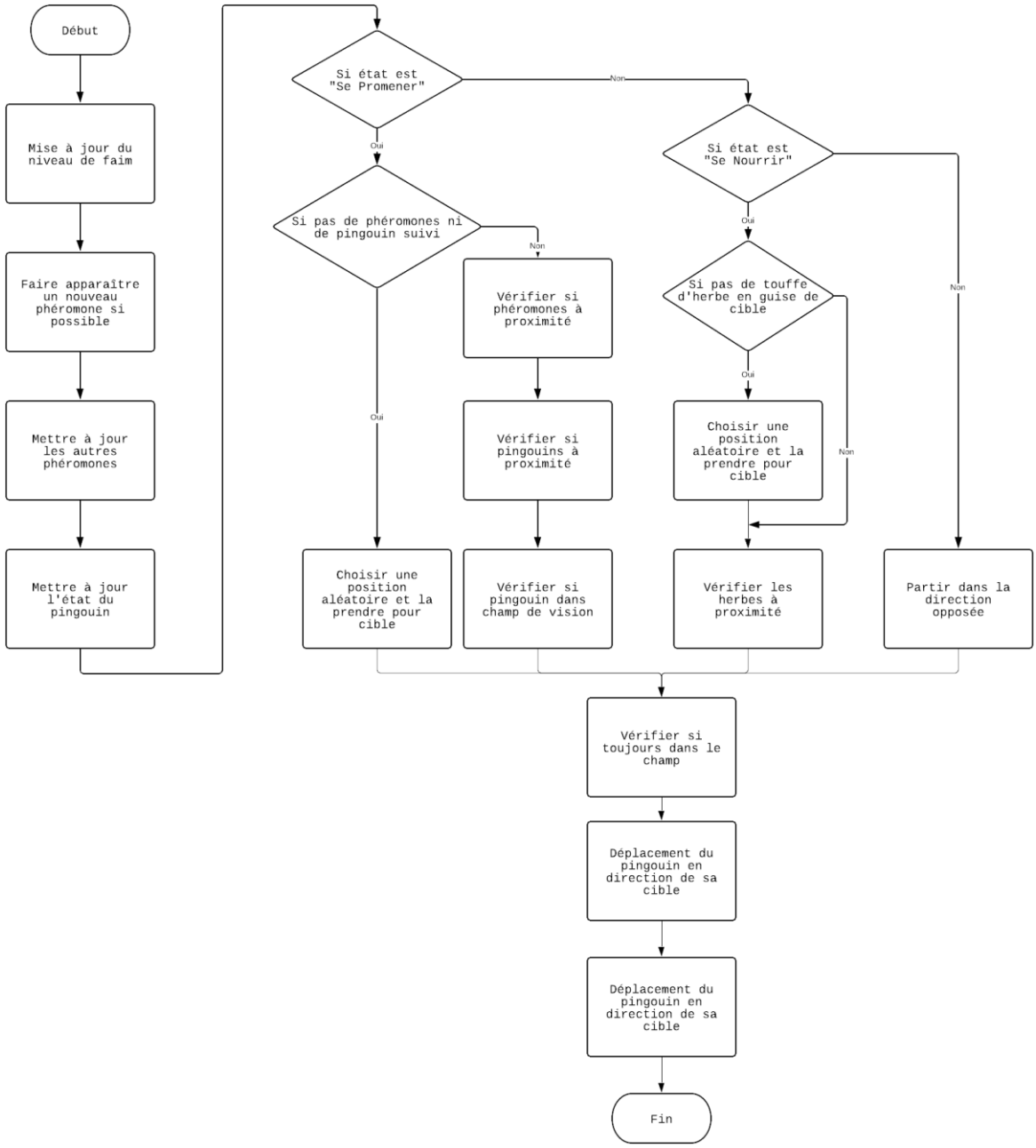

Figure 10: Algorithme de la méthode "actualiser"

Pierre LE DEZ
p5ledez@enib.fr