Université de Technologie Compiègne

IA04 : Système multi-agents

Printemps 2012

Projet

JWayOut

Yi CHEN – Maxime DAVID – Stéphan DONIN – Alain MARTIN

Table des matières

Introduction

JWayOut est un projet consistant en la simulation d'un exercice d'évacuation d'un lieu suite à un incendie. Un nombre donné d'individus est alors placé au sein de l'environnement de simulation et le superviseur humain de la simulation désigne un endroit où doit apparaître l'incendie, responsable du déclenchement de la simulation, à l'aide de la souris.

Suite à cela, la simulation peut alors débuter, simulation où tous les individus participant se dirigeront vers la sortie comme dans de réelles situations. Les influences entre les individus seront aussi prises en compte lors de la simulation.

L'objectif de ce projet est de simuler les comportements des individus face à un incendie en tenant compte de leurs propres caractéristiques et des contraintes imposées par l'environnement.

Conception

Pour que la simulation soit plus réelle, nous avons choisi la MDE (Maison Des Étudiants) de l'UTC comme lieu d'évacuation. La simulation est ainsi composée de trois grands éléments : l'environnement (murs, portes et sorties), les étudiants et le feu.

Environnement

L'environnement de l'évacuation est un espace dans lequel évoluent les comportements des étudiants et du feu ainsi que des lois qui les gouvernent. Il est donc composé de plusieurs pièces de tailles différentes communiquant les unes avec les autres, les sorties, les portes, les murs et les indicateurs sur ces derniers qui indiquent le sens des sorties.

Pour augmenter la variété des scénarios d'évacuation, le feu est placé non pas statiquement dans le fichier de configuration mais bien arbitrairement, et répondant au clic de l'utilisateur en début de simulation. Nous aurons ainsi la capacité de produire différentes situations d'incendie, ne sachant pas d'où peut venir le feu initialement.

Feu

Le feu est l'agent à l'origine des changements sur l'environnement. Il sera en mesure de progresser au sein de ce dernier. Afin de simplifier la complexité de propagation du feu, nous considérons que le feu s'étend vers des directions aléatoires et qu'il ne se propage que sur des cases adjacentes.

Le feu possède ainsi deux caractéristiques principales :

Direction de propagation

Indique la direction vers laquelle le feu va se propager. La valeur sera prise aléatoirement parmi les quatre possibilités : *NORTH*, *SOUTH*, *EAST*, *WEST*. Étant donné que tous les « morceaux de feu » peuvent prendre des valeurs différentes par itération, nous aurons donc pour chaque simulation et pour chaque itération des formes différentes du feu.

Vitesse de propagation

Le feu se propage toutes les n itérations. Nous avons attaché une attention particulière à ne pas utiliser de constantes dites magiques – c'est-à-dire des constantes au milieu d'une classe, hors de toute variable simplement modifiable par un novice. Aussi ce nombre d'itération est fixé dans la classe constante sous le nom de ***NUM_STEP_FIRE_SPREAD***.

Étudiants

Les étudiants sont des agents qui doivent réagir au changement de l'environnement, plus précisément dans notre cas, à l'apparition d'un incendie. Ils possèdent plusieurs caractéristiques internes qui font différencier les comportements des uns avec les autres.

Caractéristiques statiques

- **Capacité de vision :**

Les étudiants peuvent voir des choses (les murs, les portes, les autres étudiants, le feu, …) dans la distance de leur capacité de vision maximum à partir de leur endroit actuel.

- **Capacité auditive:**

Les étudiants peuvent entendre les bruits (cris, paroles des étudiants, etc.) venant d'endroits se trouvant dans leur champ auditif.

- **Capacité d'autonomie :**

Détermine si un étudiant est capable de se débrouiller tout seul face à un incendie.

- **Capacité de charisme :**

Détermine si un étudiant est plus enclin à suivre les décisions des autres ou celles de lui-même.

Les points de capacité de chaque étudiant sont attribués aléatoirement dans l'intervalle [`MIN_ABILITY, MAX_ABILITY`] pour ces quatre caractéristiques.

Caractéristiques dynamiques

- **Niveau de panique :**

Indique si un étudiant est encore capable de réfléchir à sa situation actuelle à un instant donné. Chaque étudiant possède son propre niveau maximum de panique.

Le niveau de panique s'amplifie de différents degrés dans les situations suivantes :

- *L'étudiant a vu le feu lui-même* :

  Son niveau de panique augmente très vite (incrémenté de *STRONG_PANIC*)

- *L'étudiant a entendu les cris ou les annonces du feu :*

  Son niveau de panique augmente un peu (incrémenté de *1*)

- *L'étudiant qui possède un bas niveau d'autonomie se trouve tout seul ou n'entend aucune décision des autres :*

  Son niveau de panique augmente très vite (incrémenté de *STRONG_PANIC*)

- **Vitesse de déplacement :**

Indique la capacité de déplacement par itération d'un étudiant à un instant donné.

Elle varie selon le niveau de panique que les étudiants possèdent. Nous aurons donc quatre niveaux de vitesse :

AGENT_SLOW_SPEED, AGENT_NORMAL_SPEED,

AGENT_HIGH_SPEED, AGENT_VERY_HIGH_SPEED

Comportements des étudiants

- **Se déplacer :**

Les étudiants peuvent se déplacer de X cases par itération selon leur vitesse de déplacement. S'ils rencontrent les murs ou les portes fermées, ils ne peuvent pas les traverser. Les déplacements se font toujours dans le but de s'éloigner du feu.

- **Se tourner :**

Les étudiants ont la capacité de tourner vers les 4 directions (*NORTH, SOUTH, EAST, WEST*) pour acquérir les différentes vues ou s'orienter vers la direction dans laquelle ils ont décidé d'aller.

- **Percevoir :**

Au début de chaque tour, un étudiant peut percevoir et mémoriser le contenu des X cases devant lui (les murs, les portes, et les autres étudiants sur la même direction), X étant sa distance de capacité de vision.

Il peut aussi bien sûr entendre et mémoriser le contenu des bruits venant des Y cases autour de lui, Y étant sa distance de capacité d'ouïe.

- **Parler :**

Un étudiant est capable de crier en raison de sa monté en panique. Ceci donne l'alerte des anomalies ou des dangers aux autres étudiants. Dans la situation d'un incendie, un étudiant peut aussi déclarer aux autres sa propre décision d'évacuation pour que ces derniers le suivent.

- **Réfléchir :**

Un étudiant possède la capacité de réflexion pour décider vers quelle direction se déplacer. Si nous considérons le fait que l'étudiant peut regarder une flèche indiquant une sortie sur les murs, regarder autour de lui pour si d'autres personnes sont présentes et écouter s'il les entend à proximité afin de prendre une décision, alors les différents paliers de panique lui permettront de se servir d'une, de deux ou de ces trois possibilités lors de la prise de décision pour une itération donnée.

La possibilité qu'il ne regarde rien pour prendre sa décision et se dirige alors aléatoirement vers une direction autre que celle du feu (s'il voit ce dernier) est également envisageable pour un niveau de panique maximum.

Algorithme d'évacuation

Chaque itération

Quand il n'y a pas de feu, tout le monde est calme.

Quand le feu apparaît :

Étudiant ayant vu le feu :
Crier fort, niveau de panique augmente très rapidement
Il prend la décision
Il se déplace en annonçant le feu aux autres

Étudiant qui n'a pas vu le feu mais qui a entendu les cris :
Niveau de panique augmente un peu
Il prend la décision
Il se déplace

Étudiant qui n'a pas vu le feu et qui n'a pas entendu non plus les cris :

S'il voit des gens bouger
Il s'approche d'eux pour demander ce qui s'est passé
Il prend la décision
Il se déplace

S'il ne voit rien et n'entend rien
Reste calme

Décision

Étudiant qui a encore la capacité de réflexion (niveau de panique moins élevé) :

S'il est proche du feu
Il évacue

Étudiant qui a un niveau de panique très proche de son niveau de panique maximum

S'il est tout seul
Il ne sait pas quoi à faire
Il reste sans bouger

S'il est avec d'autres gents
Il suit les autres

Évacuation

Si l'étudiant a un haut niveau d'autonomie et un haut niveau de charisme

Il prend la décision lui-même (décision autonome) et dit aux gens autour de lui pour qu'ils le suivent

S'il a un bas niveau de charisme

S'il entend la décision d'un autre étudiant ou s'il y a des gens à côté
Il suit la décision ou les gens

Sinon
S'il est assez autonome
Il prend une décision par lui même
Sinon
Son niveau de panique augmente très vite
Il se déplace aléatoirement

Décision autonome

Si l'étudiant voit une porte qui mène vers la sortie
    Il franchit cette porte

Sinon
    S'il voit d'autres étudiants qui se déplacent
        Il les suit

    Sinon
        S'il entend les bruits d'autres étudiants
            Il se dirige vers ces personnes

        Sinon
            Il se déplace aléatoirement

Suit les autres

S'il y a un étudiant très charismatique
    Il suit sa décision

S'il y a plusieurs étudiants très charismatiques
    Il suit celui le plus proche

S'il n'y a pas d'étudiant charismatique
    Il suit l'étudiant le plus proche

Se déplace

Si l'étudiant est très paniqué
    Il se déplace à une vitesse très haute

Sinon
    S'il a vu le feu
        Il se déplace à vitesse haute

    S'il n'a pas vu le feu mais il a entendu des cris
        Il se déplace à une vitesse normale

    S'il n'a pas vu le feu, ni entendu de cris, mais qu'il sait juste qu'il y a feu
        Il se déplace à une vitesse moyenne

Aspects techniques

Plate-forme utilisée

Cette simulation sera réalisée principalement au travers de la plate-forme MASON. Il s'agit d'une librairie de simulation disponible via le langage Java. La plate-forme MASON comprend deux grandes parties bien séparées : une librairie de modèle et un ensemble d'outils en option pour la visualisation en 2D ou en 3D.

Un grand avantage de MASON est qu'il nous fournit par défaut une console permettant de démarrer, arrêter, manipuler étape par étape la simulation ainsi que la possibilité de changer la vitesse d'avancement de cette dernière et d'inspecter les objets sur la vue, etc. Elle nous permet de contrôler plus facilement la simulation.

Concernant notre projet, nous sommes partis d'une visualisation en 2D. Donc, au niveau de la vue, il s'agit d'abord de mettre en place un plan « vu de dessus » de la Maison Des Étudiants, puis d'ajouter les individus sur la carte. Ensuite, nous avons implémenté l'algorithme d'évacuation des étudiants et l'algorithme de propagation du feu.

A la fin, nous ajoutons les inspecteurs personnalisés pour le modèle entier mais aussi pour chaque élément de la grille afin de voir leur état à un instant donné.

Mise en place du plan 2D

Le plan de la maison des étudiants est construit à partir d'un fichier XML. Ce fichier est composé de 4 balises d'objet principales : *<wall>*, *<door>*, *<exit>*, *<people>* qui représentent respectivement les murs, les portes (avec leur sens de franchissement), les sorties et les étudiants.

Chaque balise d'objet contient deux attributs basiques : le point de départ (*begin*) et le point de fin (*end*) qui ensemble définissent la localisation de cet objet sur le plan. De plus, les objets *<door>* possèdent aussi un attribut de direction (*NORTH, SOUTH, EAST, WEST*) qui indique vers quelle direction l'objet doit être franchit.

Les données de ce plan sont ajoutées dans le composant *ObjectGrid2D* du modèle de notre simulation, et la visualisation est réalisée avec l'aide du composant *Display2D* de la vue.

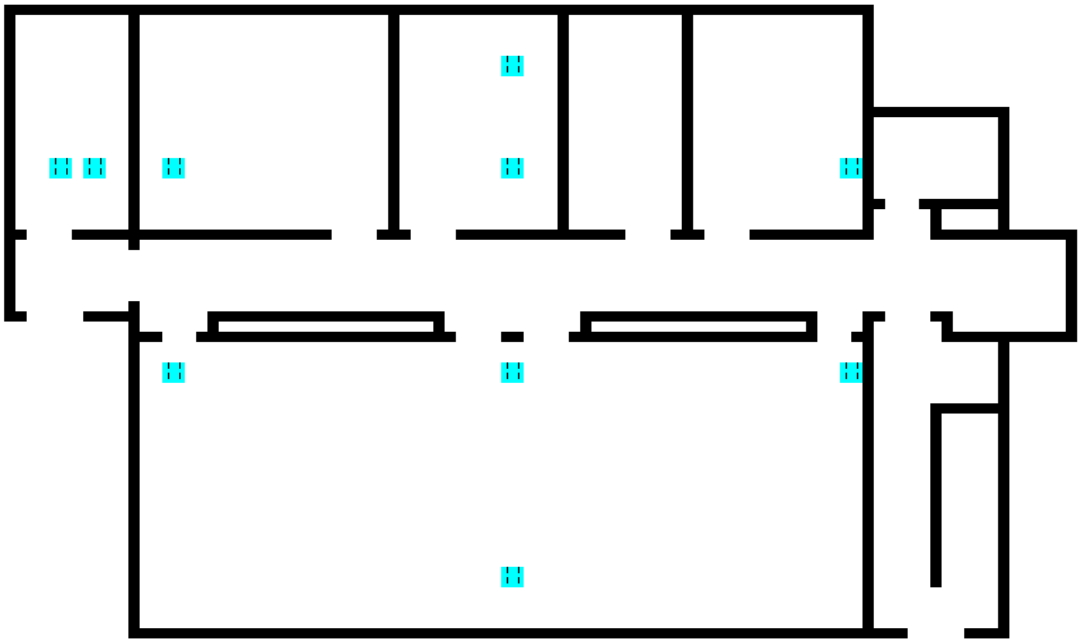

Figure 1 : Le plan 2D de la maison des étudiants

Inspectors

Nous avons implémenté deux types d'inspecteurs :

- **Un inspecteur du modèle**

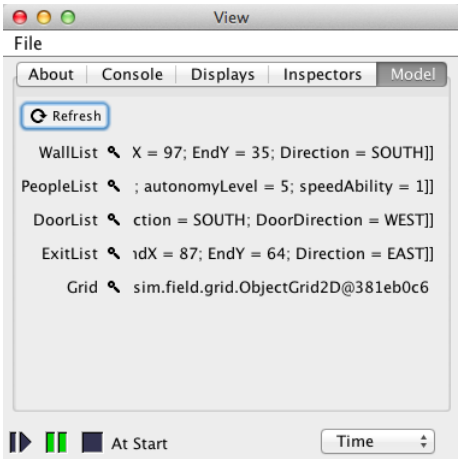

Figure 2 : L'inspecteur du modèle

- **Un inspecteur pour chaque objet de la grille.**

Lors du clic, nous remontons TOUTES ses caractéristiques afin de toujours savoir le comportement de chaque agent.

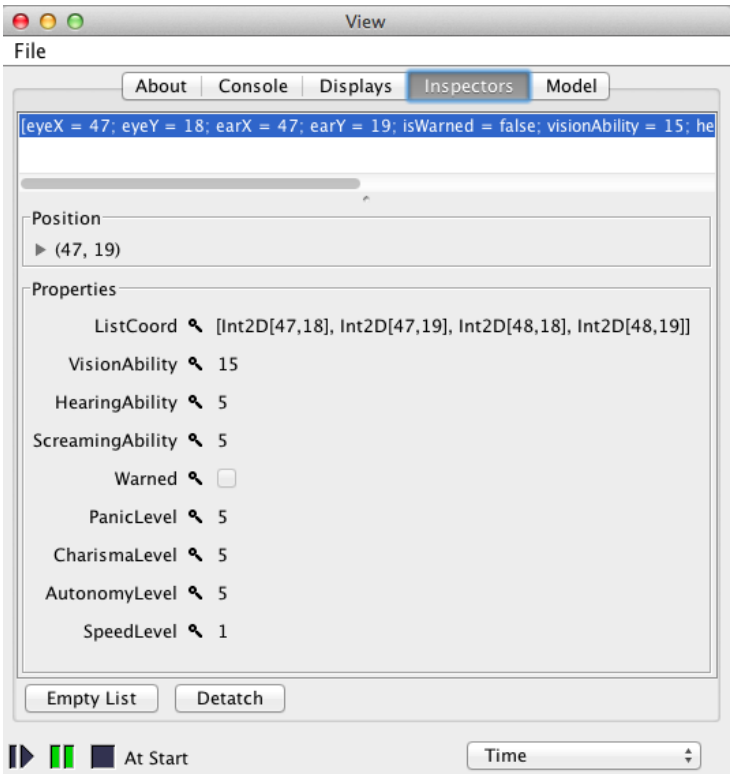

Figure 3 : L'inspecteur d'un agent « People »

Conditions initiales

```
// VIEW

public final static int GRID_WIDTH = 101;
public final static int GRID_HEIGHT = 68;
public static enum Direction { UNKNOWN, NORTH, SOUTH, EAST, WEST };

// AGENT

public final static int MAX_ABILITY = 18;
public final static int MIN_ABILITY = 12;
public final static int STRONG_PANIC = 3;
public final static int MAX_PANIC = 50;

public final static int AGENT_SLOW_SPEED = 1;
public final static int AGENT_NORMAL_SPEED = 2;
public final static int AGENT_HIGH_SPEED = 3;
public final static int AGENT_VERY_HIGH_SPEED = 4;

public static final int NUM_STEP_FIRE_SPREAD = 2;
```

Conclusion

Ce projet est à l'objectif double : utiliser une plate-forme de simulation – ici MASON – et concevoir un système multi-agents et cette dualité a été bénéfique pour nous par bien des aspects.

Dans un premier temps, il nous a permis de réutiliser les compétences acquises lors du TD sur la simulation, mais aussi de les approfondir grâce à une documentation très complète : http://cs.gmu.edu/~eclab/projects/mason/manual.pdf

Le fait d'utiliser une dizaine d'agents, bien que contraignant au départ, nous a permis de développer des comportements intelligents. En effet, les agents sont à l'écoute les uns des autres, ils interagissent : ils peuvent se suivre mais également s'ignorer (dans le cas d'une panique trop élevée).

Ce projet étant une simulation, bon nombre de fonctionnalités peuvent être ajoutées. Des critères supplémentaires pourraient en effet être intégrés :

- L'épaisseur des murs (freine ou accélère la propagation du feu)
- L'occupation des agents (écoute la musique -> temps d'alerte accru)
- La mise en place des fenêtres (le vent pourrait également agir sur la propagation du feu)
- Des obstacles pourraient également être disposés le long du parcours, rendant l'évacuation plus difficile
- Des portes de type coupe-feu pourraient stopper l'avancement du feu.

Cette liste n'est évidemment pas exhaustive et les ajouts sont quasiment illimités ce qui rend ce projet réellement intéressant et modulable.