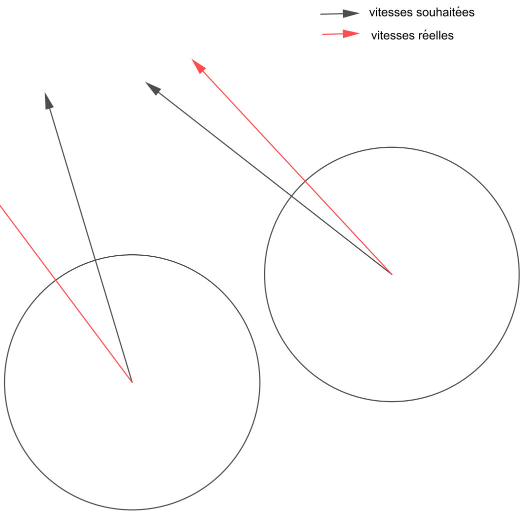
Un modèle de mouvements de foule

Description du modèle



Sommaire

Principe de l'algorithme

L'algorithme de FMM, permet de modéliser la propagation d'une onde dans un milieu. Ainsi, il peut être utilisé pour construire une carte des distances à la sortie, où la distance d'un point à cette dernière correspond au temps que met une onde se propageant depuis la sortie pour atteindre le point.
Cet algorithme utilise le même principe que l'algorithme de Dijkstra : on calcule de proche en proche les distances de chaque point à l'origine en acceptant comme calculé à chaque itération le point dont la distance est la plus faible, puis en déterminant la distance à l'origine de ses voisins.

Structure du programme

```
for all points en sortie do
    déterminer leur distance à la sortie
    if elle est inférieure à la valeur précédente then
        la définir comme la nouvelle distance
        actualiser le tas
    end if
    marquer le point comme considéré
end for
while le tas est non vide do
    extraire la tête du tas et la marquer comme calculée
    actualiser le tas
    for all points dans le voisinage de la tête do
        déterminer leur distance à la sortie
        if elle est inférieure à la valeur précédente then
            la définir comme la nouvelle distance
            actualiser le tas
        end if
        marquer le point comme considéré
    end for
end while
```

Calcul des distances

On considère que l'onde se propage de manière uniforme donc la norme de son gradient est constante que l'on fixe à 1.
On note $x_{i,j}$ le point de coordonnées (i, j) et $x_{i-1,j}, x_{i+1,j}, x_{i,j-1}, x_{i,j+1}$ ses voisins. On note pour tout point x $D(x)$ sa distance.
Le calcul de la distance de $x_{i,j}$ se fait à partir des distances de ces voisins qui ont été considérés comme calculés, et repose sur l'approximation :
$\left\|\vec{grad}(D)(x_{i,j})\right\|^2 \approx (max(D(x_{i,j}) - D(x_{i-1,j}), D(x_{i,j}) - D(x_{i+1,j}))^2 + (max(D(x_{i,j}) - D(x_{i,j-1}), D(x_{i,j}) - D(x_{i,j+1}))^2$

Calcul des distances

Comme on prend $\left\|\vec{grad}(D)(x_{i,j})\right\|^2 = 1$, on obtient en notant $d_h = min(D(x_{i-1,j}), D(x_{i+1,j}))$ et $d_h = min(D(x_{i,j-1}), D(x_{i,j+1}))$:
$1 = (D(x_{i,j}) - d_h)^2 + (D(x_{i,j}) - d_v)^2$
On déduit $D(x_{i,j}) = \frac{d_v - d_h + \sqrt{2-(d_v-d_h)^2}}{2}$

Utilisation d'un tas

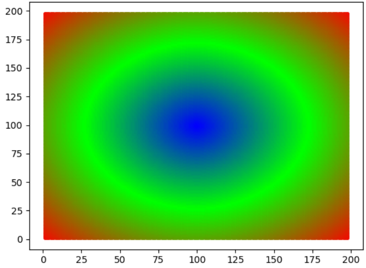
Afin d'accéder rapidement au point non accepté de distance la plus faible, on stocke les points non acceptés dont la distance a été calculée dans un tas, un arbre binaire dans lequel la distance d'un nœud est plus faible que celle de ses fils. Il suffit alors d'extraire la tête de l'arbre pour obtenir le point recherché.
Le tas est actualisé au cours de l'exécution de l'algorithme à l'aide des procédures suivantes :

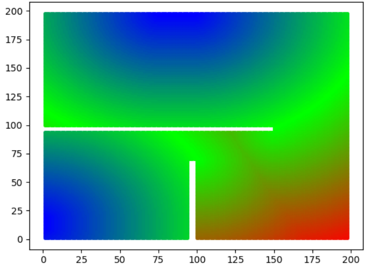
- Remonter, qui permet de rétablir le tas après que la valeur d'une distance ait été modifiée
- Insérer, qui permet d'ajouter un nouveau point au tas
- Extraire qui permet d'extraire la tête tout en conservant la structure de tas

Preuve et complexité

- La preuve repose sur l'invariant de boucle suivant : "les distances des points marqués comme calculés sont minimales"
- Si l'on considère les appels à la fonction de calcul de distance, alors l'algorithme a une complexité linéaire par rapport à l'aire de la grille
- Si l'on prend également en compte la comparaison de distances, utilisée par les procédures de gestion du tas qui ont une complexité linéaire par rapport à la taille de ce dernier, la complexité de l'algorithme passe en O(n log n) avec n le nombre de cases de la grille

Résultats





Introduction

Avec la carte des distances obtenues précédemment, on peut déterminer la vitesse souhaitée d'un individu en en prenant le gradient, et ainsi construire la vitesse souhaitée du groupe d'individu : $V_s = (v_1, ..., v_n) \in \mathbb{R}^{2N}$

Pour éviter les collisions, la vitesse réelle V doit appartenir à l'ensemble

$Adm = \{V \in \mathbb{R}^{2N} | \forall (i,j) \in [\![1, N]\!]^2, h \langle (v_i - v_j) \mid e_{i,j} \rangle < D_{i,j} - R\}$

Avec $D_{i,j}$ la distance entre deux individus et R leur rayon.

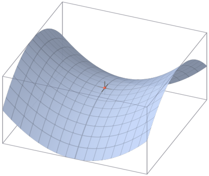
On va donc projeter la vitesse souhaitée sur cet ensemble.

Vitesse réelle

Principe de l'algorithme

L'ensemble des vitesses admissibles est convexe, donc on est assuré de l'existence et de l'unicité du projeté, que l'on cherche désormais à calculer.
Cela revient à trouver l'élément de Adm minimisant la distance à V_s.
On ne peut pas appliquer la méthode du gradient, car elle nécessite de projeter à chaque étape sur l'ensemble Adm, ce que l'on cherche à réaliser.
On va donc se ramener à un problème où l'on pourra utiliser la méthode du gradient.

Points selles



Soit f une application d'un produit cartésien $U \times M$ vers $\mathbb{R}$, on dit que (u, λ) est point selle de f si $f(v,\lambda) = \inf_{u \in U} f(u,\lambda) = \sup_{\mu \in M} f(v,\mu)$

On va montrer que le projeté de Vs sur Adm est le premier argument d'un point selle du lagrangien défini par :

$$L : \mathbb{R}^{2N} \times \mathbb{R}_+^{\frac{N(N-1)}{2}}, (v,\lambda) \mapsto \frac{1}{2}\|v - V_s\|^2 + \sum_{1 \leqslant i < j \leqslant N} \lambda_{i,j}(h\langle (v_i - v_j) \mid e_{i,j}\rangle - D_{i,j} + R)$$

On pourra alors chercher à trouver le second argument du point selle, en utilisant la méthode du gradient, car la projection sur $\mathbb{R}_+^{\frac{N(N-1)}{2}}$ est facile à réaliser.

Si (v, λ) est point selle de L, v est le projeté de V_s sur Adm

Comme $L(v,\lambda) = \inf\left\{L(v,\mu), \mu \in \mathbb{R}_+^{\frac{N(N-1)}{2}}\right\}$, pour tout μ dans $\mathbb{R}_+^{\frac{N(N-1)}{2}}$, $\sum_{1\leqslant i<j\leqslant N} (\lambda_{i,j} - \mu_{i,j})(h\langle (v_i - v_j) \mid e_{i,j}\rangle - D_{i,j} + R) \geqslant 0$

Donc en fixant tous les $\mu_{i,j} = \lambda_{i,j}$ sauf un que l'on fait tendre jusqu'à l'infini, on obtient $\forall 1 \leqslant i < j \leqslant N, h\langle (v_i - v_j) \mid e_{i,j}\rangle - D_{i,j} + R \leqslant 0$ donc $v \in Adm$

Et de même avec 0 à la place de l'infini,

$$\sum_{1\leqslant i<j\leqslant N} \lambda_{i,j}(h\langle (v_i - v_j) \mid e_{i,j}\rangle - D_{i,j} + R) = 0$$

Enfin,

$$\begin{aligned}
\forall u \in Adm, \tfrac{1}{2}\|v - V_s\|^2 &= L(v,\lambda) \\
&\leqslant L(u,\lambda) \\
&\leqslant \tfrac{1}{2}\|u - V_s\|^2 + \sum_{1\leqslant i<j\leqslant N} \lambda_{i,j}(h\langle (u_i - u_j) \mid e_{i,j}\rangle - D_{i,j} + R) \\
&\leqslant \tfrac{1}{2}\|u - V_s\|^2 \text{ donc } v \text{ est bien le projeté sur } Adm
\end{aligned}$$

Si v est le projeté sur Adm, un point selle existe

On utilisera les relations de Kuhn et Tucker, qui affirment que si J et $(\varphi_i)_{1\geqslant i\geqslant m}$ sont des fonctions convexes différentiables sur un espace vectoriel E dans $\mathbb{R}$, si on note $U = \{v \in E | \forall i \in [\![1, m]\!], \varphi_i(v) \leqslant 0\}$

Si u est un minimum de J par rapport à U et si les $(\varphi_i)_{1\geqslant i\geqslant m}$ sont affines, alors il existe λ dans $\mathbb{R}_+^m$ tel que :

$$\begin{cases} dJ(v) + \sum_{i=1}^{m} \lambda_i d\varphi_i(v) = 0 & (1) \\ \sum_{i=1}^{m} \lambda_i \varphi_i(v) = 0 & (2) \end{cases}$$

De (2) on tire

$$\begin{aligned} \forall \mu \in \mathbb{R}^{\frac{N(N-1)}{2}}, L(v,\mu) &= \frac{1}{2}\|v - V_s\|^2 + \sum_{1\leqslant i<j\leqslant N} \mu_{i,j}(h\left\langle (v_i - v_j) \mid e_{i,j}\right\rangle - D_{i,j} + R \\ &\leqslant \frac{1}{2}\|v - V_s\|^2 = L(v,\lambda) \end{aligned}$$

Et (1) est une condition suffisante de minimum d'une fonction convexe, donc (λ, μ) est point selle.

La méthode de Uzawa

On cherche à trouver le second argument du point selle, c'est-à-dire trouver $\lambda \in \mathbb{R}_+^{\frac{N(N-1)}{2}}$ tel que $G(\lambda) = \sup \left\{ G(\mu), \mu \in \mathbb{R}_+^{\frac{N(N-1)}{2}} \right\}$, avec $G(\mu) = \inf_{v \in \mathbb{R}^{2N}} L(v, \mu)$. Pour cela, on va utiliser la méthode du gradient et construire par récurrence deux suites $(u_k)_{k \in \mathbb{N}}$ et $(\lambda_k)_{k \in \mathbb{N}}$ en prenant $\lambda_0 \in \mathbb{R}_+^{\frac{N(N-1)}{2}}$ de manière arbitraire, puis :

$$\begin{cases} u_k \text{ tel que } L(u_k, \lambda_k) = \inf_{v \in \mathbb{R}^{2N}} L(v, \lambda_k) \\ (\lambda_{k+1})_i = max((\lambda_k)_i + \rho(h \langle ((u_k)_i - (u_k)_j) \mid e_{i,j} \rangle - D_{i,j} + R), 0) \end{cases}$$

Avec ρ une constante positive que l'on déterminera.

Preuve de la convergence

L'application $h : \mathbb{R}^{2N} \to \mathbb{R}^{\frac{N(N-1)}{2}}, v \mapsto (h\langle (v_i - v_j) \mid e_{i,j} \rangle)_{1 \leqslant i < j \leqslant N}$ est linéaire, notons C sa matrice dans les bases canoniques. On peut alors écrire $L(v, \mu) = \frac{1}{2}\|v - V_s\|^2 + \langle \mu | Cv \rangle - \langle \mu | D \rangle$. Avec $D = (D_{i,j} - R)_{1 \leqslant i < j \leqslant N}$

Et donc avec (v, λ) un point selle de L, on obtient en différenciant selon v : $v - V_s + {}^tC\lambda = 0$

De plus, $\langle \mu - \lambda | Cv - D \rangle \leqslant 0$ $\langle \mu - \lambda | \lambda - (\lambda + \rho(Cv - D)) \rangle \geqslant 0$

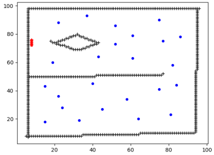
D'où $\begin{cases} v - V_s + {}^tC\lambda = 0 \\ \lambda = P_+(\lambda + \rho(Cv - D)) \end{cases}$. Et par construction, $(u_k, \lambda_k, \lambda_{k+1})$ vérifient les mêmes relations donc :

$$\begin{cases} u_k - v + {}^tC(\lambda_k - \lambda) = 0 \\ \|\lambda_{k+1} - \lambda\| \leqslant \|\lambda_k - \lambda + \rho C(u_k - v)\| \end{cases}$$

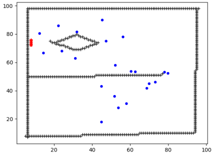
En élevant au carré l'inégalité :
$\|\lambda_{k+1}-\lambda\|^2 \leqslant \|\lambda_k-\lambda\|^2 + 2\rho\left\langle {}^tC(\lambda_k-\lambda)|u_k-v\right\rangle + \rho^2\|C(u_k-v)\|^2$
Ainsi, $\|\lambda_{k+1}-\lambda\|^2 \leqslant \|\lambda_k-\lambda\|^2 - \rho(2-\rho|||C|||^2)\|u_k-v\|^2$
Donc si $2-\rho|||C|||^2>0$, la suite $(\|\lambda_k-\lambda\|^2)_{k\in\mathbb{N}}$ est décroissante donc converge.
Ainsi, comme $\|u_k-v\|^2 \leqslant \frac{\|\lambda_k-\lambda\|^2-\|\lambda_{k+1}-\lambda\|^2}{\rho(2-\rho|||C|||^2)}$, la suite $(u_k)_{k\in\mathbb{N}}$ converge vers v.

Implémentation

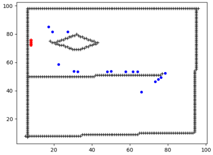
- Utilisation de `NumPy` pour représenter les vecteurs et matrices
- Distances et vitesses souhaitées calculées une seule fois
- $\lambda_0 = 0$
- $|||C|||^2 = \sum_{1 \leqslant i < j \leqslant N} |h \langle (v_i - v_j) \mid e_{i,j} \rangle|^2 \leqslant \frac{N(N-1)}{2} 4h^2 \|v\|^2 < 2N^2h^2$
  Donc $\rho = \frac{1}{N^2h^2}$
- Interface graphique avec `Tkinter`



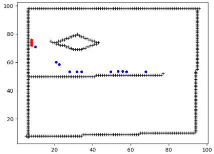
(a) t=1



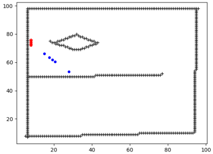
(b) t=150



(c) t=300

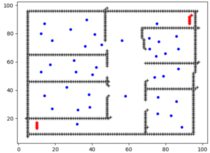


(d) t=450

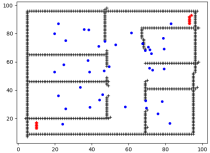


(e) t=650

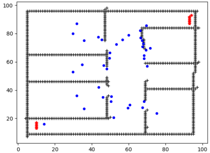
FIGURE – Première simulation



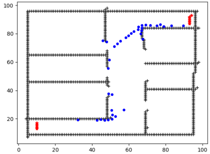
(a) t=1



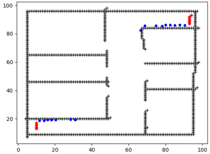
(b) t=50



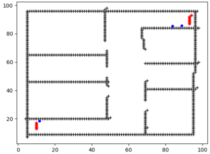
(c) t=100



(d) t=200



(e) t=400



(f) t=500

FIGURE – Seconde simulation