Gabriel Lefloch
Numéro de dossier: **17543**

Gestion automatisée de voitures autonomes au sein d’une intersection

Comment peut-on mettre en place un système multi-agent entièrement automatisé dans le cadre d'une intersection?

Sommaire

1. Modèle et méthodes utilisés
2. Simulation avec python
3. Résulats
4. Conclusion

Modèles et méthodes utilisés

Intelligent Driver Model (IDM)

- L'IDM est un **modèle microscopique**

- Chaque voiture est un **agent** dont la vitesse dépend des :

    - voitures de devant

    - des propriétés intrinsèques à cette voiture (ie: accélération maximale, décélération minimale, longueur, etc)

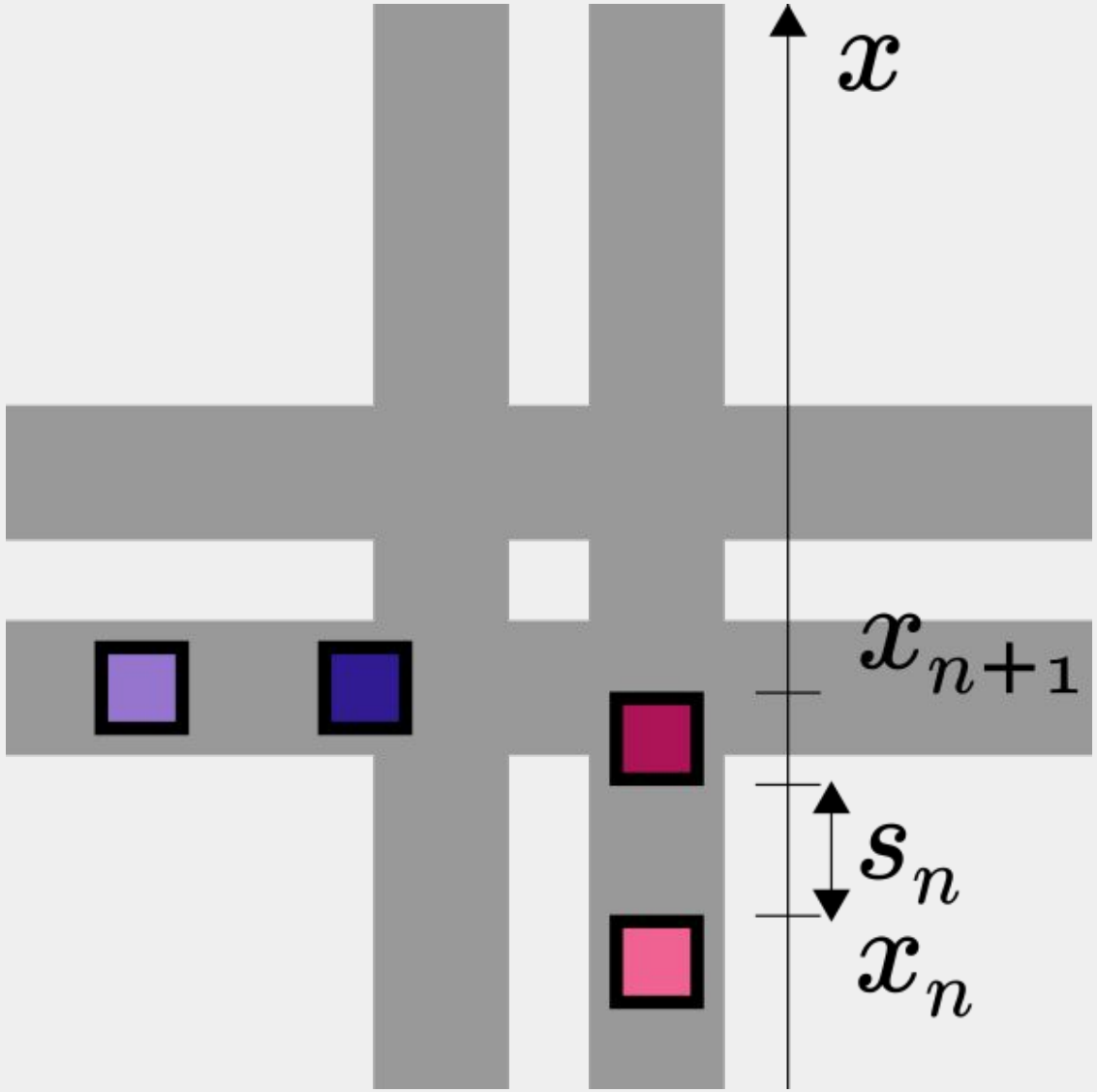
Forme générale

Soit $n \in \mathbb{N}^*$. On pose $\mathbf{V_n} = \{V_1, V_2, \ldots, V_n\}$ un ensemble de n voitures autonomes.

Pour $k \in [\![1; n]\!]$, $x_k(t)$ désigne la position de la voiture V_k.

$$\ddot{x}_k(t) = a_k \left(1 - \left(\frac{\dot{x}_k}{v_d} \right)^{\delta} - \left(\frac{s^*(\dot{x}_k, \dot{x}_{k+1})}{s_k} \right)^2 \right)$$

- $x_n(t)$: la position de la voiture n
- $x_{n+1}(t)$: la position de la voiture devant la voiture n
- $s_n = x_{n+1} - x_n - l_{n+1}$
- $s^*(\dot{x}, \dot{x}_{n+1}) = s_n^0 + T_n \dot{x}_n(t) + \dot{x}_n(t) \frac{\dot{x}_{n+1}(t) - \dot{x}_n(t)}{\sqrt{a_n b_n}}$



Description du modèle IDM

Paramètres du modèle IDM

- s_n^0 : la distance minimale entre deux voitures
- T_n : le temps minimale entre deux voitures
- a_n : l'accelération maximale de la voiture n
- b_n : décélération maximale de la voiture n

s^* : fonction homogène à une distance qui représente l'écart désirée entre deux voitures

Comportement asymptotique

$$\ddot{x}_k(t) = \ddot{x}_k^{\text{libre}}(t) + \ddot{x}_k^{\text{int}}(t)$$

- En circulation fluide, l'interdistance s_n devient grande, ainsi :

$$\ddot{x}_k(t) \approx a_n \left(1 - \left(\frac{x'_n}{v_d}\right)^{\delta}\right) = \ddot{x}_k^{\text{libre}}(t)$$

- En circulation dense, la vitesse diminue et donc on a :

$$\ddot{x}_k(t) \approx -a_n \left(\frac{s^*(\dot{x}_k, \dot{x}_{k+1})}{s_k}\right)^2 = \ddot{x}_k^{\text{int}}(t)$$

Méthodes de gestion des voitures

Soit V_k la $k^{\text{ième}}$ voiture d'une file dénoté par l'ensemble $\mathbf{V_n} = \{V_1, \ldots, V_n\}$. En cas de rejet de requète pour rentrer dans l'intersection, plusieurs options sont possibles.

Un principe de file où la première voiture arrivée sera la première sortie

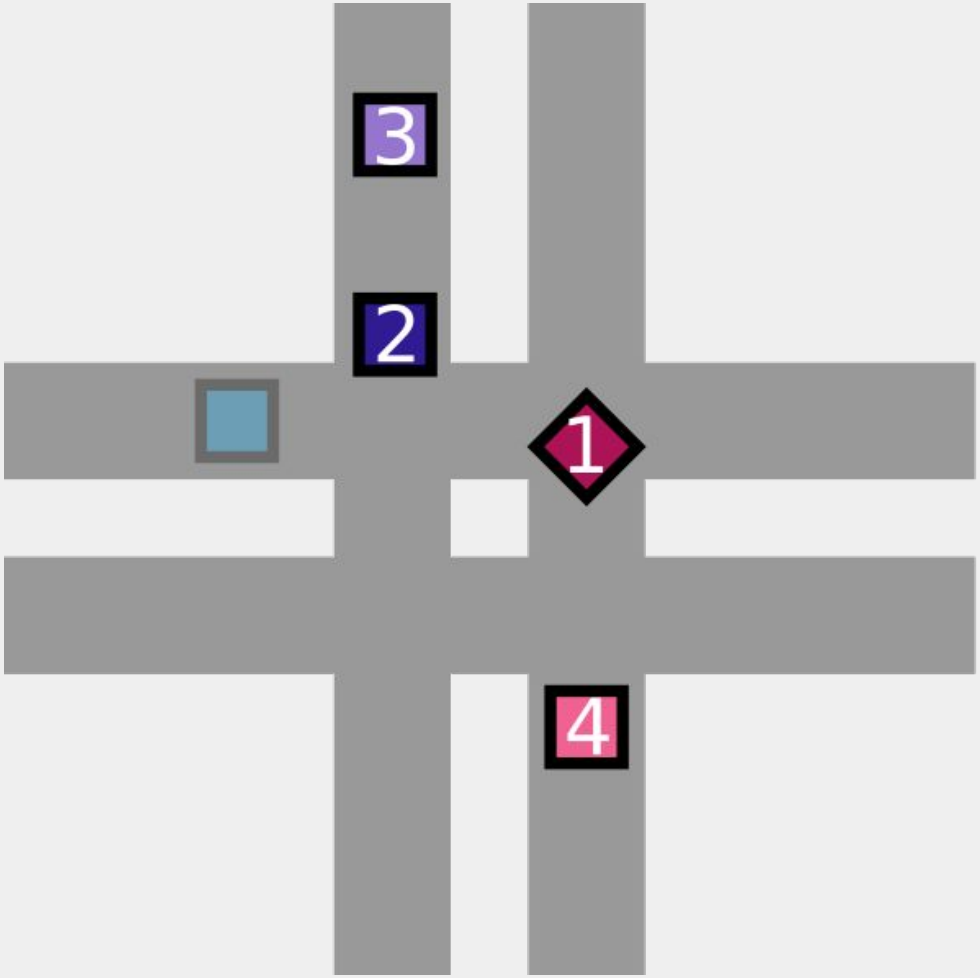
Ou bien employer une méthode similaire à celle des feux tricolores, nommée la **méthode BATCH**

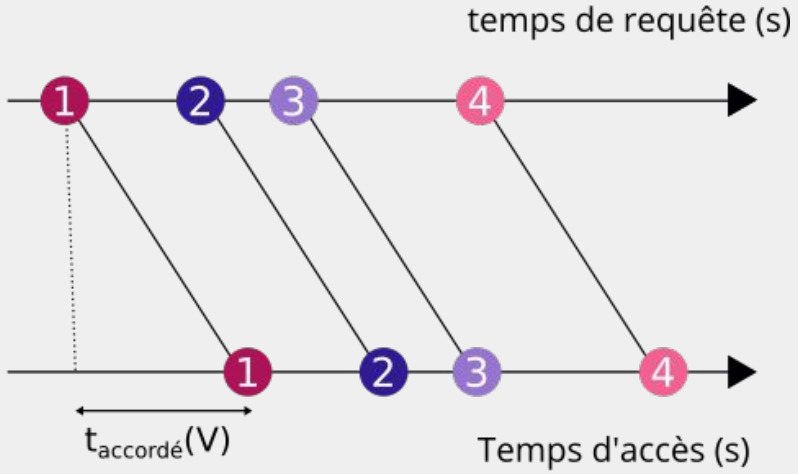
Principe de file

Soit $V \in \mathbf{V_n} = \{V_1, \ldots, V_n\}$.

On note $t_{arrivée}(V)$ le plus petit temps que prendra la voiture V pour arriver à l'intersection.

- Un temps $t_{accordé}(V)$ est calculé par le gestionnaire de l'intersection puis envoyé à la voiture V **en fonction de sa requête**
- Nécessairement, $t_{arrivée}(V) \leq t_{accordé}(V)$
- La voiture entrera donc dans l'intersection à $t_{accordé}(V)$





Principe de file (communément noté PEPS)

La méthode BATCH

Soit $V \in \mathbf{V_n} = \{V_1, \ldots, V_n\}$ une voiture et $N \in \mathbb{N}$. On suppose que V doit attendre un certain temps avant d'arriver à l'intersection.

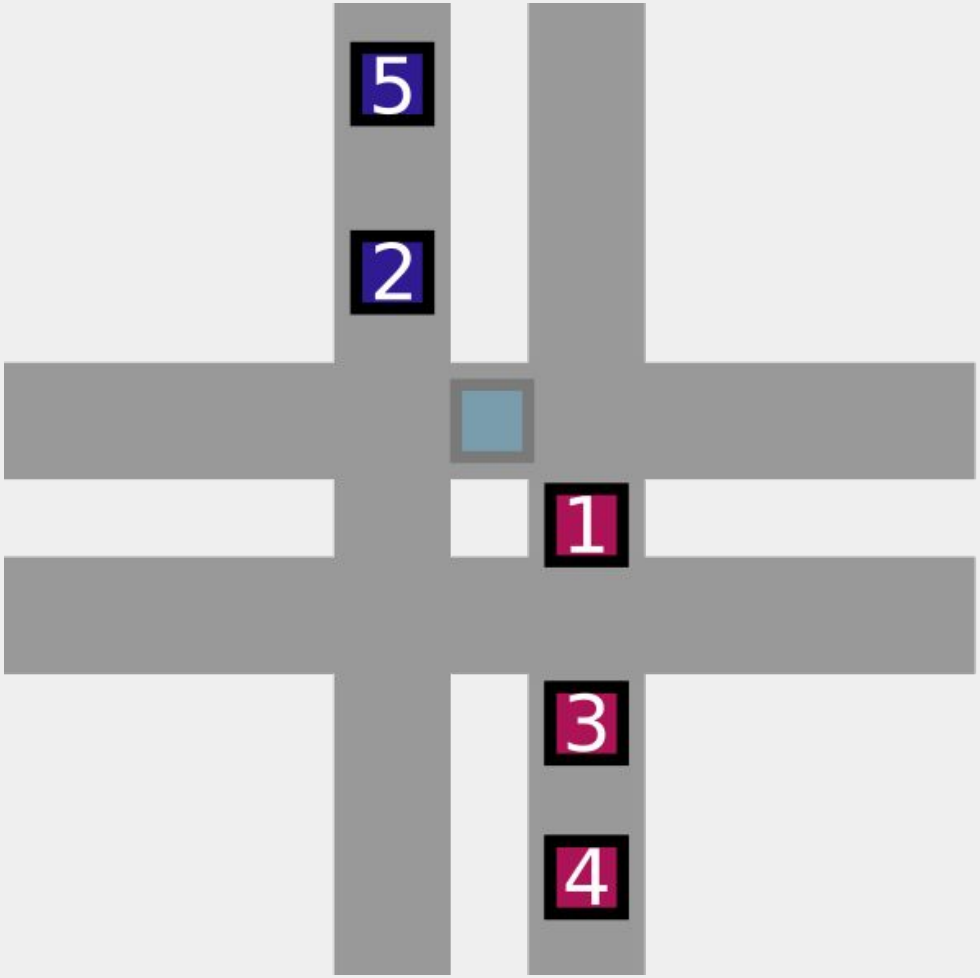
On note :

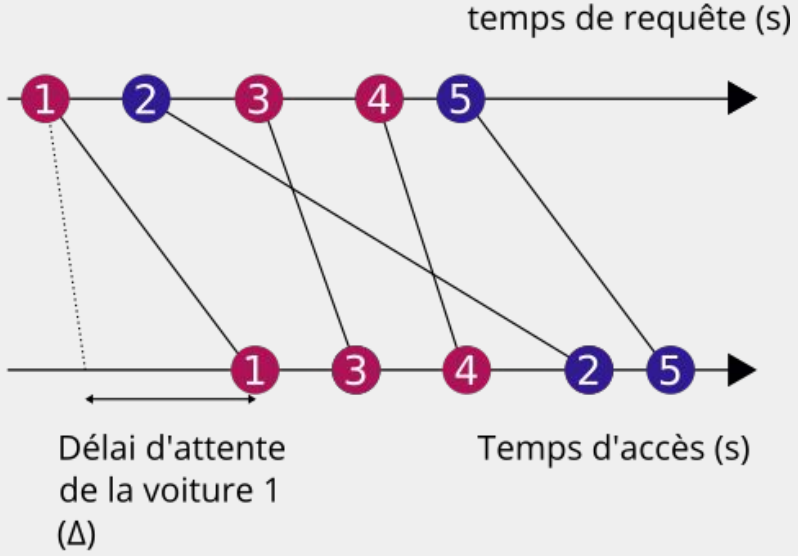
$$\begin{cases} t_{\text{arrivée}}(V) \text{ le plus petit temps que prendra} \\ \text{la voiture } V \text{ pour arriver à l'intersection} \\ t_{\text{accordé}}(V) \text{ le temps de passage de} \\ \text{la voiture } V \text{ calculé par l'intersection} \end{cases}$$

1. Le temps d'attente est: $\Delta = t_{arrivée}(V) - t_{accordé}(V)$
2. Pendant Δ, on sépare l'ensemble $\mathbf{V_n}$ en 4 listes $\mathbf{V_n^N}, \mathbf{V_n^E}, \mathbf{V_n^O}, \mathbf{V_n^S}$ qui regroupent les voitures par leurs voies. Supposons que :

$$\begin{cases} V \in \mathbf{V_n^S} \\ \forall V_k \in \mathbf{V_n}, t_{accordé}(V) < t_{accordé}(V_k) \end{cases}$$

3. Alors on fait passer N voitures de la liste $\mathbf{V_n^S}$, puis N voitures de la liste suivante, etc.

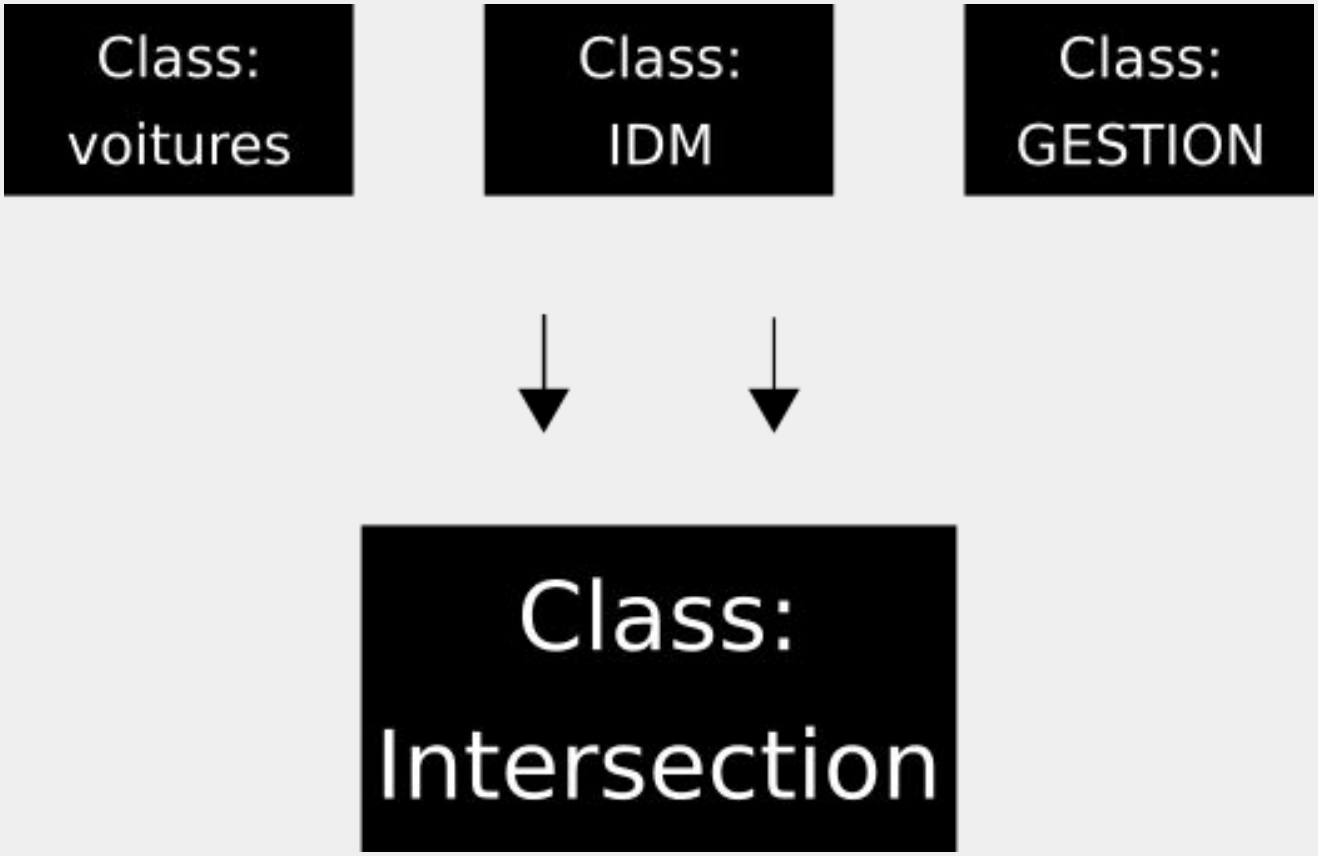




Méthode BATCH

Simulation avec python

Structure de la simulation



La classe Intersection hérite des classes Voitures, IDM et GESTION

Classe Voiture

- L'accélération est dirigée par le modèle **IDM**
- On fait l'approximation suivante pour intégrer l'accélération puis la vitesse (avec h petit)

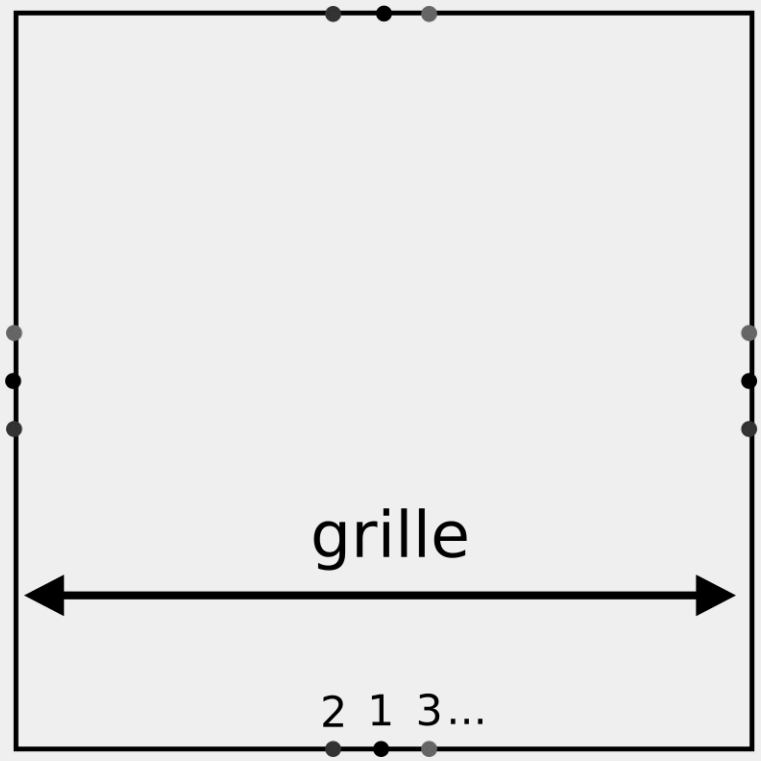
$$v(t+h) = v(t) + h \times a(t)$$

- les coordonnées de chaque voiture sont de la forme:

$$\text{coordonnées} = [x, y, \text{'pos'}]$$

- où 'pos' peut prendre comme valeur : 'b'; 'r'; 't'; 'l'. Ces lettres désignent la provenance de la voiture (bottom, right, ...)

- Points de départ



$$\begin{cases} \text{Points de départ}_n = \{\text{grille} + i \times (-1)^i \times \frac{\text{épaisseur de la route}}{n} \mid i \in [0, n]\} \\ n = \text{nombre de voies} \end{cases}$$

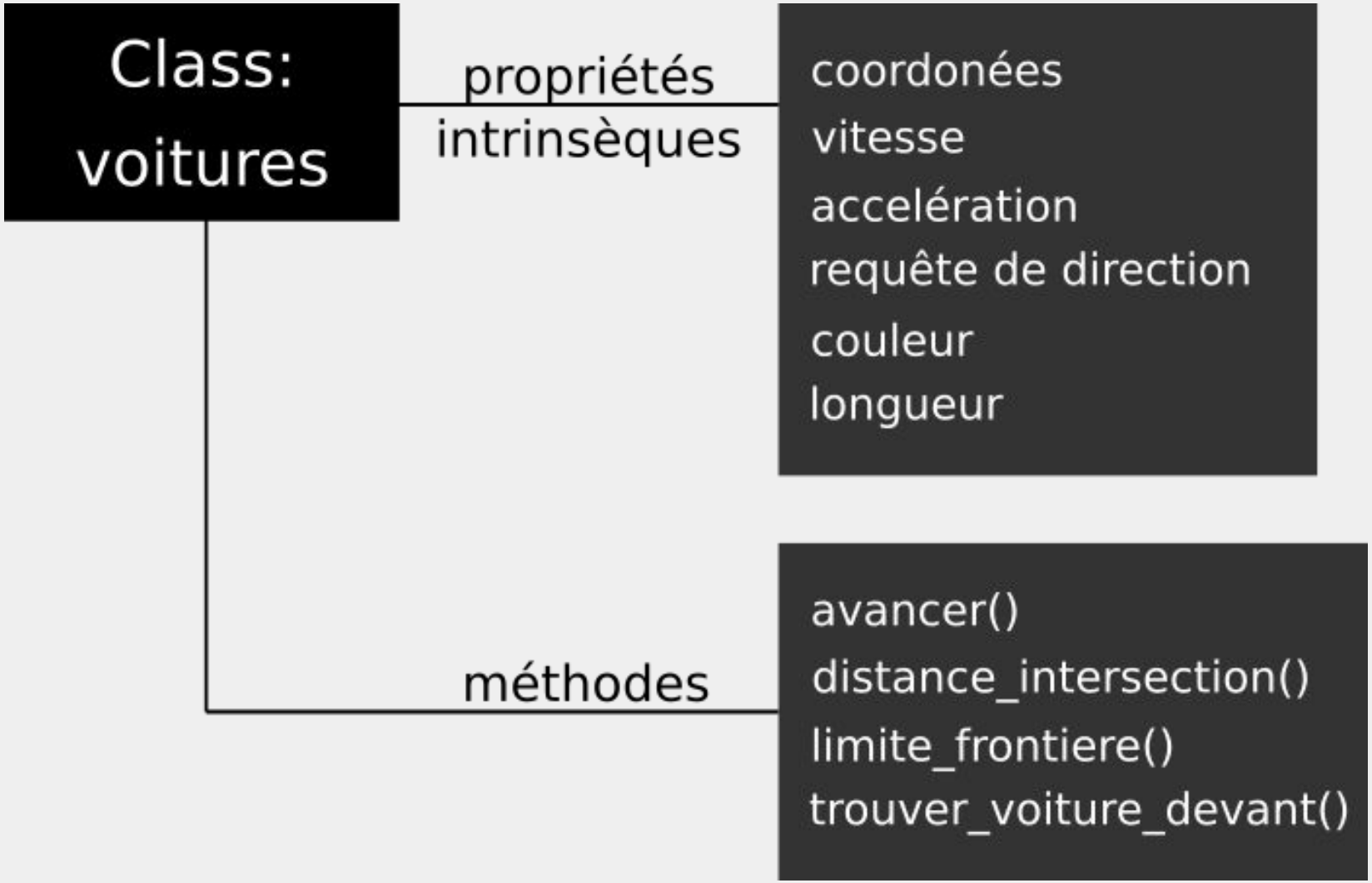
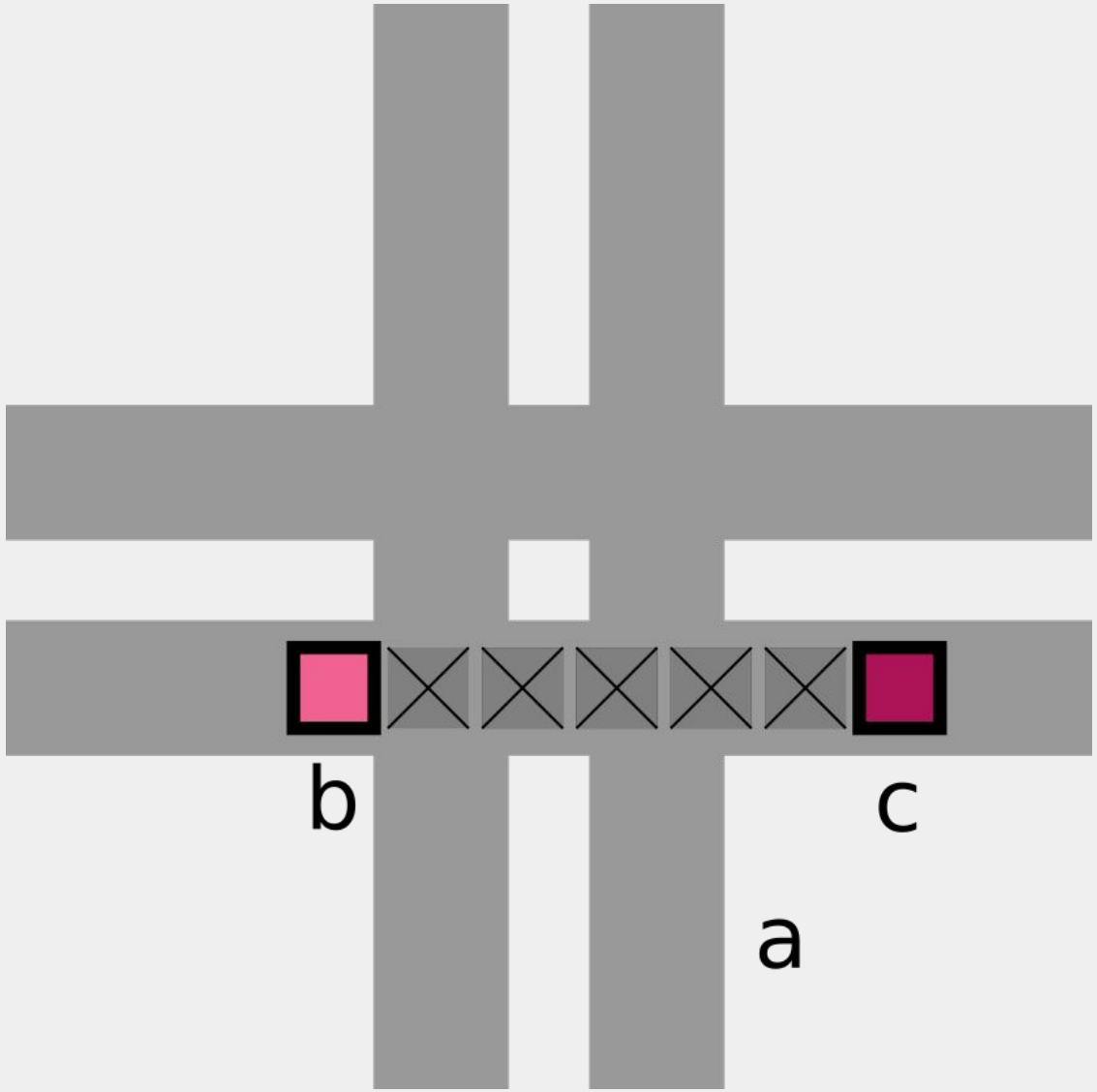


Schéma reprenant les caractéristiques de la classe voiture

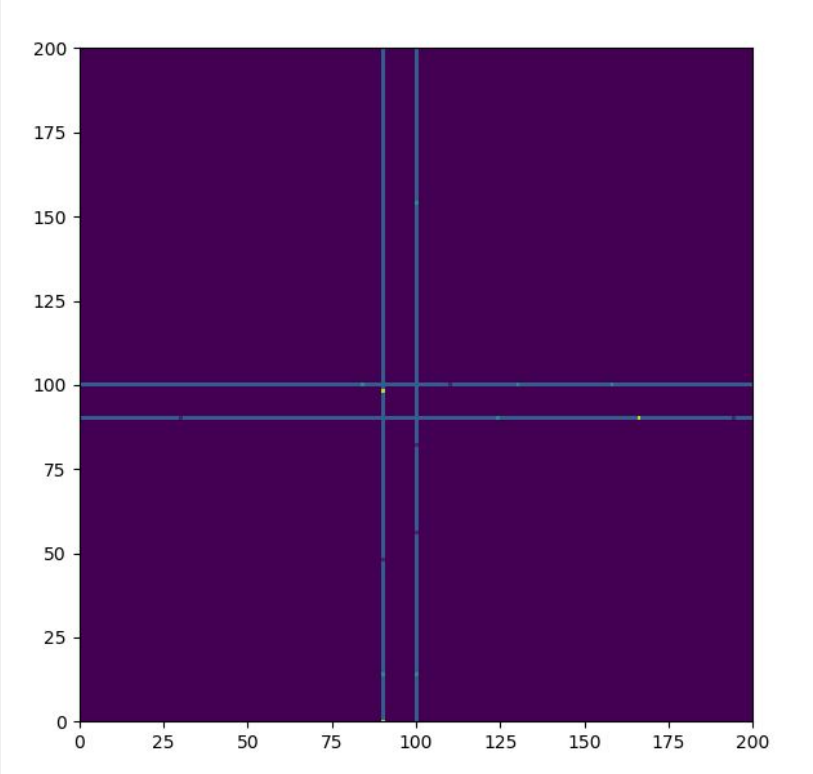
Algorithme de recherche de voiture devant



b et c sont des voitures. Les cases avec une croix sont celles qui ont été examinées.

Classe intersection

- l'intersection est représentée par une matrice carrée de taille n (liste de listes)



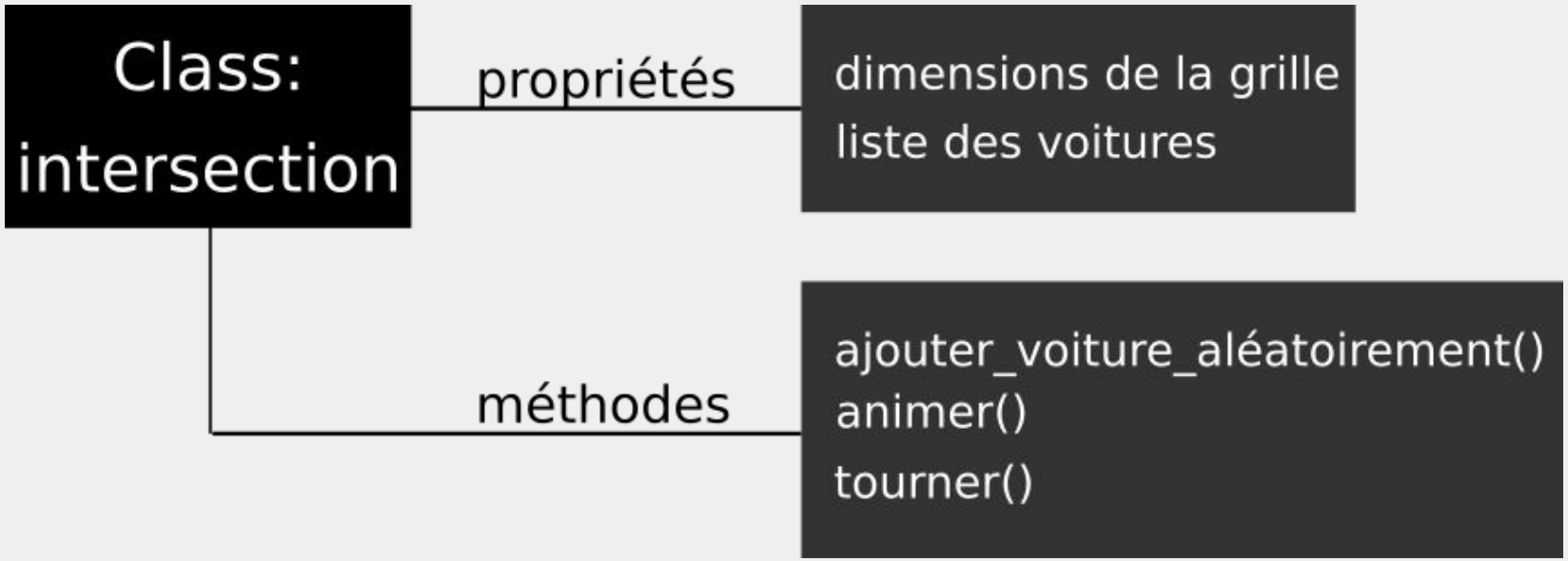


Schéma des caractéristiques de la classe intersection

Classe IDM

- La classe IDM permet de calculer à tout instant l'accélération puis la vitesse de chaque voiture. Dans cette simulation, les paramètres sont les mêmes pour chaque voiture

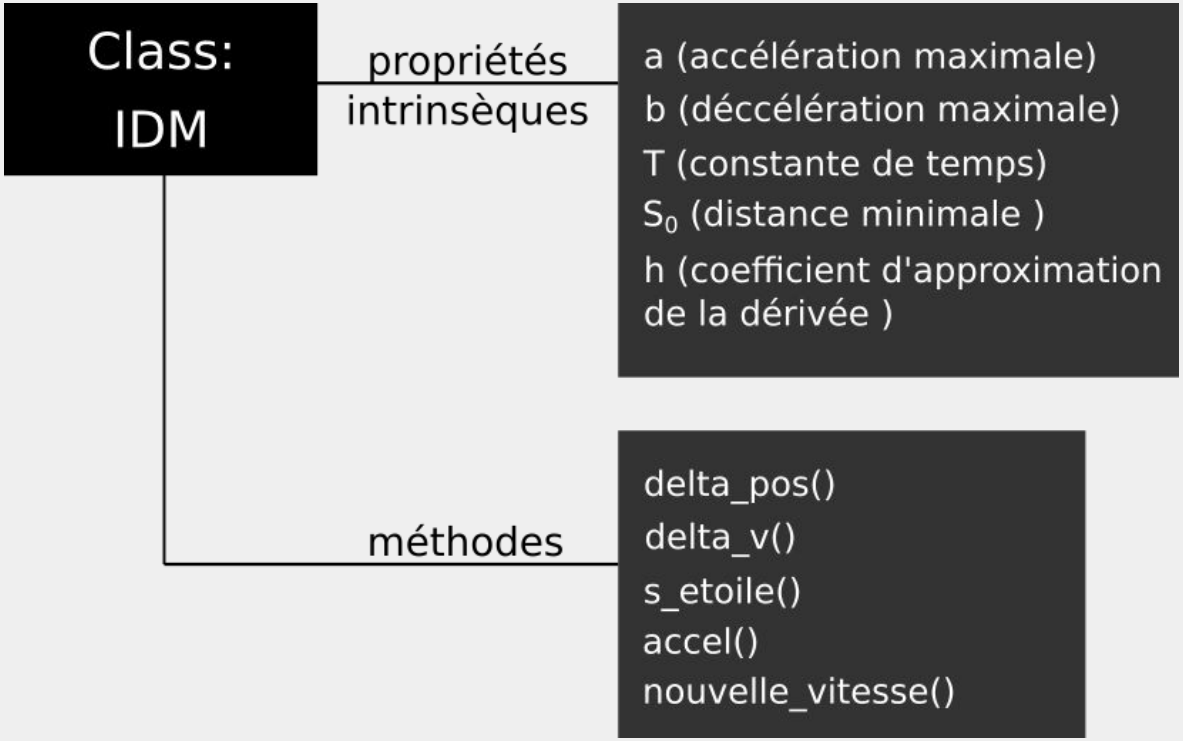


Schéma des caractéristiques de la classe intersection

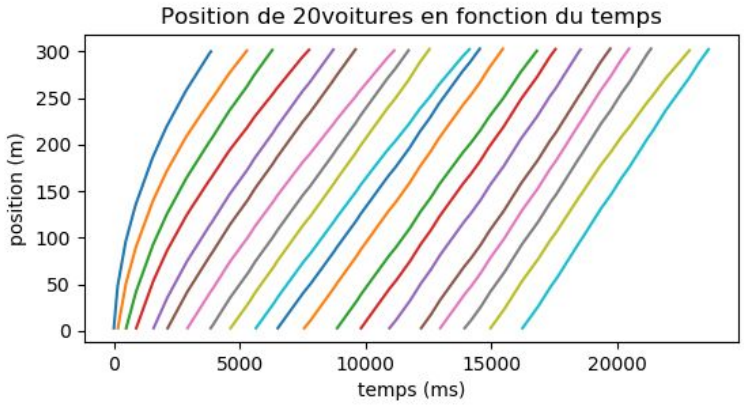
Complexité

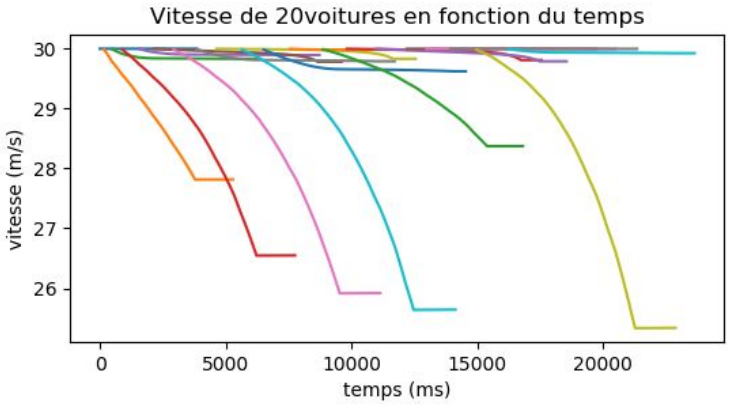
On pose :

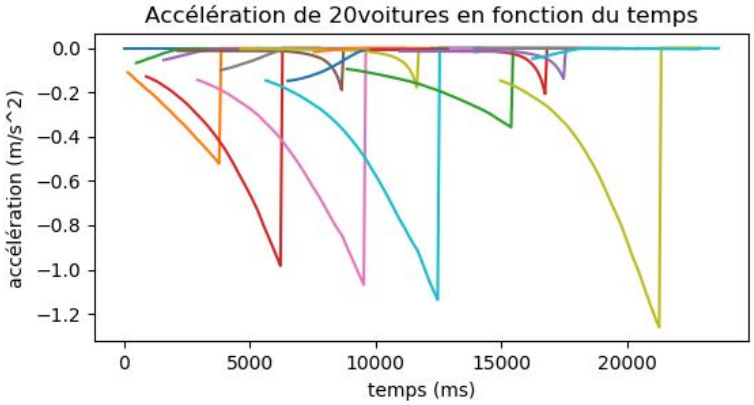
- l = nombre de voitures dans l'intersection
- m = nombre maximal de voitures
- n = nombre d'appels de la fonction 'animer'
- grille = longueur de la grille

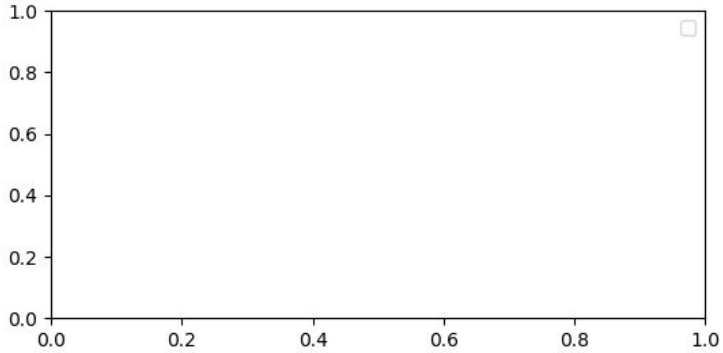
$$C(total) = n(99 + grille(1+l) + 3n + m)$$

Résultats

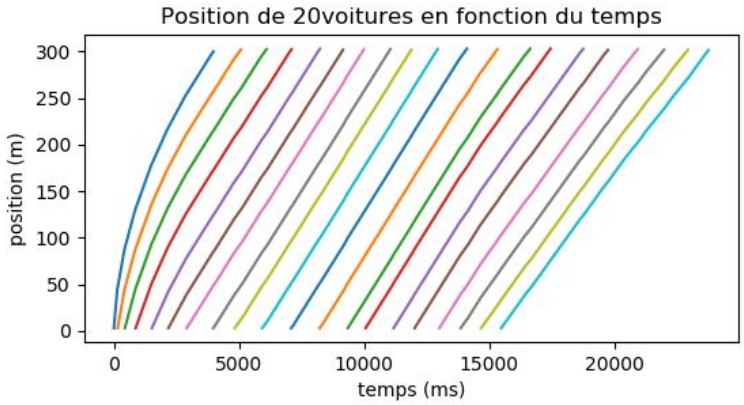


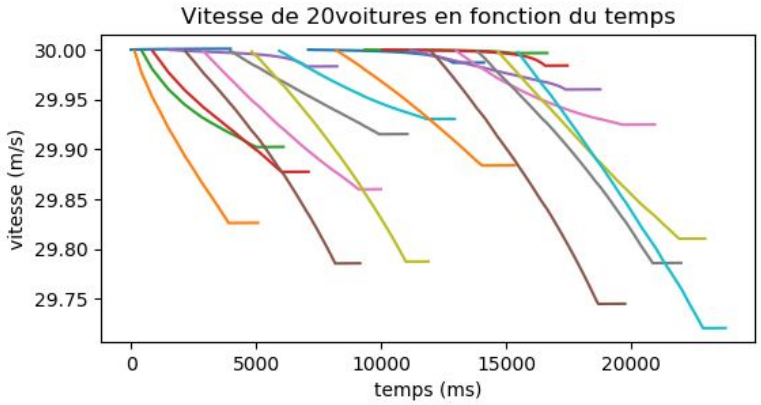


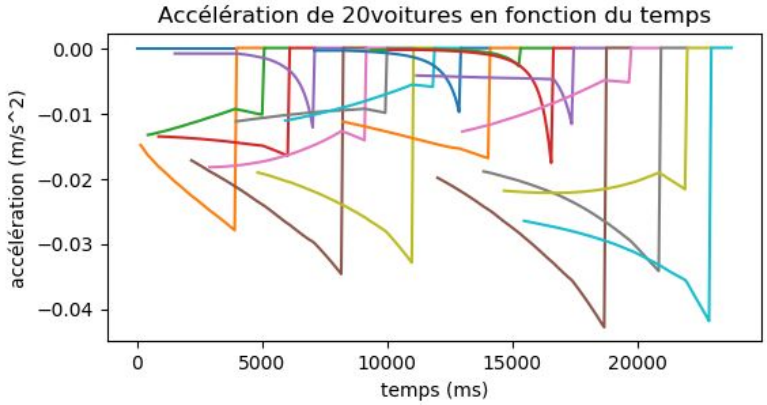


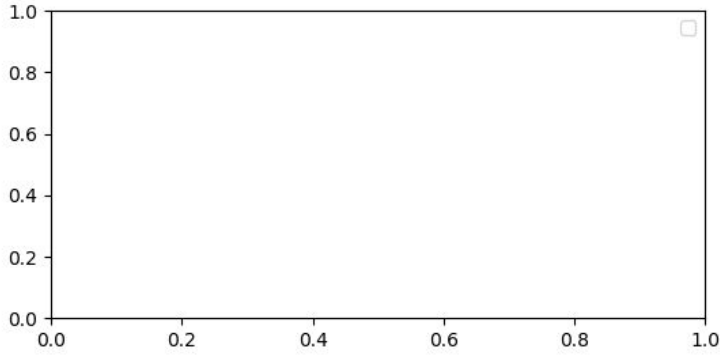


Résultats obtenus à partir de la simulation de voitures autonomes dans une intersection. vd = 33.3m/s, a = .73m/s^2, b = 1.67m/s^2, s0=2m, delta = 4, T=0.5s

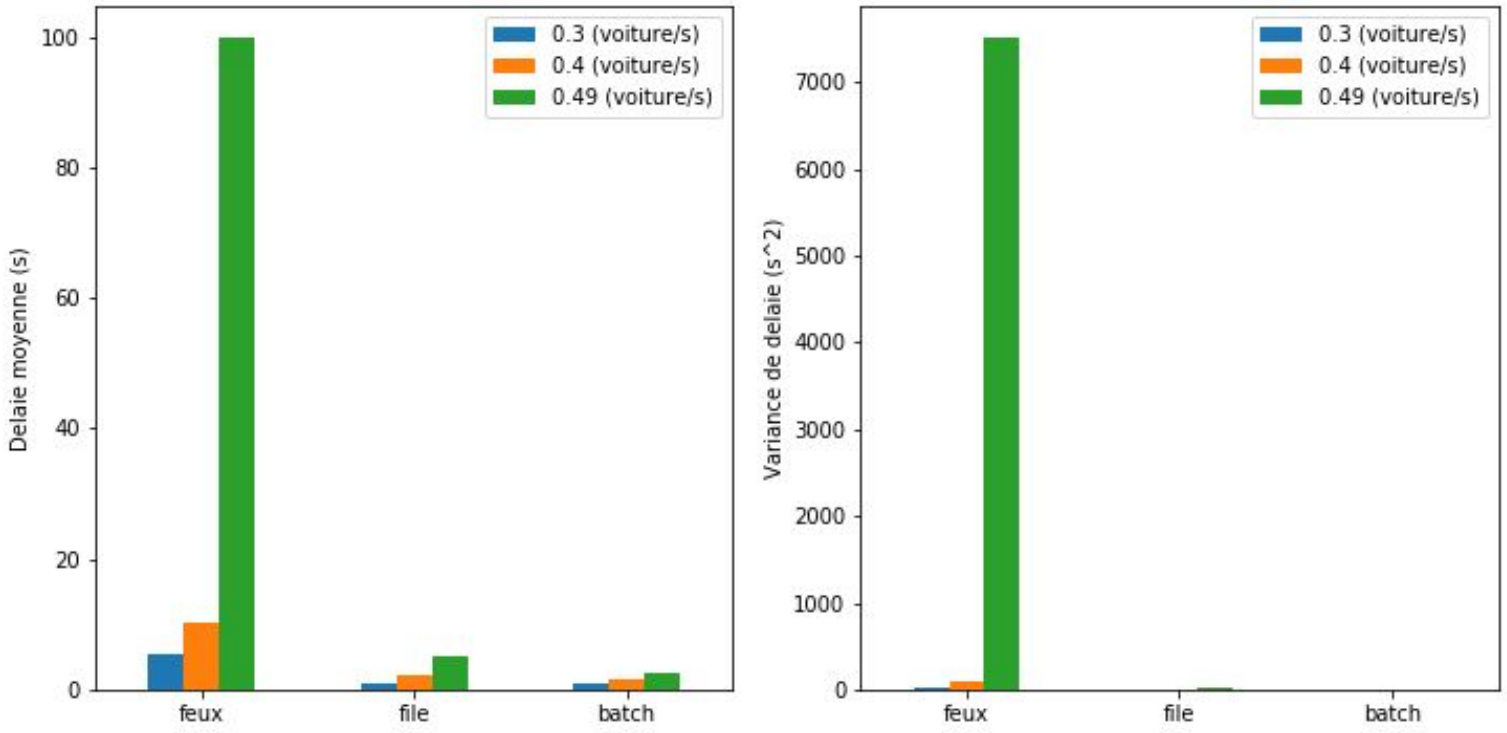








Résultats obtenus à partir de la simulation de voitures autonomes dans une intersection. vd = 33.3m/s, a = 1m/s^2, b = 1.67m/s^2, s0=2m, delta = 4, T=0.1s



Comparaison des délais de temps de passage en fonction de 3 méthodes: des feux tricolores, un principe de fil et la méthode BATCH

Conclusion