Évolution d’une file de voiture en fonction du comportement des automobilistes.

1 Description du problème

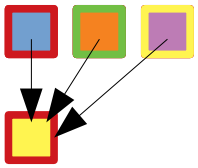
Imaginons une file de voiture et décrivons son évolution possible. Des automobilistes peuvent rentrer dans la file, d’autres en sortir. Certains doivent freiner à cause d’un ralentissement, d’autres accélérer pour retrouver la vitesse autorisée. Un accident peut survenir. Certains ont un comportement nerveux et d’autres pratiquent l’écoconduite. Certains se basent sur la voiture précédente pour anticiper le freinage. Il serait intéressant d’obtenir des statistiques après avoir laissé fonctionner le modèle selon certaines hypothèses.

Pour modéliser cette situation on utilise des automates cellulaires à une dimension. Chaque voiture est représentée par une case dans une ligne (la file de voitures) et chaque ligne représente la situation à l’instant t. L’évolution dans le temps se fait du haut vers le bas, on obtient un diagramme en 2D qui s’étend sur une longueur correspondant au temps de la simulation. Les lignes peuvent aussi grandir ou rétrécir en largeur au gré des rentrées et sorties des voitures dans la file. La première ligne peut être remplie aléatoirement avec un nombre aléatoire de voitures, d’au minimum une voiture, sans espace entre les cellules. Pour simuler un feu de circulation ou un rond-point on peut faire entrer un début de ligne une voiture fictive ayant un comportement mémorisé comme freiner, s’arrêter ou ralentir, et sortir.

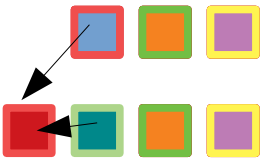
2 Présentation d’un automate cellulaire à une dimension

Un automate est une case dans une ligne. Cette case a plusieurs états possibles qui peuvent être représentés par des symboles, des numéros, des hachures ou des couleurs, ou d’autres encore …

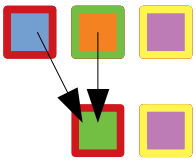
De l’instant t à l’instant t+1 chaque case peut changer d’état. Ce changement se fait en fonction de l’état des cases environnantes et d’une certaine probabilité. Ce terme de cases environnantes ou de voisinage reste à définir pour bien répondre à la situation. On peut opérer un décalage de certaines cases dans l’une ou l’autre direction de façon à ajouter ou à retirer des cellules avant d’appliquer les règles de transitions.



Règle de transition de t à t+1



Avec décalage à gauche



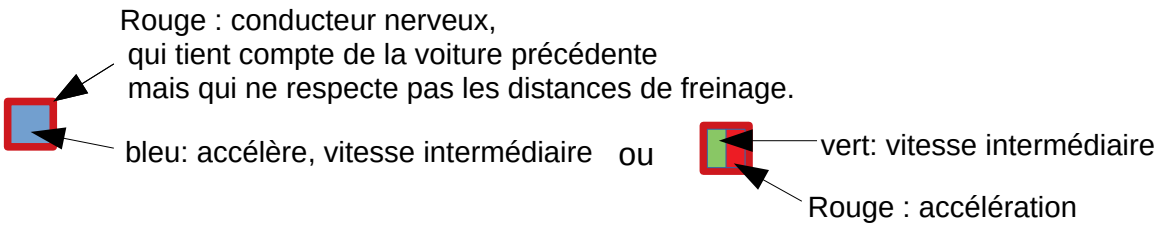
Avec décalage à droite

La représentation se fait dans une grille dans laquelle une ligne représente un instant et la ligne suivante, l’instant suivant.

3 Description du travail à effectuer

Il s'agit d'implémenter cette modélisation en Python avec Tkinter pour l'interface graphique ou en JavaScript et P5.js pour l'interface graphique. Il doit être possible de modifier les différents paramètres dans l'interface graphique. À la fin de la simulation dont le nombre d'itérations fait partie des paramètres, un résumé statistique à définir sera affiché à l'écran.

On peut considérer 3 comportements du conducteur qui induit trois types d'accélération ou de freinage possibles, 2 comportements selon qu'il tient compte ou non de la voiture précédente, 2 comportements selon le respect ou non, ce qui peut induire un accident, des distances de freinage. Cela fait donc 3x2x2=12 combinaisons possibles, à choisir lors de la création d'une cellule selon certaines probabilités. On peut indiquer le type de cellule par des bordures de cases de couleurs différentes (la membrane de la cellule). Il faut ajouter une cellule pour représenter un feu et une cellule pour représenter un rond-point.



Chaque case peut être en accélération ou en freinage de type 1, 2 ou 3, ou à vitesse constante. Il y a 4 types de vitesse, la vitesse limite autorisée, intermédiaire, lente ou à l'arrêt, soit 7x4 = 28 possibilités qui peuvent être représentées par 7 couleurs de 4 dégradés ou par deux parties à l'intérieur de la cellule.

Ces combinaisons peuvent naturellement être redéfinies en fonction de l'évolution du projet.

Il faut définir les règles de transitions, par exemple une voiture suivant une plus lente devra freiner selon son profil, si une collision se produit, la simulation se terminera après quelques itérations.

Pour les statistiques, on peut par exemple retenir comme paramètre la vitesse moyenne, la moyenne des accélérations et des freinages, et calculer le couple (moyenne, écart-type) ou tracer un diagramme en boîte.

4 Étapes du projet

1. Discussion autour du modèle pour déterminer les règles de transitions et choix du langage de programmation.

2. Discussion autour d'une analyse descendante : découpage du projet en plusieurs programmes indépendants, création d'un diagramme de GANTT avec le logiciel libre GanttProject.

3. Répartition des tâches et utilisation de Github (compte gratuit) pour partager les différentes versions du code et pour une synthèse des discussions.

4. Réalisation des programmes et discussion sur la répartition des tâches en fonction de leur avancement.

5. Mise en commun et définition des tests.

6. Créations de scénarios pour étoffer le rapport.

7. Discussion autour du rapport et écriture de celui-ci en LATEX.