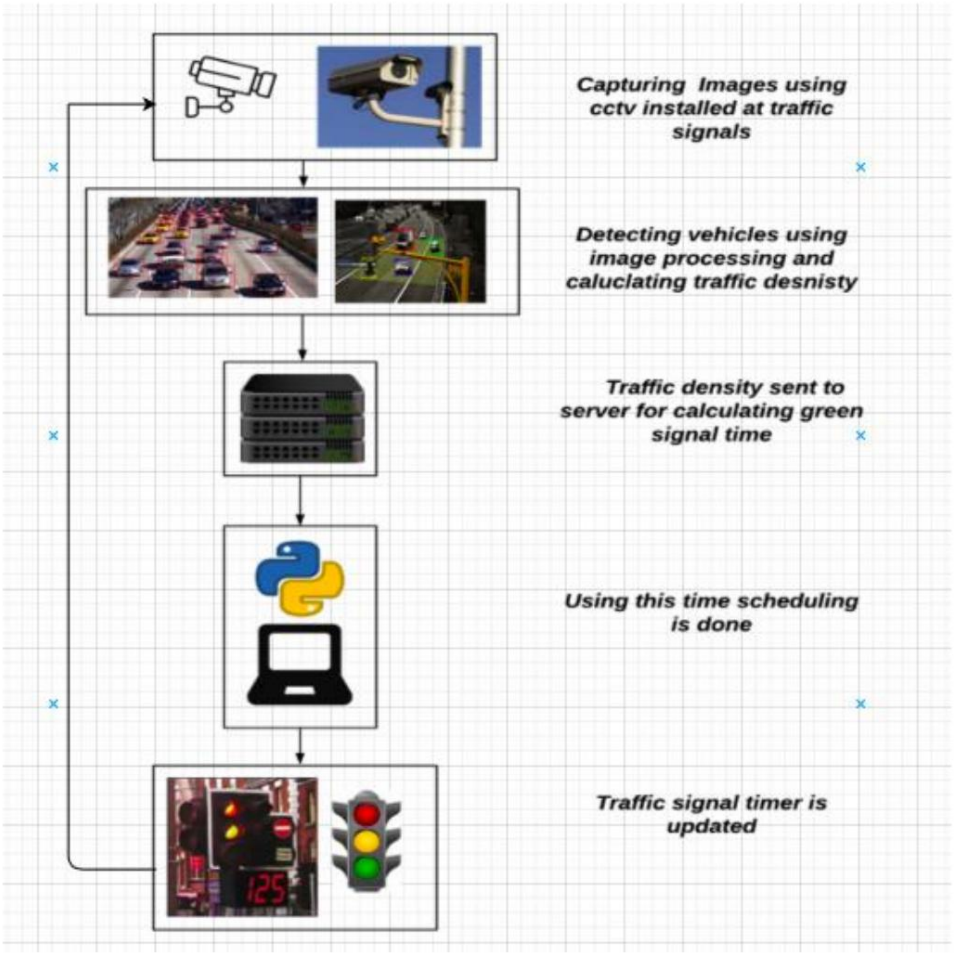
Minuterie de feux de circulation adaptatifs

Détails de mise en œuvre

Aperçu du système proposé

Notre système utilise une image des caméras de vidéosurveillance installées aux carrefours pour calculer la densité du trafic en temps réel grâce au traitement d'images et à la détection d'objets. Ce système se décompose en trois modules : détection de véhicules, algorithme de commutation des signaux et simulation. Comme illustré dans la figure ci-dessous, cette image est transmise à l'algorithme de détection de véhicules, qui utilise YOLO. Le nombre de véhicules de chaque catégorie (voitures, vélos, bus et camions) est détecté, ce qui permet de calculer la densité du trafic. L'algorithme de commutation des signaux utilise cette densité, entre autres facteurs, pour régler la minuterie du feu vert de chaque voie. Les durées des feux rouges sont mises à jour en conséquence. La durée du feu vert est limitée à une valeur maximale et minimale afin d'éviter la saturation d'une voie. Une simulation est également développée pour démontrer l'efficacité du système et la comparer au système statique existant.



Module de détection de véhicule

- Le système proposé utilise YOLO (You Only Look Once) pour la détection des véhicules, offrant la précision et le temps de traitement souhaités. Un modèle YOLO personnalisé a été formé pour la détection des véhicules, capable de détecter des véhicules de différentes catégories, comme les voitures, les motos, les poids lourds (bus et camions) et les pousse-pousse.

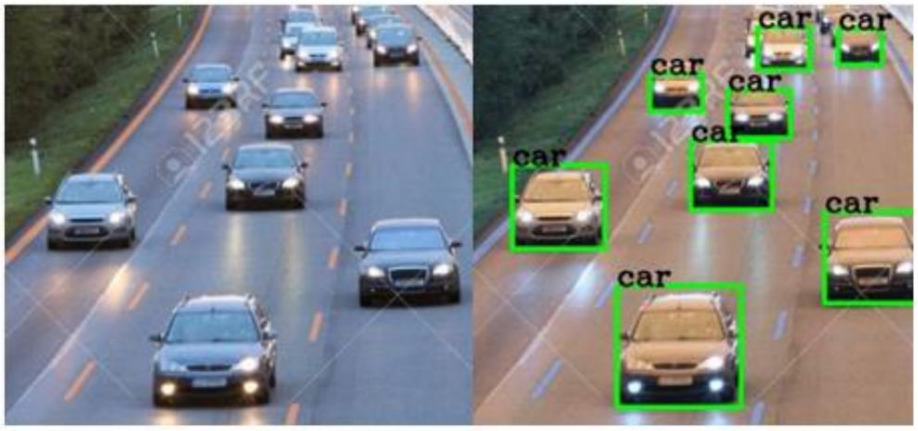
- L'ensemble de données pour la formation du modèle a été préparé en récupérant des images de Google et en les étiquetant manuellement à l'aide de LabelIMG, un outil d'annotation d'images graphiques.

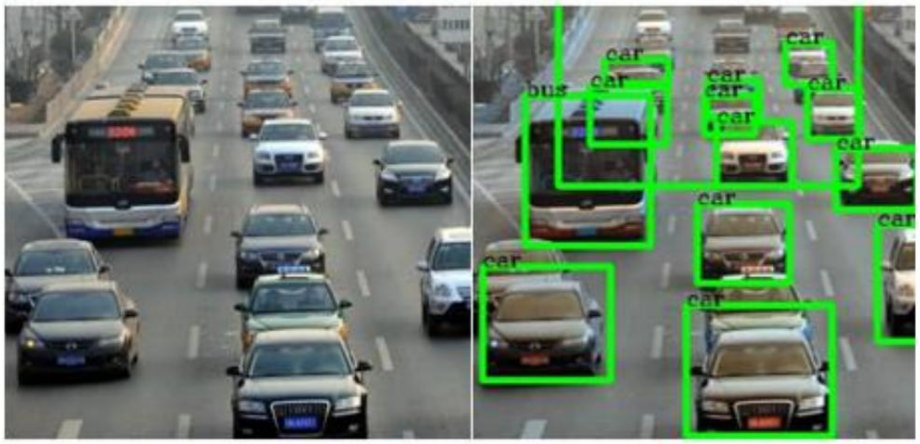
- Le modèle a ensuite été entraîné à l'aide des pondérations pré-entraînées téléchargées depuis le site web de YOLO. La configuration du fichier .cfg utilisé pour l'entraînement a été modifiée conformément aux spécifications de notre modèle. Le nombre de neurones de sortie de la dernière couche a été fixé au nombre de classes que le modèle est censé détecter en modifiant la variable « classes ». Dans notre système, ce nombre était de 4 : voiture, vélo, bus/camion et pousse-pousse. Le nombre de filtres doit également être modifié selon la formule 5 * (5 + nombre de classes), soit 45 dans notre système.
  cas.

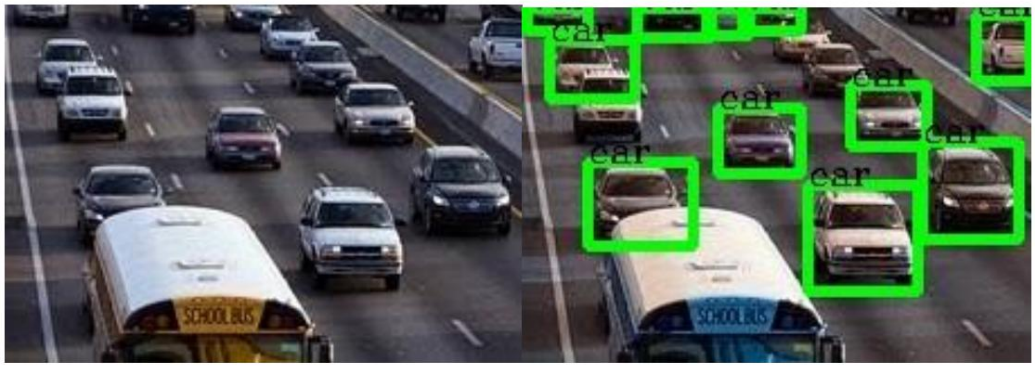
- Après ces modifications de configuration, le modèle a été entraîné jusqu'à ce que la perte soit significativement moindre et ne semble plus diminuer. Ceci a marqué la fin de l'entraînement, et les pondérations ont été mises à jour selon nos exigences.

- Ces pondérations ont ensuite été importées dans le code et utilisées pour la détection de véhicules à l'aide de la bibliothèque OpenCV. Un seuil est défini comme la confiance minimale requise pour une détection réussie. Une fois le modèle chargé et une image fournie, le résultat est affiché au format JSON, c'est-à-dire sous forme de paires clé-valeur, où les étiquettes sont des clés, et leur confiance et leurs coordonnées sont des valeurs. OpenCV permet également de dessiner les cadres de délimitation des images à partir des étiquettes et des coordonnées reçues.

Voici quelques images de la sortie du module de détection de véhicule :







Algorithme de commutation de signal

L'algorithme de commutation des signaux règle la minuterie du feu vert en fonction de la densité du trafic renvoyée par le module de détection de véhicules et met à jour les minuteries des autres feux rouges en conséquence. Il commute également cycliquement entre les signaux en fonction des minuteries.

L'algorithme prend en entrée les informations relatives aux véhicules détectés par le module de détection, comme expliqué dans la section précédente. Ces informations sont au format JSON, avec l'étiquette de l'objet détecté comme clé, la confiance et les coordonnées comme valeurs. Ces données sont ensuite analysées pour calculer le nombre total de véhicules de chaque classe. Ensuite, la durée du feu vert est calculée et attribuée à chaque feu, et la durée du feu rouge des autres feux est ajustée en conséquence. L'algorithme peut être étendu à n'importe quel nombre de feux à une intersection.

Les facteurs suivants ont été pris en compte lors du développement de l'algorithme : 1. Le
temps de traitement de l'algorithme pour calculer la densité du trafic, puis la durée du feu vert - cela détermine à quel moment l'image doit être acquise

2. Nombre de voies
3. Nombre total de véhicules de chaque classe comme les voitures, les camions, les motos, etc.
4. Densité du trafic calculée à l'aide des facteurs ci-dessus 5.
Temps ajouté en raison du décalage subi par chaque véhicule au démarrage et de l'augmentation non linéaire du décalage subi par les véhicules qui sont à l'arrière [13]
6. La vitesse moyenne de chaque classe de véhicule lorsque le feu vert s'allume, c'est-à-dire la vitesse moyenne temps nécessaire pour franchir le signal par chaque classe de véhicule [14]
7. La durée minimale et maximale du feu vert - pour éviter famine

Fonctionnement de l'algorithme

Lors de la première exécution de l'algorithme, l'heure par défaut est définie pour le premier feu du premier cycle, et les heures de tous les autres feux du premier cycle et des cycles suivants sont définies par l'algorithme. Un thread distinct est lancé pour gérer la détection des véhicules dans chaque sens, tandis que le thread principal gère le chronomètre du feu actuel. Lorsque le chronomètre du feu vert du feu actuel (ou le chronomètre du feu rouge du feu vert suivant) atteint 0 seconde, les threads de détection prennent un instantané du sens suivant. Le résultat est ensuite analysé et le chronomètre du feu vert suivant est défini. Tout cela se déroule en arrière-plan, pendant que le thread principal compte à rebours le chronomètre du feu vert actuel. Cela permet une affectation fluide du chronomètre et évite ainsi tout décalage. Une fois le chronomètre du feu vert du feu actuel à zéro, le feu suivant passe au vert pendant la durée définie par l'algorithme.

L'image est capturée lorsque le temps de passage au vert du prochain feu est de 0 seconde. Le système dispose ainsi de 5 secondes (équivalent à la durée du feu jaune) pour traiter l'image, détecter le nombre de véhicules de chaque classe présents, calculer la durée du feu vert et définir en conséquence les durées de ce feu et du feu rouge du feu suivant. Pour déterminer la durée optimale du feu vert en fonction du nombre de véhicules de chaque classe à un feu, les vitesses moyennes des véhicules au démarrage et leurs temps d'accélération ont été utilisés, ce qui a permis d'estimer le temps moyen de traversée d'une intersection par chaque classe. La durée du feu vert est ensuite calculée selon la formule ci-dessous.

$$GST = \frac{\sum_{vehicleClass} (NoOfVehicles_{vehicleClass} * AverageTime_{vehicleClass})}{(NoOfLanes + 1)}$$

où:

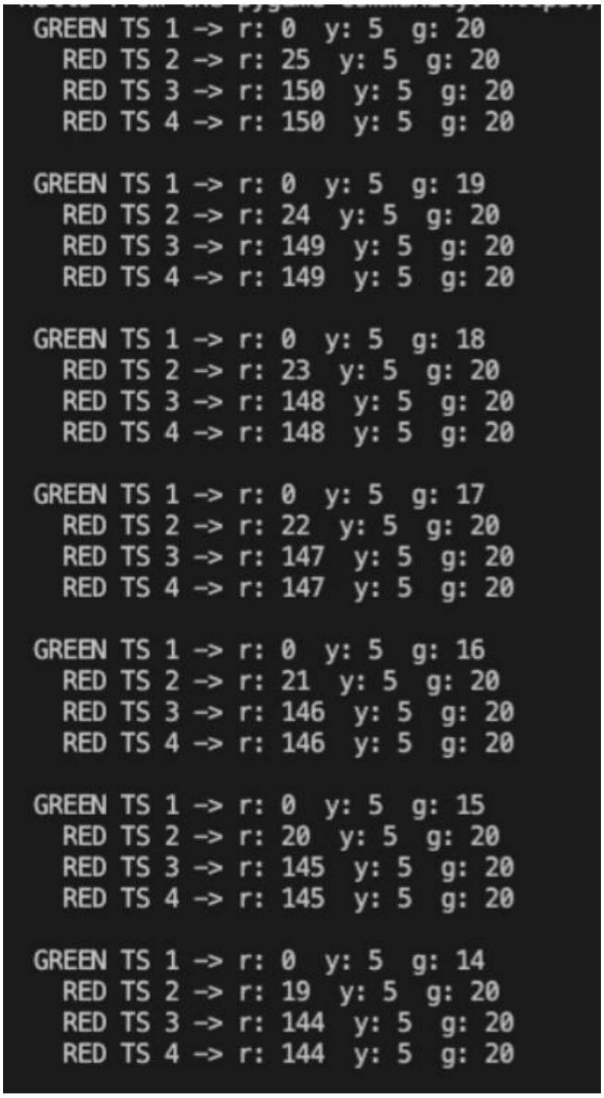
- La TPS est l'heure du feu vert •
noOfVehiclesOfClass est le nombre de véhicules de chaque classe de véhicule au signal tel que détecté par le module de détection de véhicule,
- averageTimeOfClass est le temps moyen que mettent les véhicules de cette classe pour traverser une intersection, et
- noOfLanes est le nombre de voies à l'intersection.

Le temps moyen de traversée d'une intersection par chaque catégorie de véhicule peut être défini en fonction de la localisation (région, ville, localité, voire intersection selon les caractéristiques de l'intersection), afin d'optimiser la gestion du trafic. Les données des autorités de transport concernées peuvent être analysées à cet effet.

Les feux s'allument de manière cyclique, et non selon la direction la plus dense. Ce principe est conforme au système actuel : les feux passent au vert les uns après les autres selon un schéma fixe, sans que les usagers ne modifient leur comportement ni ne créent de confusion. L'ordre des feux est également le même que dans le système actuel, et les feux jaunes ont également été pris en compte.

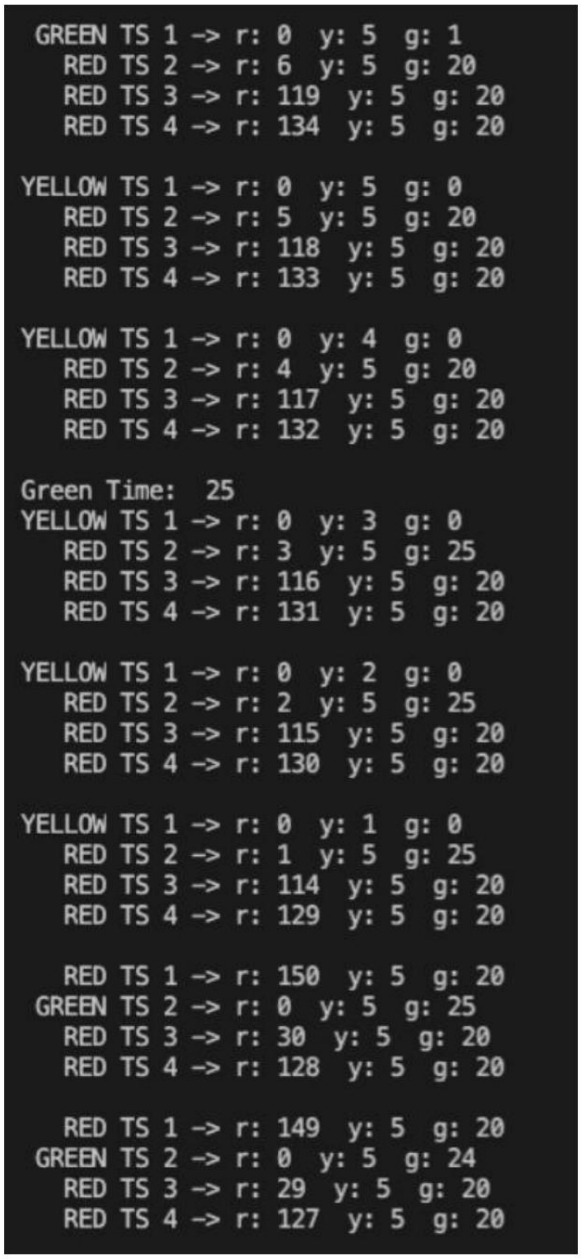
Ordre des signaux : Rouge → Vert → Jaune → Rouge

Voici quelques images de la sortie de l'algorithme de commutation de signal :



(i) : Initialement, tous les signaux sont chargés avec des valeurs par défaut, seul le temps du signal rouge du deuxième signal est défini en fonction du temps vert et du temps jaune du premier signal.

(ii) : La colonne de gauche indique l'état du feu (rouge, jaune ou vert), suivi du numéro du feu et des temporisations actuelles (rouge, jaune et vert). Ici, le feu 1 (TS 1) passe du vert au jaune. À mesure que le compte à rebours du jaune s'écoule, les résultats de l'algorithme de détection de véhicule sont calculés et un temps de vert de 9 secondes est renvoyé pour le TS 2. Cette valeur étant inférieure au temps minimum de vert de 10 secondes, le temps de vert du TS 2 est fixé à 10 secondes. Lorsque le temps jaune du TS 1 atteint 0, le TS 1 passe au rouge et le TS 2 au vert, et le compte à rebours continue. Le temps de rouge du TS 3 est également mis à jour comme la somme des temps jaune et vert du TS 2, soit 5 + 10 = 15.



(iii) : Après un cycle complet, le feu 1 passe à nouveau du vert au jaune. À mesure que le compte à rebours jaune s'écoule, les résultats de l'algorithme de détection de véhicule sont traités et un temps de feu vert de 25 secondes est calculé pour le feu 2. Comme cette valeur est supérieure au temps de feu vert minimal et inférieure au temps de feu vert maximal, le temps de feu vert du feu 2 est fixé à 25 secondes. Lorsque le temps jaune du feu 1 atteint 0, le feu 1 passe au rouge et le feu 2 au vert, et le compte à rebours continue. Le temps de feu rouge du feu 3 est également mis à jour comme la somme des temps jaune et vert du feu 2, soit 5 + 25 = 30.

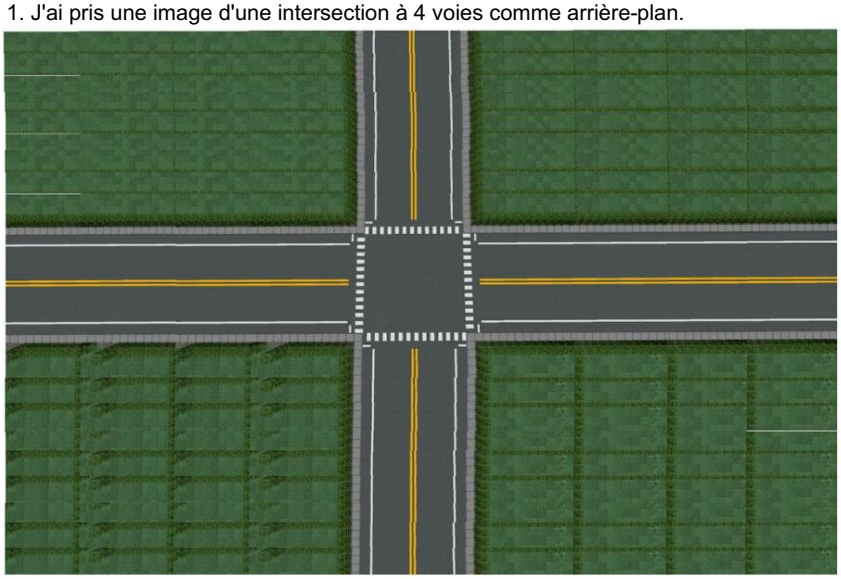
Module de simulation

Une simulation a été développée de toutes pièces avec Pygame pour simuler le trafic réel. Elle permet de visualiser le système et de le comparer au système statique existant. Elle comprend un carrefour à quatre voies avec quatre feux de circulation. Chaque feu est surmonté d'un chronomètre indiquant le temps restant pour passer du vert à l'orange, de l'orange au rouge ou du rouge au vert. Le nombre de véhicules ayant traversé l'intersection est également affiché à côté de chaque feu. Des véhicules tels que des voitures, des motos, des bus, des camions et des pousse-pousse arrivent de toutes les directions. Afin de rendre la simulation plus réaliste, certains véhicules sur la voie la plus à droite tournent pour traverser l'intersection.

Le fait qu'un véhicule tourne ou non est également défini à l'aide de nombres aléatoires lors de la génération du véhicule.
Il contient également un minuteur qui affiche le temps écoulé depuis le début de la simulation

Étapes clés du développement de la simulation

1. J'ai pris une image d'une intersection à 4 voies comme arrière-plan.

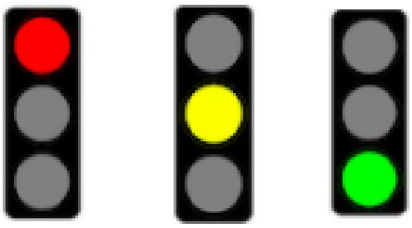


2. J'ai rassemblé des images de vue de dessus de voiture, de vélo, de bus, de camion et de pousse-pousse.

3. Redimensionnez-les.

4. Faites-les pivoter pour les afficher dans différentes directions.

5. Images rassemblées des feux de circulation - rouges, jaunes et verts.



6. Code : Pour restituer l'image appropriée du signal selon qu'il est rouge, vert ou jaune.

7. Code : Pour afficher l'heure actuelle du signal, c'est-à-dire le temps restant pour qu'un signal vert passe au vert Un feu jaune ou rouge passe au vert, ou un feu jaune passe au rouge. La durée du feu vert est définie selon un algorithme, en fonction du nombre de véhicules à chaque feu. La durée du feu rouge des autres feux est mise à jour en conséquence.

8. Génération de véhicules selon la direction, la voie, la classe de véhicule et la possibilité de tourner ou non, le tout étant déterminé par des variables aléatoires. La répartition des véhicules entre les quatre directions est contrôlable. Un nouveau véhicule est généré et ajouté à la simulation toutes les 0,75 seconde.

9. Code : Pour la façon dont les véhicules se déplacent, chaque classe de véhicule a une vitesse différente, il y a un écart entre 2 véhicules, si une voiture suit un bus, alors sa vitesse est réduite pour qu'elle ne s'écrase pas dans le bus.

10. Code : Comment ils réagissent aux feux de circulation, c'est-à-dire s'arrêter au feu jaune et au feu rouge, et avancer au feu vert. S'ils ont franchi la ligne d'arrêt, ils continuent d'avancer si le feu passe au jaune.

11. Code : Pour afficher le nombre de véhicules ayant franchi le signal.

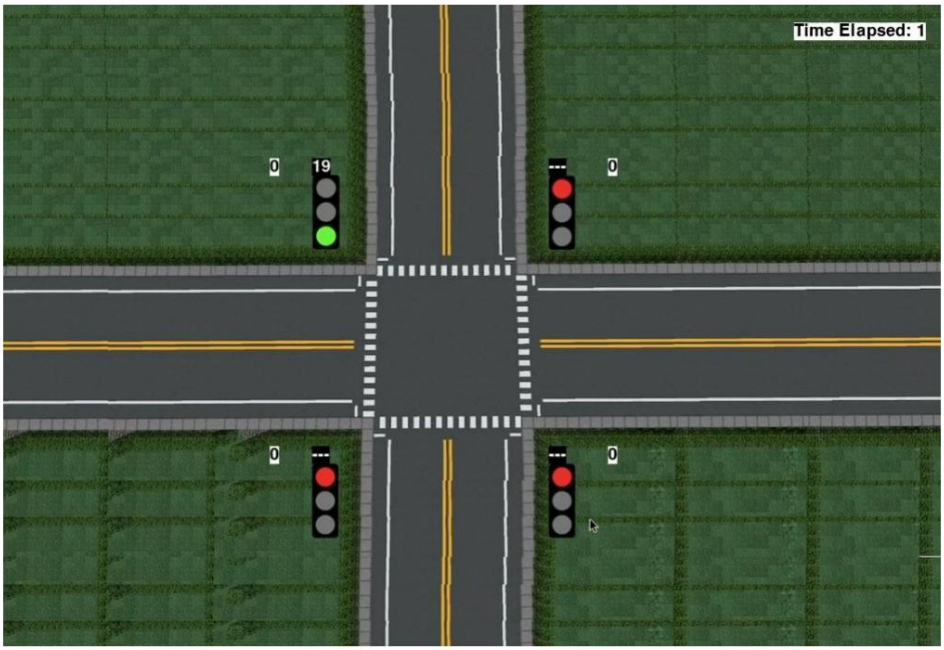
12. Code : Pour afficher le temps écoulé depuis le début de la simulation.

13. Code : Pour mettre à jour le temps écoulé au fur et à mesure de la progression de la simulation et quitter lorsque le temps écoulé est égal au temps de simulation souhaité, puis imprimer les données qui seront utilisées pour la comparaison et l'analyse.

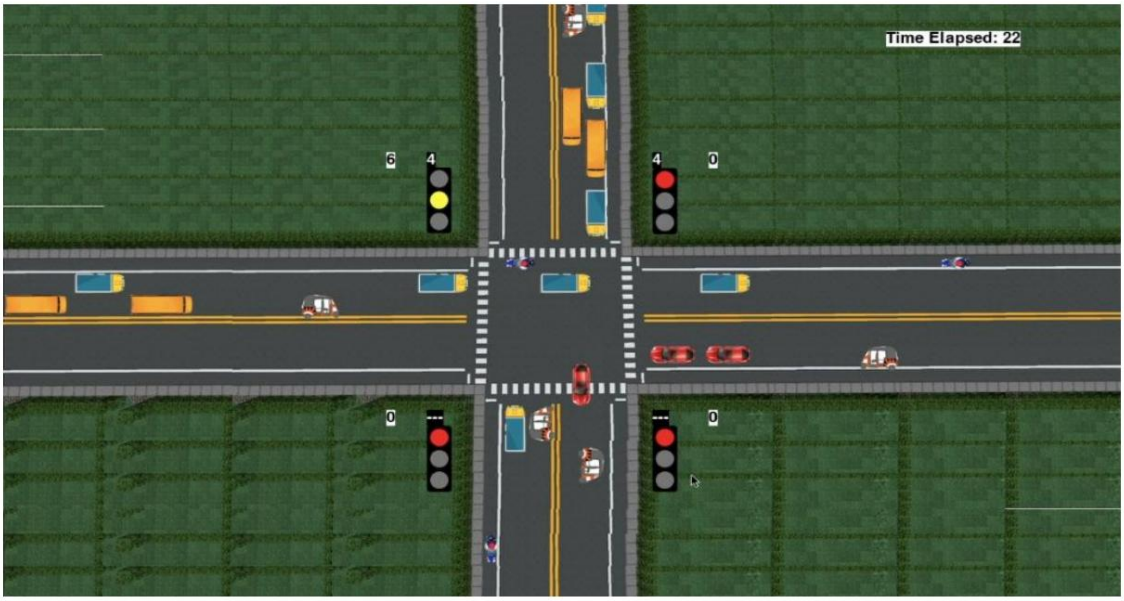
14. Pour rendre la simulation plus proche de la réalité, même s'il n'y a que 2 voies dans l'image, nous ajoutons une autre voie à gauche de celle-ci qui n'a que des vélos, ce qui est généralement le cas dans de nombreuses villes.

15. Véhicules tournant et traversant l'intersection dans la simulation pour la rendre plus réaliste.

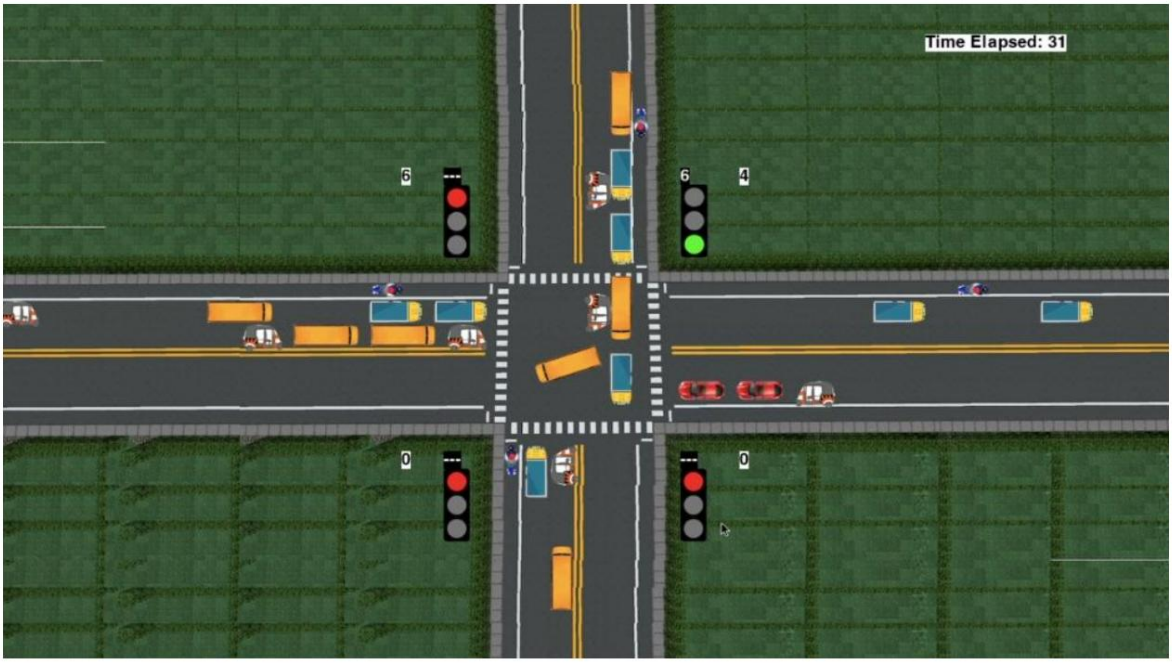
Voici quelques images de la simulation finale :



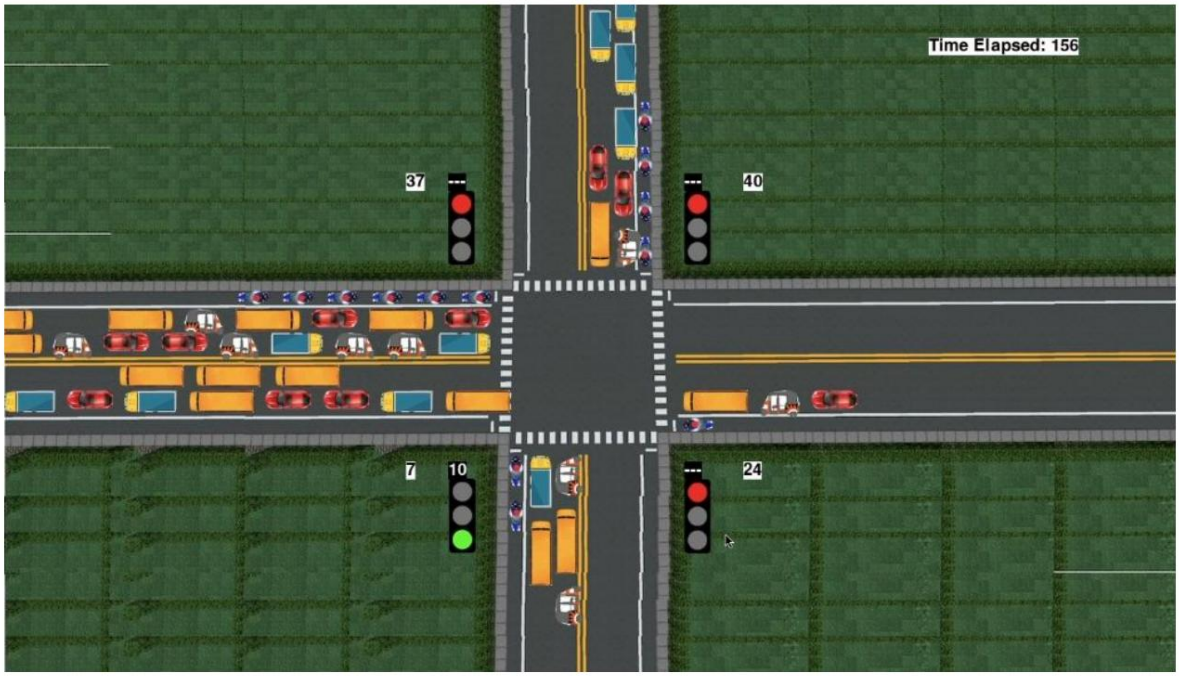
(i) : Simulation juste après le démarrage : feux rouges et verts, décompte de la durée du signal vert à partir de 20 secondes par défaut, et temps d'inactivité du signal rouge suivant. Lorsque le signal est rouge, l'affichage est inactif jusqu'à 10 secondes. Le nombre de véhicules ayant traversé est indiqué à côté du signal, tous à 0 initialement. Le temps écoulé depuis le début de la simulation est indiqué en haut à droite.



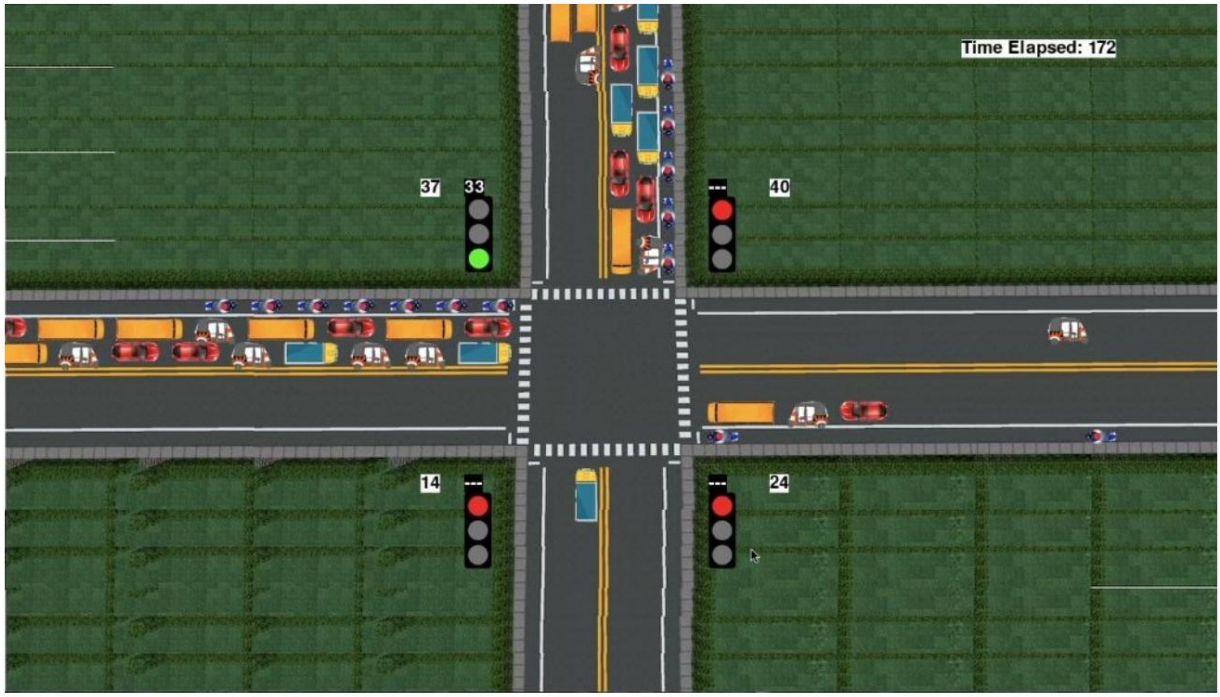
(ii) : Simulation montrant le temps restant avant le feu jaune et le feu rouge. Lorsque le temps restant avant le feu rouge est inférieur à 10 secondes, le compte à rebours est affiché afin que les véhicules puissent démarrer et se préparer à circuler dès que le feu passe au vert.



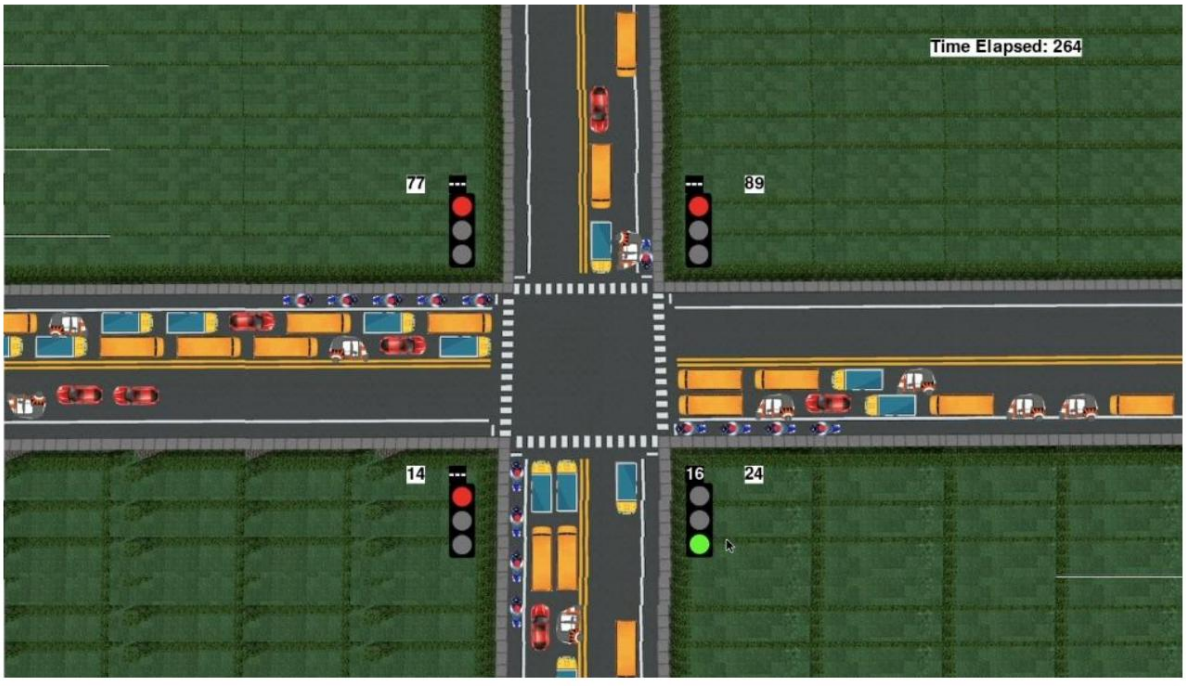
(iii) : Simulation montrant des véhicules tournant



(iv) : Simulation montrant un temps de passage au vert pour les véhicules en mouvement, fixé à 10 secondes en fonction du nombre de véhicules dans cette direction. Comme on peut le constater, le nombre de véhicules est bien inférieur ici par rapport aux autres voies. Avec le système statique actuel, le temps de passage au vert aurait été le même pour tous les feux, environ 30 secondes. Mais dans ce cas, la majeure partie de ce temps aurait été gaspillée. Or, notre système adaptatif détecte qu'il y a peu de véhicules et fixe le temps de passage au vert en conséquence, soit 10 secondes dans ce cas.



(v) : Simulation montrant le temps de passage au vert du signal pour les véhicules se déplaçant vers la droite fixé à 33 secondes en fonction des véhicules dans cette direction.



(vi) : Simulation montrant le temps de passage au vert du signal pour les véhicules se déplaçant vers la gauche fixé à 24 secondes en fonction des véhicules dans cette direction.