CAHIER DES CHARGES

Système Intelligent de Gestion du Trafic Basé sur la Segmentation Instantanée et l'Apprentissage par Renforcement

1. Contexte Général et Problématique

La gestion du trafic urbain repose encore sur des systèmes à feux statiques, incapables de s'adapter aux conditions réelles de circulation. L'émergence des caméras intelligentes et des techniques avancées de **traitement d'images** permet aujourd'hui une analyse fine du trafic routier.

La **segmentation instantanée (Instance Segmentation)** constitue une avancée majeure, car elle permet non seulement de détecter les objets, mais aussi d'identifier **chaque instance individuellement au niveau pixel**, ce qui est essentiel pour une estimation précise de l'occupation des voies.

2. Objectifs du Projet

Objectif général :

Concevoir un **système intelligent de contrôle de trafic** utilisant la **segmentation instantanée** pour analyser visuellement le trafic et un module d'**apprentissage par renforcement** pour optimiser dynamiquement les feux de circulation.

Objectifs spécifiques :

- Appliquer le traitement d'images classique.
- Utiliser la segmentation instantanée pour identifier chaque véhicule.
- Calculer des métriques précises de trafic.
- Prioriser les véhicules d'urgence.
- Comparer un mode statique à un mode intelligent.

3. Architecture Générale

Backend Python :

- Simulation du trafic (SUMO)
- Traitement d'images (OpenCV)
- Segmentation instantanée (YOLOv8-Seg / Mask R-CNN)
- Prise de décision intelligente (DQN ou règles avancées)
- API REST (FastAPI)

Frontend :

- Tableau de bord interactif
- Visualisation des indicateurs clés

4. Segmentation Fonctionnelle du Projet

- Segment 1 : Simulation du Trafic (Environnement)

Objectif

Créer un environnement réaliste de circulation routière.

Fonctionnalités

- Réseau routier simple (1 à 2 intersections).
- Génération du trafic (faible / fort).
- Feux statiques pour comparaison.

Livrable

- Simulation fonctionnelle sous SUMO.

- Segment 2 : Traitement d'Images Classique (Pré-traitement)

Objectif

Améliorer la qualité des images avant l'analyse avancée.

Techniques utilisées

- Filtrage Gaussien (réduction du bruit)
- Filtrage Médian
- Amélioration du contraste
- Détection de contours (Canny)
- Morphologie mathématique :
    - érosion
    - dilatation
    - ouverture / fermeture

Livrable

- Images avant / après traitement
- Justification scientifique des filtres

- Segment 3 : Segmentation Instantanée (Cœur Vision du Projet)

Objectif

Identifier **chaque véhicule individuellement** par masque pixel.

Méthodes

- Utilisation de **YOLOv8-Seg** (ou Mask R-CNN)
- Modèle pré-entraîné (pas d'entraînement obligatoire)

Objets segmentés

- Véhicules légers
- Poids lourds
- Véhicules d'urgence

Sorties

- Masque par instance
- Classe de l'objet
- Surface occupée par véhicule
- Position par voie

Livrables

- Visualisation des masques
- Comptage précis des véhicules
- Estimation d'occupation des voies

- Segment 4 : Extraction des Métriques de Trafic

Objectif

Transformer les résultats de la segmentation en données exploitables.

Métriques calculées

- Nombre de véhicules par voie
- Surface totale occupée
- Longueur des files d'attente
- Densité par voie
- Détection de véhicule d'urgence (flag)

Livrable

- Données structurées pour l'IA

- Segment 5 : Intelligence Décisionnelle

Objectif

Optimiser la gestion des feux de circulation.

Approche

- DQN simplifié ou règles intelligentes
- États : densité, surface occupée, attente
- Actions : choix de la phase de feu

Fonction de récompense

Rt=w1·De´bit−w2·Attente−w3·CongestionR_t = w_1 \cdot Débit - w_2 \cdot Attente - w_3 \cdot CongestionRt=w1·De´bit−w2·Attente−w3·Congestion

Priorité absolue

- Si véhicule d'urgence détecté → feu vert immédiat

Livrable

- Algorithme décisionnel fonctionnel
- Comparaison avec feux statiques

- Segment 6 : API et Interface

Objectif

Visualiser les performances du système.

Backend

- API FastAPI :
    - /metrics
    - /traffic-state
    - /emergency-status

Frontend

- Dashboard React :

- KPIs en temps réel
- Graphiques comparatifs
- Alerte **MODE URGENCE**

5. Indicateurs de Performance (KPIs)

- Temps d'attente moyen
- Nombre de véhicules par voie
- Surface occupée par voie
- Débit de trafic
- Comparaison :
  - Mode statique
  - Mode intelligent

6. Stack Technique

Domaine	**Technologie**
Langage	Python 3.9+
Traitement d'images	OpenCV
Segmentation instantanée	YOLOv8-Seg / Mask R-CNN
Simulation	SUMO + TraCI
IA	PyTorch
API	FastAPI
Frontend	React