SMART TMC – Vehicle Detection and Reporting System

Project Overview

SMART TMC is a smart traffic video analysis system designed to detect, track, and classify vehicles using advanced computer vision techniques. The project utilizes the YOLOv8 object detection model to accurately identify moving vehicles, logs their crossing times, captures their images, and generates detailed Excel reports for further analysis
It is developed with efficiency and flexibility in mind, optimized to run smoothly on Google Colab, and capable of processing long videos segmented by hour. It is particularly useful for traffic engineers, urban planners, and researchers needing structured insights from surveillance footage

Objectives

- Detect and count vehicles in surveillance videos
- Classify vehicle types using deep learning
- Divide detections into left and right lanes
- Save visual evidence (images) and timestamps of each vehicle
- Automatically export all results into organized Excel reports
- Support long video durations and hourly segmentation

Technical Stack

- Language: Python
- Libraries: OpenCV, Ultralytics YOLOv8, Pytesseract, OpenPyXL, Matplotlib, NumPy
- Platform: Google Colab
- Input: Any video file (typically uploaded to Google Drive)
- Output: Excel reports + cropped car images + annotated video

Key Features

- **YOLOv8-based Detection**
  - The system uses YOLOv8 (e.g., yolov8m.pt or yolov8x.pt) to detect vehicles in each frame.
  - Object tracking is implemented to handle continuous motion across frames.

- **Line-Crossing Logic**
  - Users define horizontal lines to represent virtual checkpoints.
  - Vehicles are counted when crossing these lines within a specified offset range.
  - Direction-based separation: detections are categorized into Left and Right lanes.
- **Speed-aware Tracking**
  - A tracking dictionary manages object IDs, positions, and time gaps to avoid double-counting.
  - Vehicles moving faster are given shorter cooldown windows to enhance accuracy in high-traffic footage.
- **Image Capture and Cropping**
  - When a vehicle crosses the counting line, a cropped image is saved.
  - The frame is split by direction, so only the relevant section (left/right) is stored.
- **OCR-Based Metadata Extraction**
  - The first frame of the video is analyzed using Tesseract OCR to automatically extract:
  - The start time of the video
  - The site code (e.g., “ATC,52”)
  - If OCR fails, manual input is requested.
- **Automated Excel Reports**
  - For every 15-minute interval, an Excel file is generated per direction.
  - Each file contains:
  - Timestamps
  - Cropped images of each detected vehicle
  - Vehicle type (classified again using YOLO)
  - A summary sheet shows the count of each vehicle type.

Performance Benchmark

- **On Google Colab with a standard GPU runtime:**
  - 1 hour of video is processed in approximately 17 minutes, including detection, classification, and Excel export.
  - Frame skipping and optimized thresholds are applied for balance between speed and accuracy.

Project Output

- output.mp4: Annotated video with line overlays and live count.
- car_screenshots : Folder containing:
- Cropped car images
- Excel reports (.xlsx) per 15-minute segment and lane

Use Cases

- Urban traffic monitoring
- Smart city data collection
- Research on traffic density, flow, and peak hours
- Surveillance analytics and automation

Future Improvements

- Add GUI interface (Streamlit or Gradio).
- Support live webcam or RTSP stream input.
- Improve OCR with advanced preprocessing.
- Allow batch video processing and dashboard summaries.

Limitations

- Works best with fixed-angle CCTV footage.
- Detection quality depends on video resolution and lighting.
- Real-time deployment would require further optimization for streaming input.

The Project

SMA

T

TMC

Developed By

Amr El-Sherif **Omar Alaa** **Shahd Alaa**

SMART TMC – Vehicle Detection and Reporting System

الفكرة العامة

SMART TMC هو مشروع بيحلل فيديوهات مرور (كاميرات شوارع) علشان:

- يعدّي العربيات اللي بتظهر في الفيديو.
- يصورهم وهم بيعدّوا.
- يسجّل الوقت اللي عدّوا فيه.
- وفي الآخر يطلّعلك ملف Excel فيه:
  - صورة كل عربية
  - توقيتها
  - نوعها (عربية؟ أتوبيس؟ نقل؟)

المشروع بيشتغل على 3 مراحل

المرحلة 1: قراءة الفيديو وتحضير الإعدادات

- الفيديو بيتحمّل من Google Drive.
- أول فريم بيتاخد ويتحلل باستخدام OCR:
  - لو فيه وقت أو كود موقع ظاهر في الفيديو، بيتقري تلقائي.
  - لو OCR معرفش يقرأ، المستخدم بيدخّلهم يدوي.

بعد كده، المستخدم بيحدد خطوط العد:

- فين الخط اللي هنتابع عليه (Y).
- وحدود خط اليسار وخط اليمين (X).

المرحلة 2: عدّ العربيات

- هنا بيبدأ البرنامج يحلل الفيديو:
  - بس مهم جدًا تعرف ان : YOLO مش بيشتغل هنا خالص!
  - البرنامج بيعتمد على حاجة خفيفة اسمها background subtraction:
  - يعني بيشوف إيه اللي بيتحرّك في الفريم (العربيات).
  - بيحدد شكل كل جسم بيتحرّك، ويشوف حجمه.
  - لو حجمه كبير كفاية → بنعتبره عربية.
  - بعد كده بيحسب "مركز الجسم"، ويشوف:
  - هل هو عدى من على الخط اللي إحنا محددينه؟
  - لو آه:
  - بيصور العربية (يقص الصورة من الفريم).
  - ويسجل وقتها.
  - ويحط كل ده في car_data.
  - ده بيخلي الكود سريع جدًا لأننا ما بنستخدمش YOLO على كل فريم.

كمان أثناء العد بيحسب سرعات العربيه عشان نتجنب مشكله التكرار او ان لو عربيه سريعه جدا متتلقطش:
لو العربية ماشية بسرعة جدًا ← بيقلل الوقت المطلوب بين كل عربية والتانية عشان متتحسبش مرتين
لو ماشية ببطء ← بيستنى وقت أكتر قبل ما يعتبر دي عربية جديدة

المرحلة 3: التصنيف والتقارير

بعد ما الفيديو يخلص:

- البرنامج بياخد الصور اللي سجلها للعربيات.
- بيشغل عليها YOLOv8 مرّة واحدة بس علشان يعرف نوع كل عربية:
  - Car / Bus / Truck / Motorcycle...
- بعدها، بيقسم النتائج كل 15 دقيقة.
- ولكل جهة (يمين/شمال)، بيعمل ملف Excel فيه:
  - التوقيت اللي العربية عدّت فيه.
  - صورة للعربية.
  - نوعها (من YOLO).
  - وفي الآخر، بيعمل صفحة "Summary" فيها عدد العربيات حسب النوع.

الأداء

- كل ساعة فيديو بتاخد حوالي 17 دقيقة معالجة على Google Colab (GPU).
- ده بيشمل كل حاجة: كشف، تتبع، قص صور، تصنيف، وتصدير التقارير.