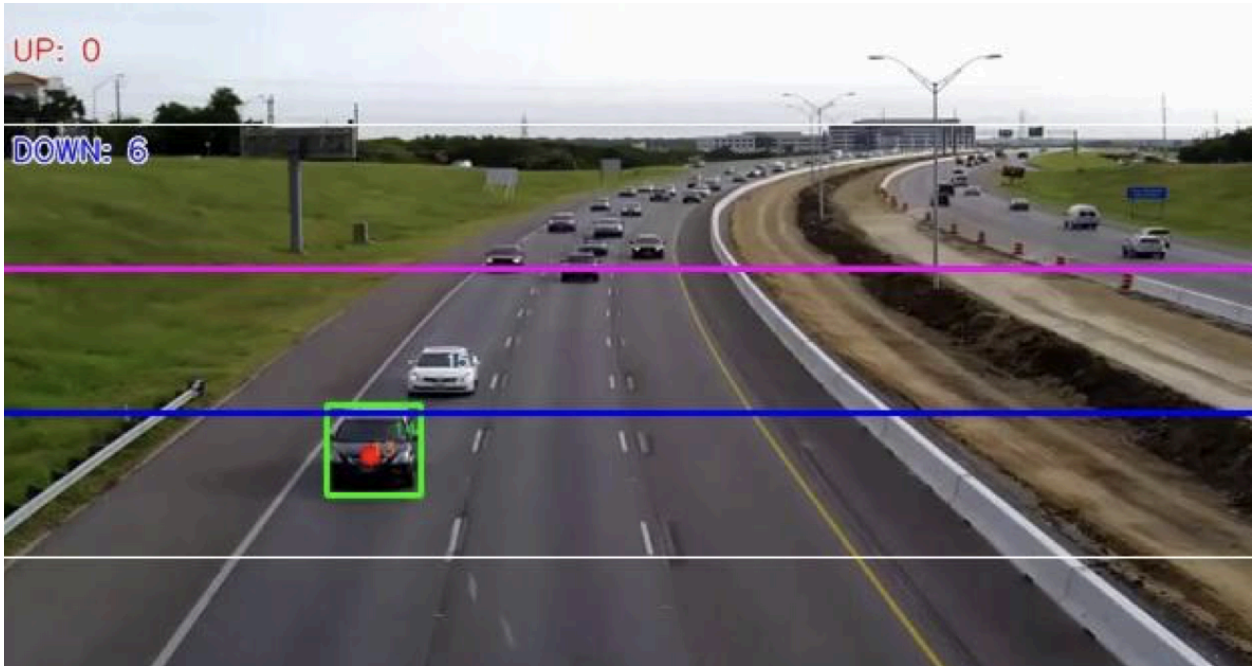
Documentation for Vehicle Detection and Counting



Introduction:

The "Vehicle Detection and Counting" web application is a project that aims to count the number of vehicles passing through a video stream. The project utilizes computer vision techniques and machine learning algorithms to identify and count the vehicles. The application is built using the OpenCV library and Flask framework, making it easy to use and accessible from a web browser. In this documentation, we will go through the project architecture, its functionality, and how to use the application.

Architecture:

The project is built using Python programming language, utilizing OpenCV library and Flask framework. The web application consists of two main components:

Object Detection Algorithm:

1. This component performs the detection of the vehicles in the video stream. It uses OpenCV's built-in object detection algorithm, which employs Haar-like features and the Viola-Jones algorithm to detect vehicles in an image. The object detection algorithm identifies the position of each vehicle in the frame and draws a bounding box around it.

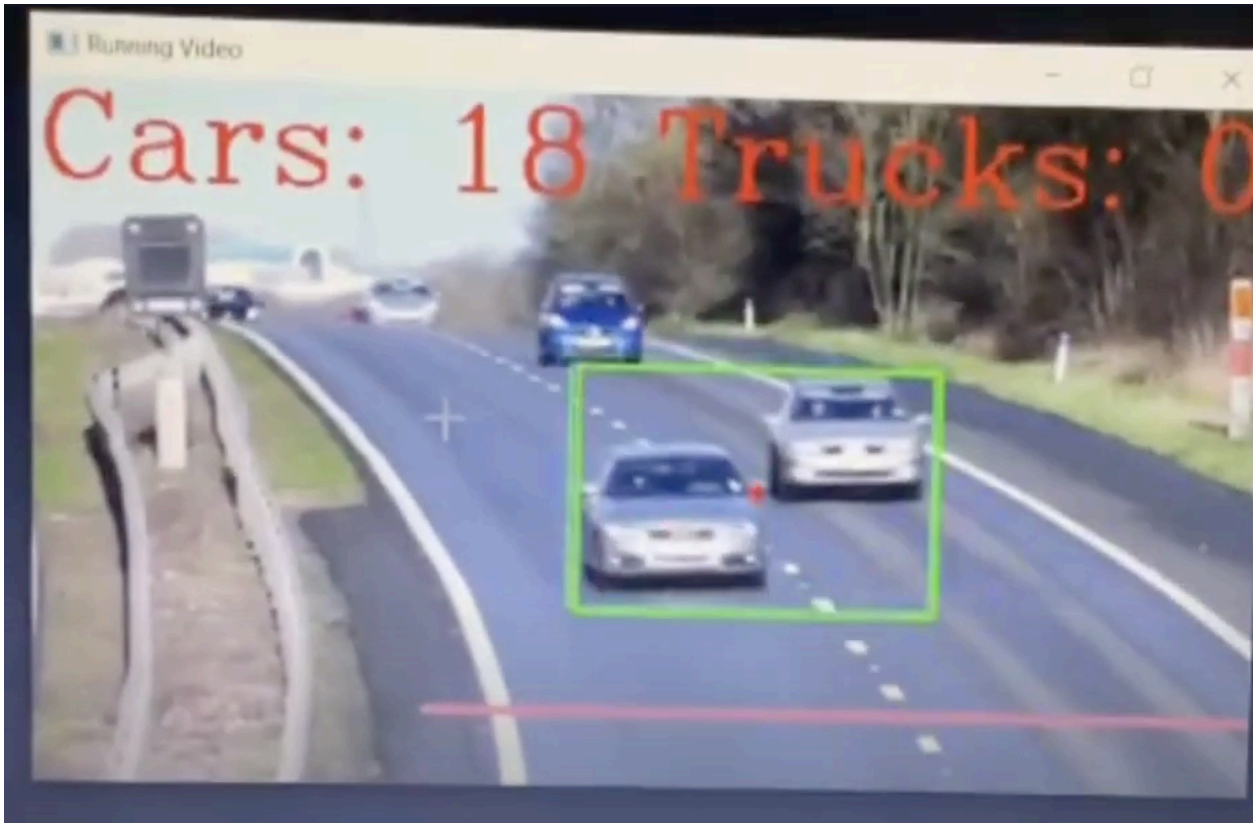
Web Application:

2. This component provides a user interface for accessing the project. It is built using the Flask framework, which provides a lightweight web server and a template engine. The web application allows the user to upload a video file, view the processed video with bounding boxes around the detected vehicles, and view the count of vehicles passing through the video.

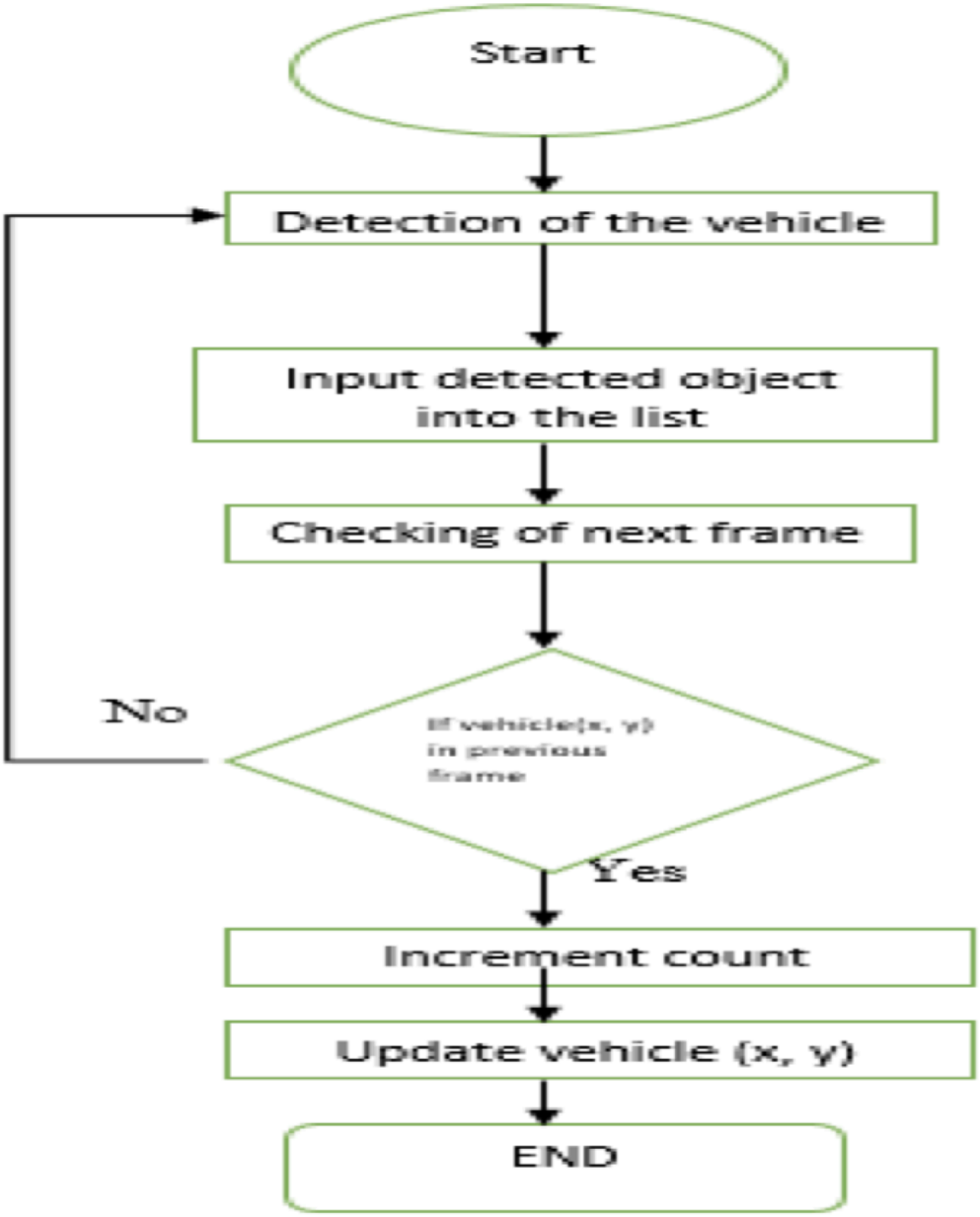
This web app is developed in Flask that uses OpenCV to detect and count vehicles in a video. Here is a brief explanation of the files:

- `app.ipynb`: This is a Jupyter Notebook file that contains the initial implementation of the vehicle detection algorithm. It is likely that the code in this file was later moved to `app.py` for deployment to a web server.
- `app.py`: This is the main file of the Flask application that serves as the web server. It contains the Flask routes that define the behavior of the web application. This file likely includes the vehicle detection code from `app.ipynb`, but adapted to work in a Flask context.
- `index.html`: This file is the main HTML file that defines the structure and layout of the web page. It is likely that this file includes an HTML5 video element that references `video.mp4`.
- `video.mp4`: This is the input video file that the vehicle detection algorithm will analyze. It is possible that this file is stored locally on the web server or on a cloud storage service.

Overall, the web application seems to work by allowing a user to upload a video file through the web page. Once the video is uploaded, the Flask server reads the video file and passes it to the OpenCV vehicle detection algorithm. The algorithm analyzes each frame of the video to detect and count the number of vehicles that appear in the video. Finally, the web application displays the output of the algorithm on the web page for the user to see.



Functionality:



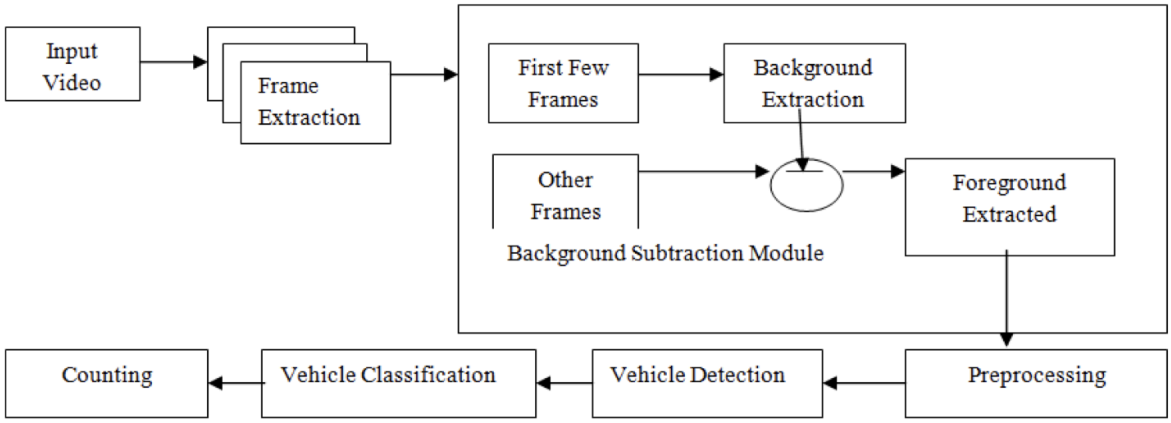
Flowchart of vehicle detection

The "Vehicle Detection and Counting" web application works as follows:

1. The user accesses the web application through a web browser.
2. The user uploads a video file through the web application.

3. The video file is passed to the Object Detection Algorithm component.
4. The Object Detection Algorithm component detects the vehicles in the video and draws bounding boxes around them.
5. The processed video with bounding boxes is displayed to the user through the web application.
6. The number of vehicles passing through the video is counted and displayed to the user through the web application.

The code creates a web application that can detect and count the number of vehicles passing through a video, using OpenCV and Flask. Here is a brief explanation of the code:



Block Diagram of Vehicle Detection and Counting

- First, the necessary libraries and the video capture object are imported.
- A Flask application is created and the initial car and truck counts are set to zero.
- A function is defined to find the center of the moving object and check if it crosses the count line position. If it does, the function increments the car or truck counter accordingly.
- The video frames are read in a while loop and processed one by one.
- A background subtractor is used to detect moving objects in each frame.
- Morphological dilation and closing are performed on the binary image obtained from background subtraction to smooth the image and remove noise.
- The contours of the objects in the binary image are found using the findContours() function.
- If the contour satisfies certain conditions (e.g., it is large enough to be considered a vehicle), a rectangle is drawn around it.

- The center of the rectangle is added to a list, and if any point in the list crosses the count line position, the car or truck counter is incremented accordingly.
- Finally, an HTML template is created to display the car and truck counts on a web page.

How to use the application:

To use the "Vehicle Detection and Counting" web application, follow these steps:

1. Open the web browser and access the web application URL.
2. Click on the "Upload Video" button.
3. Select the video file you want to process and click "Open."
4. Click on the "Process Video" button to start the video processing.
5. Wait for the video processing to complete. The processed video with bounding boxes will be displayed.
6. The total number of vehicles passing through the video will be displayed.

Conclusion:

The "Vehicle Detection and Counting" web application is a project that provides an efficient and easy-to-use solution for counting the number of vehicles passing through a video stream. The project utilizes computer vision techniques and machine learning algorithms to detect and count the vehicles. The web application is built using the Flask framework, making it easy to use and accessible from a web browser. With this project, the user can easily count the number of vehicles in a video stream and get accurate results.

REF:

1. https://pyimagesearch.com/2019/12/02/opencv-vehicle-detection-tracking-and-speed-estimation/
2. https://techvidvan.com/tutorials/opencv-vehicle-detection-classification-counting/
3. http://www.arresearchpublication.com/images/shortpdf/286a.pdf
4. https://core.ac.uk/download/pdf/83533829.pdf
5. https://www.jetir.org/papers/JETIRFM06016.pdf