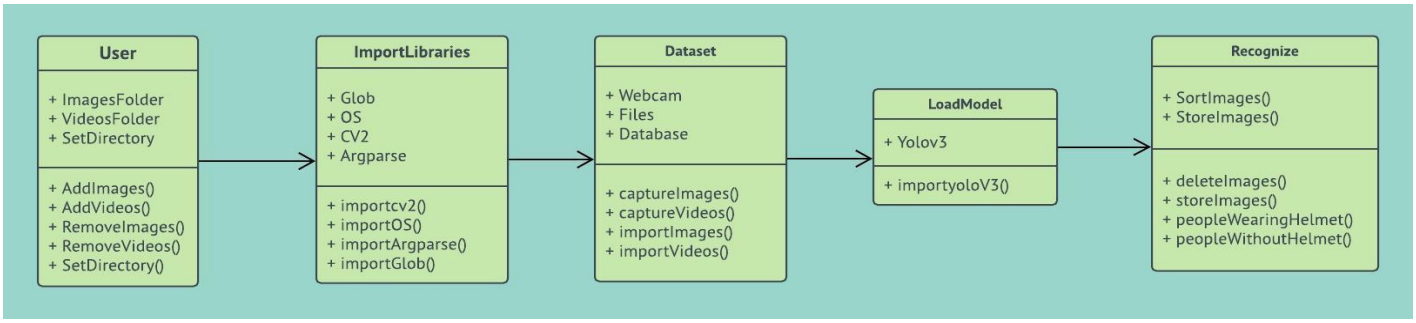
CHAPTER - 6

LOW LEVEL DESIGN

6.1. Class Diagram:



6.1.1. Helmet Detection

6.1.1.1. Class USER:

In this scenario, Traffic policemen are the users.

6.1.1.2. Data members of USER:

ImagesFolder/VideosFolder: They have access to the videos/images that are captured during traffic.

AddImages/AddVideos: The Traffic Police can feed a particular city/junction images to the helmet detection model.

RemoveImages/RemoveVideos: The Traffic Police can remove particular images/videos from the helmet detection model.

6.1.1.3. Class Import Libraries:

In this class, a set of required libraries and modules are being imported.

6.1.1.4. Data members and Methods of USER:

Glob: This module is used to retrieve files/path names matching a specified pattern.

OS: This module is used in Python that will provide functionalities for interaction with the operating system. OS module is one of the standard utility modules in python.

Argparse: This python module is used to generate help and issues messages and errors too whenever the user provides invalid arguments in the program.

(Below modulus has to be added in line violation not in helmet detection expt cv2)

Tkinter: This module provides a fast and easy way to create a Graphic-user interface for python applications.

OpenCV: This module mainly focuses on image/video processing and analysis , it helps to solve computer vision problems.

ImageIo: This module provides a way to read/write the wide range of image data.

PIL: This module adds additional functionality like opening, manipulating, and saving many different image file formats.

6.1.1.5. Class Load Model:

Yolov3: This algorithm is helpful in creating layers of convolutional and stack together as a whole.

6.1.1.6. Class Recognize:

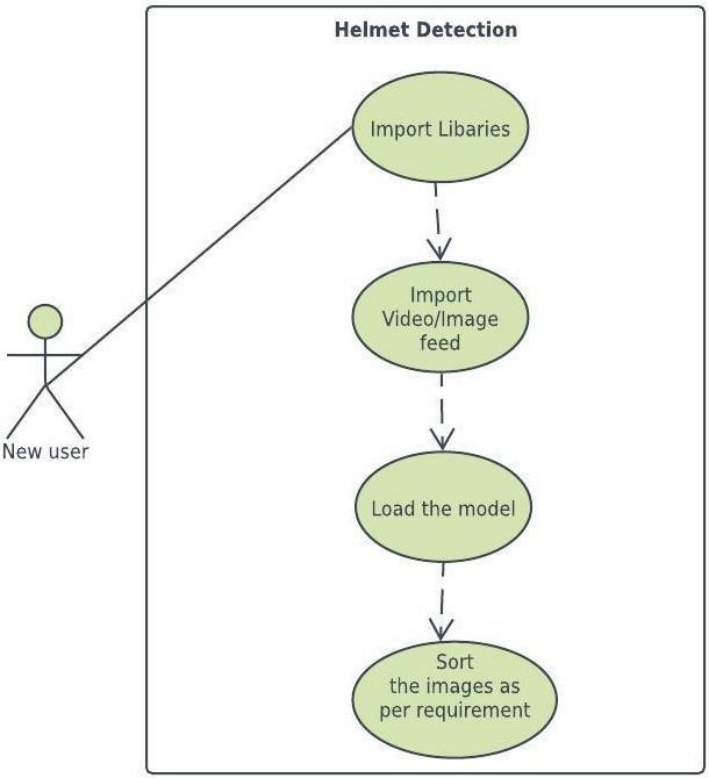
Description: In this class, images are sorted and stored.

6.1.1.7. Methods of Class Recognize:

peopleWearingHelmet(): Images of people wearing helmets are detected.

peopleWithoutHelmet(): Images of people who are not wearing helmets are detected.

6.2. Use Case Diagram:



Considering Helmet Detection, the use case diagram consists of the following use case items,

i) Importing the Libraries: All the necessary modules and libraries are to be imported for the model to work.

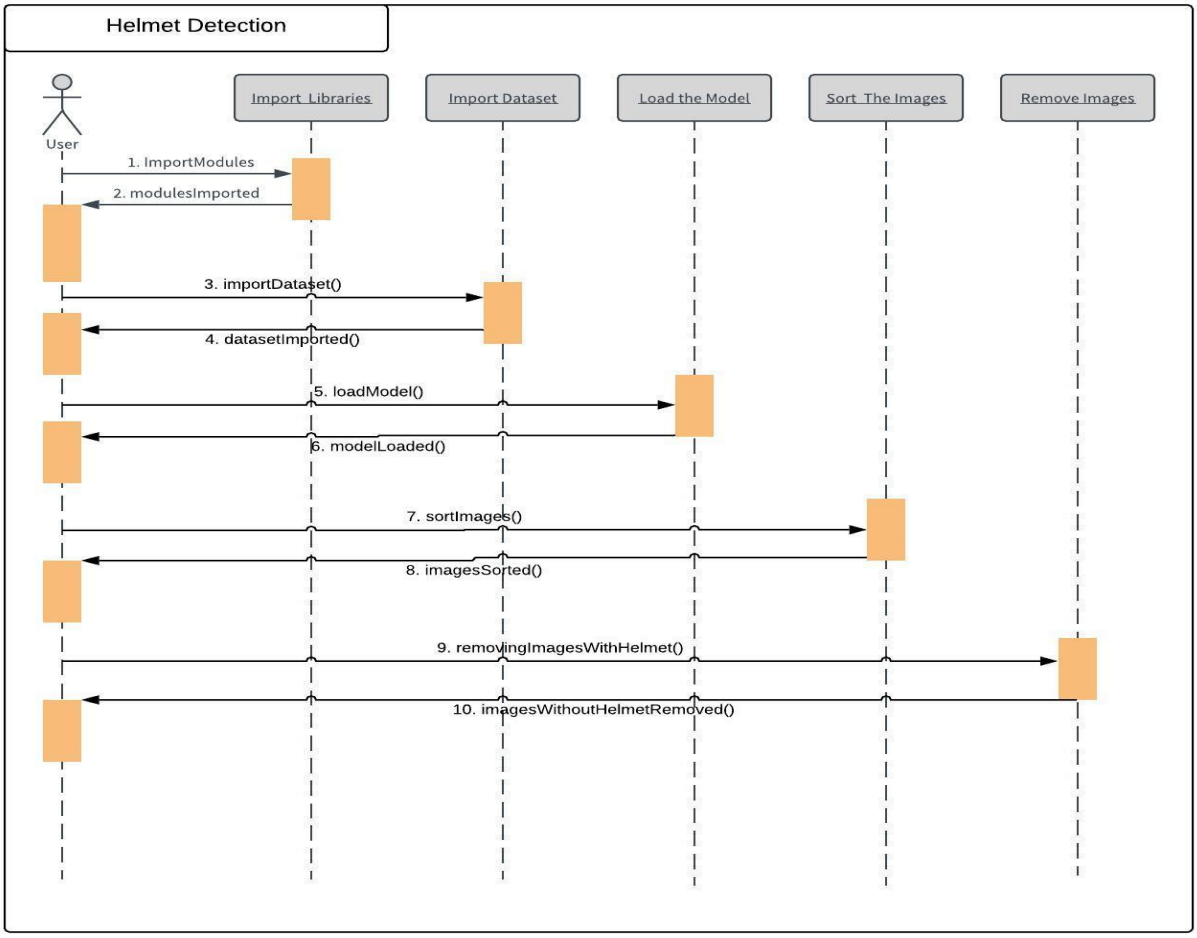
ii) Importing the Video/Image Feed: All the recorded videos/images of the traffic are taken as input by importing it.

iii) Loading the model: Helmet detection model is being loaded to detect the helmets.

iv) Sort the images: We sort the images as per the requirement so that the images of people with helmets are removed and without helmets are being detected.

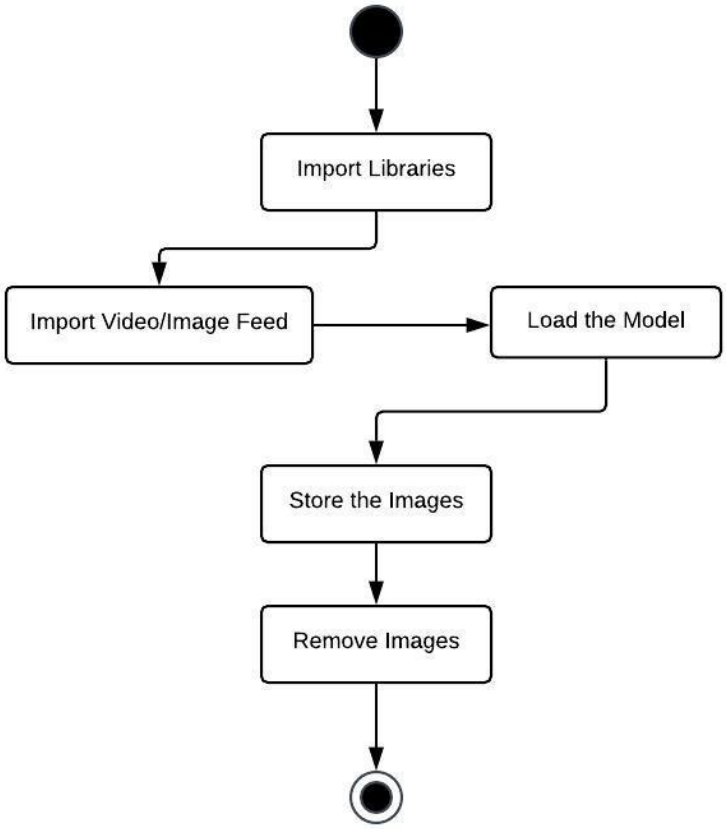
6.3. Sequence Diagram:

Below is the sequence diagram for Helmet Detection that consists of all the classes that are described in above paragraph

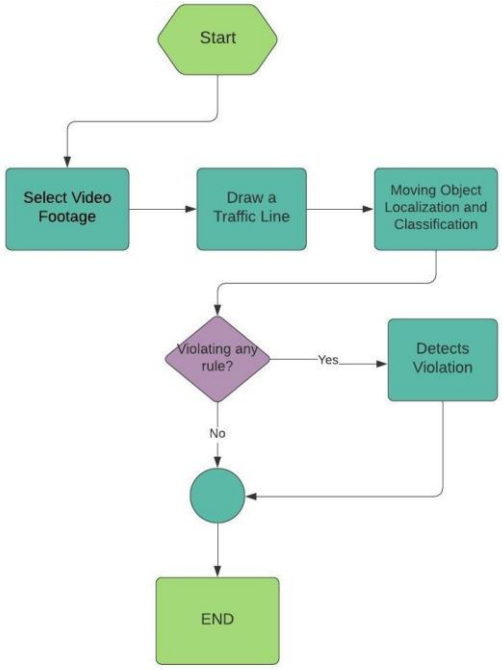


6.4. Activity Diagram:

Below is the activity diagram for helmet detection and line violation.



6.5. Activity/Flow chart for Line Violation:

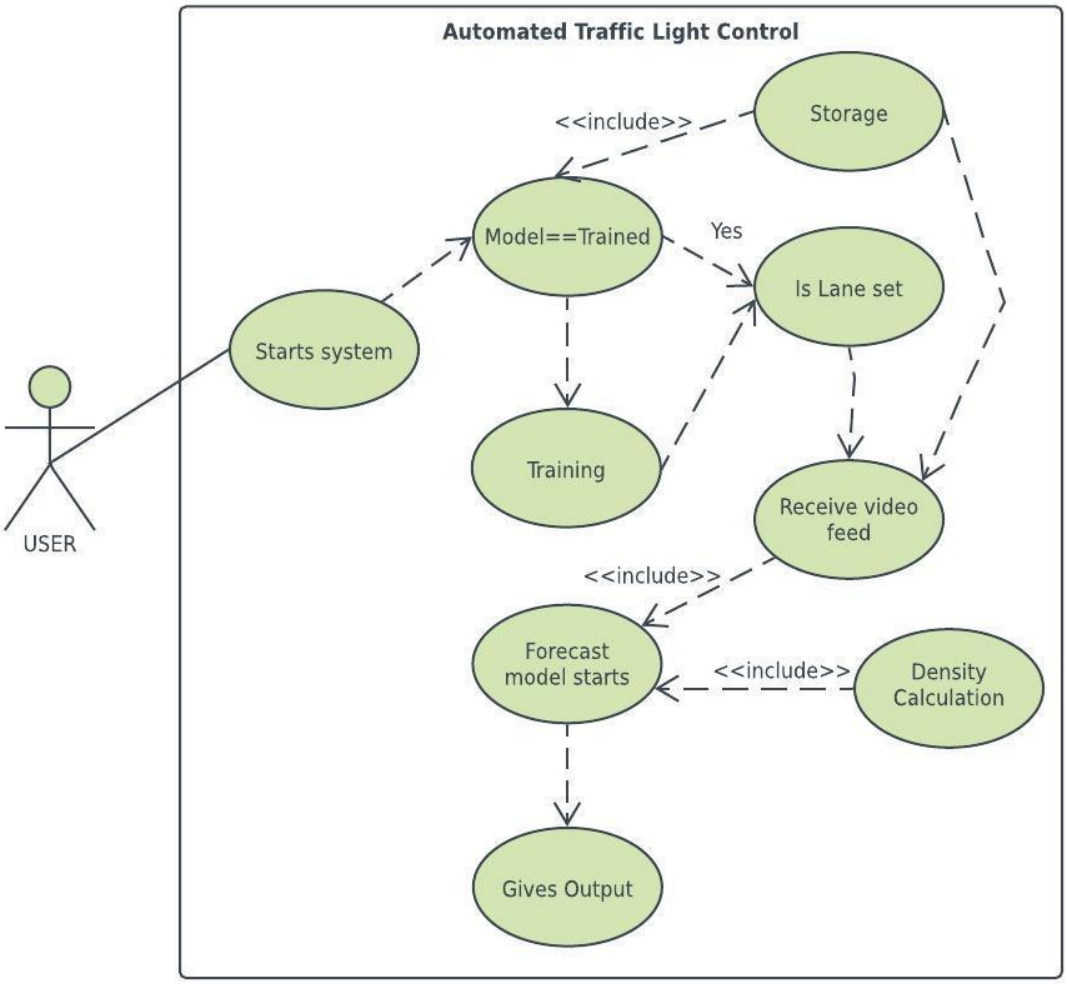


The following are the steps involved in line violation,

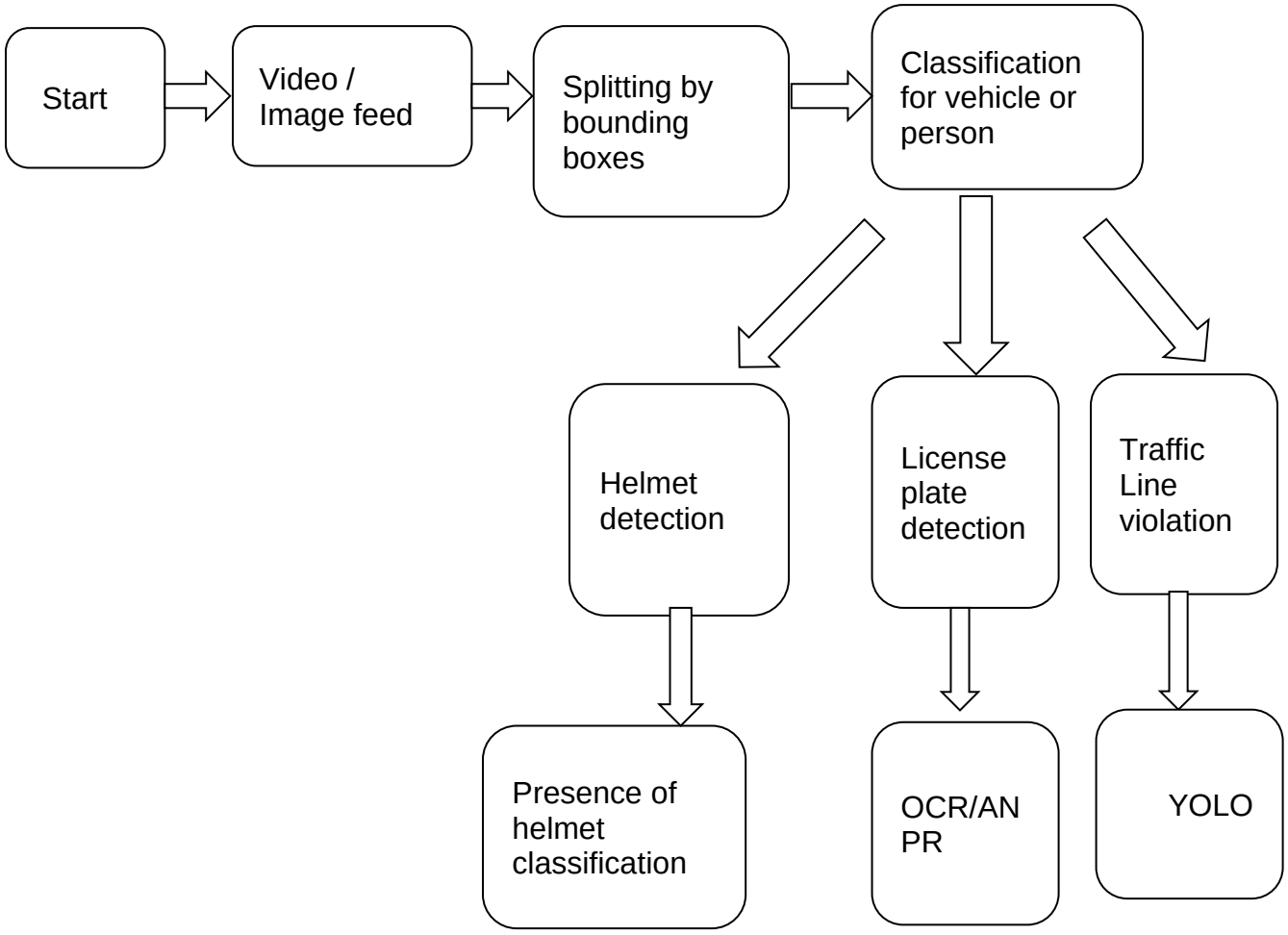
1.The user needs to select the video footage
2.the user needs to draw a traffic violation line at the signal junction.
3.Now, the model will try to detect the moving objects and also classify the vehicles which cross the line drawn in the previous step.
4.Vehicles which cross the line during red light is considered a violation.
5.Now, the pictures of these violated vehicles are stored and further sent to license plate detection.

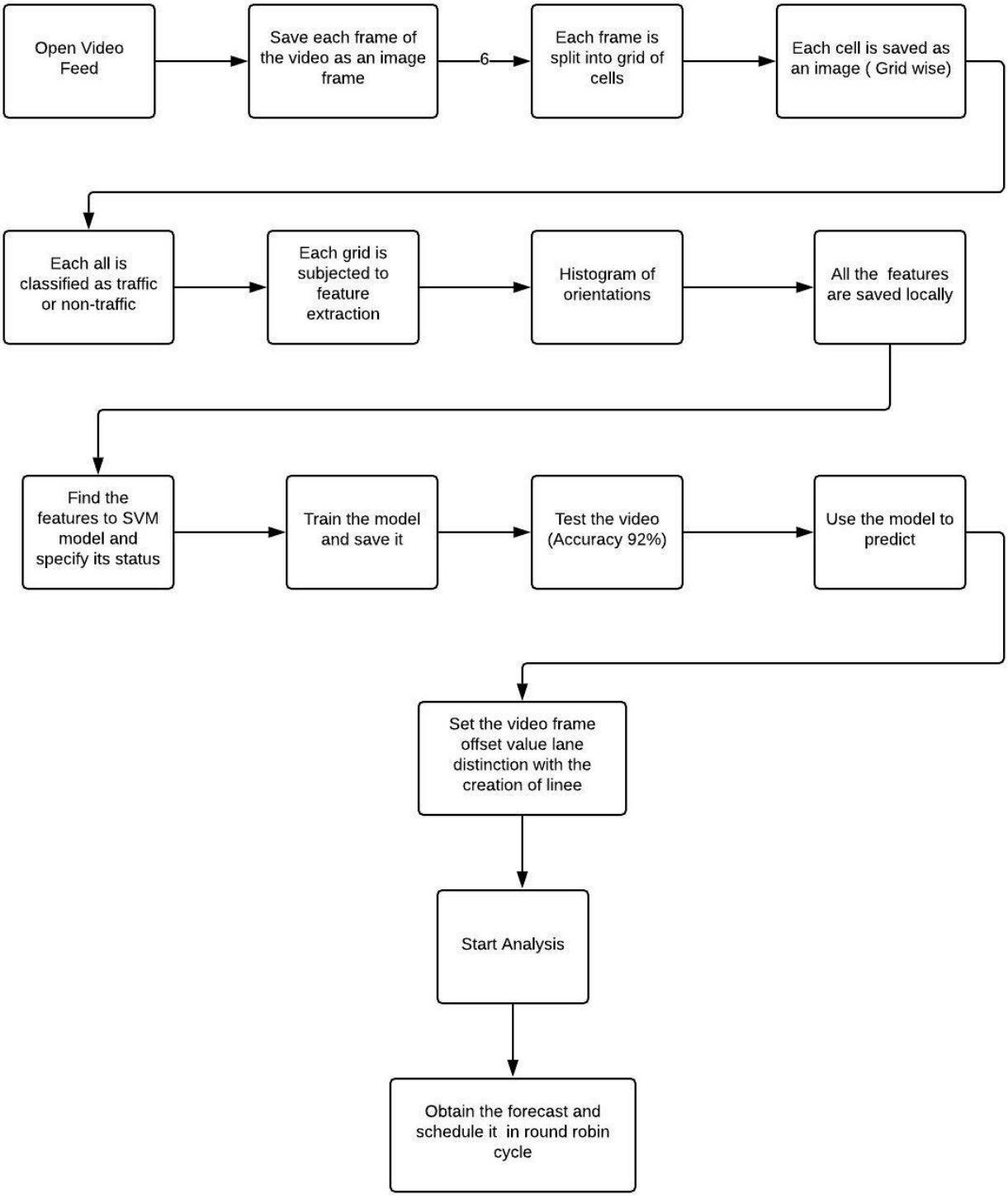
6.6. Use Case Diagram for Automated Traffic Light Control:

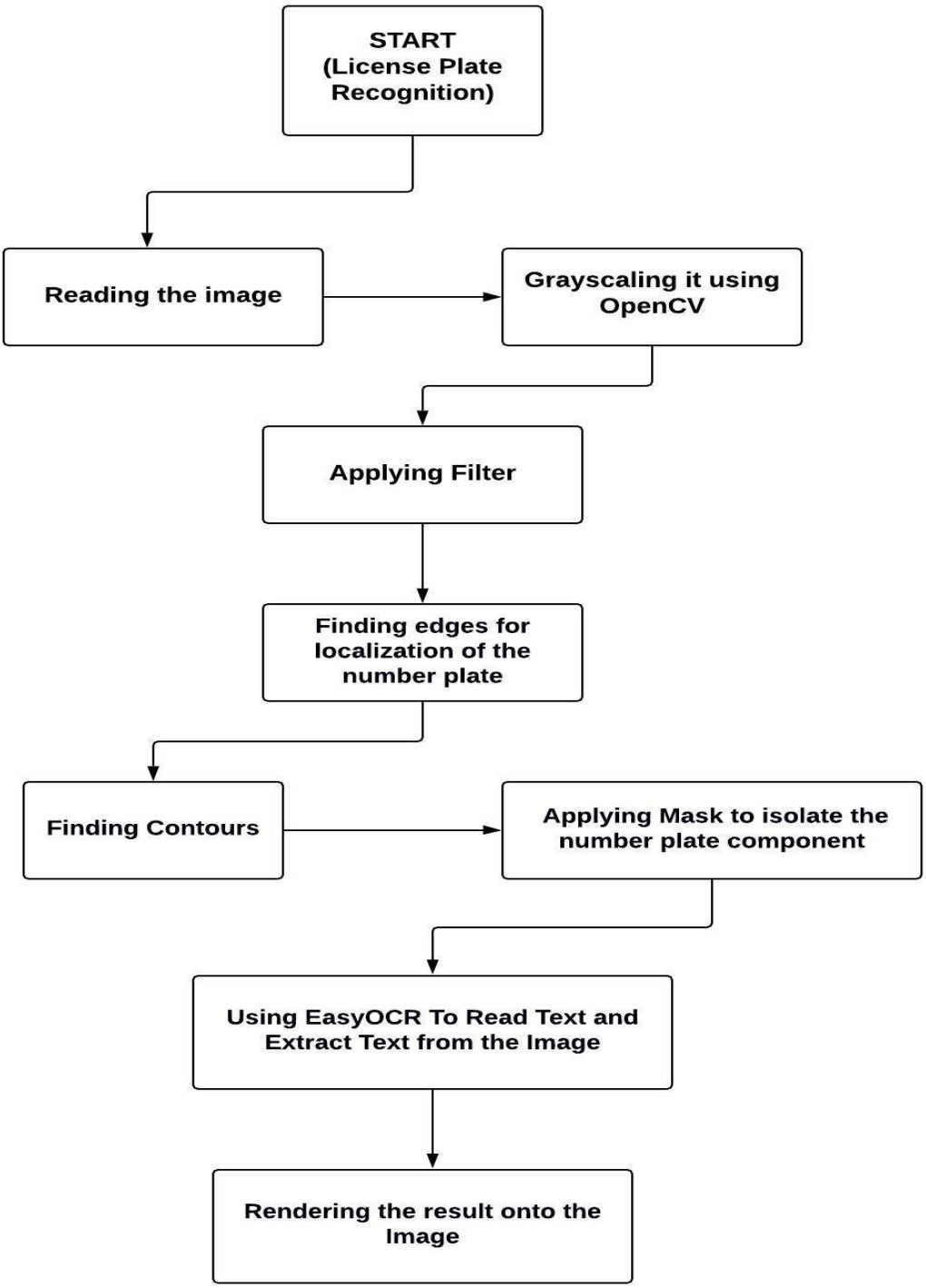
User as in the traffic policemen starts the system. This model is being trained to provide accurate results. Video feed is sent as input which again will be divided into image frames and used as an input to the model. We calculate the density and then forecast the model to output the desired results.



Flowchart for a complete traffic violation detection system:




Open Video Feed
Save each frame of the video as an image frame
6
Each frame is split into grid of cells
Each cell is saved as an image (Grid wise)
Each all is classified as traffic or non-traffic
Each grid is subjected to feature extraction
Histogram of orientations
All the features are saved locally
Find the features to SVM model and specify its status
Train the model and save it
Test the video (Accuracy 92%)
Use the model to predict
Set the video frame offset value lane distinction with the creation of linee
Start Analysis
Obtain the forecast and schedule it in round robin cycle


START
(License Plate
Recognition)
Reading the image
Grayscaling it using
OpenCV
Applying Filter
Finding edges for
localization of the
number plate
Finding Contours
Applying Mask to isolate the
number plate component
Using EasyOCR To Read Text and
Extract Text from the Image
Rendering the result onto the
Image