CGC ASSIGNMENT – 4

Name: RAHUL VARMA

Roll No: S20200010212

Vehicle Counting IN and OUT:

Task of this assignment:

In the previous assignment CLAHE is added to improve the quality of the frames.

- ➔ CLAHE

and, when the image captures many rectangles the send back the rectangle to the processing unit and the reject the smaller rectangles.

Now, We Should count the number of vehicles coming IN and number of vehicles going OUT.

We Should take a reference line where we can say the vehicle crossed the line.

And the count is increased when the vehicle is crossed the line in the left side and right side.

More Detailed Explanation in the end of pdf.

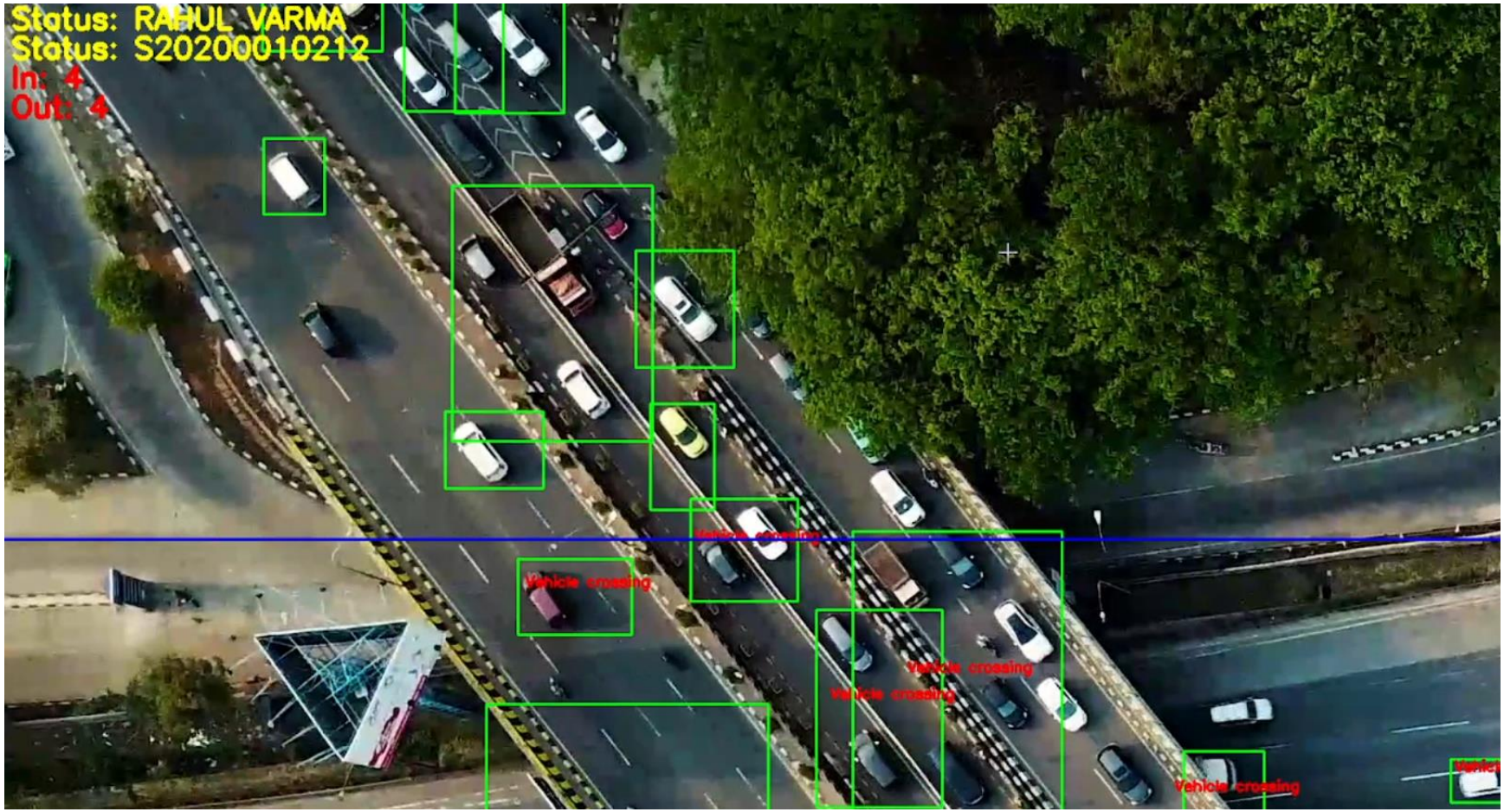
Before Counting the Lanes:

Object Moving is detected in Green Color Rectangle.

And the vehicle which comes towards the reference line it is shown in red color.

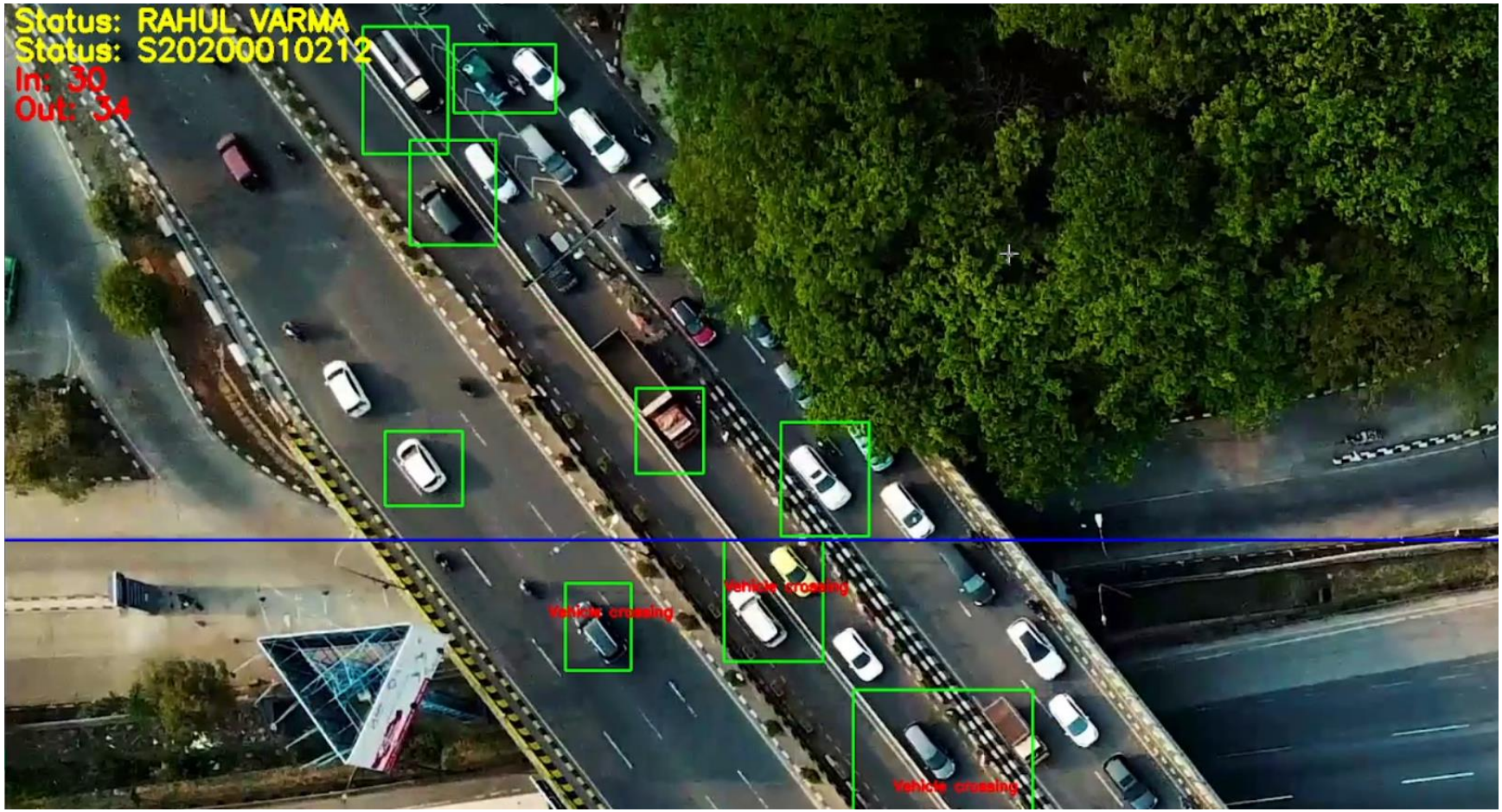
IN Count and OUT Count Shown:

Four Vehicles crossed IN and OUT:



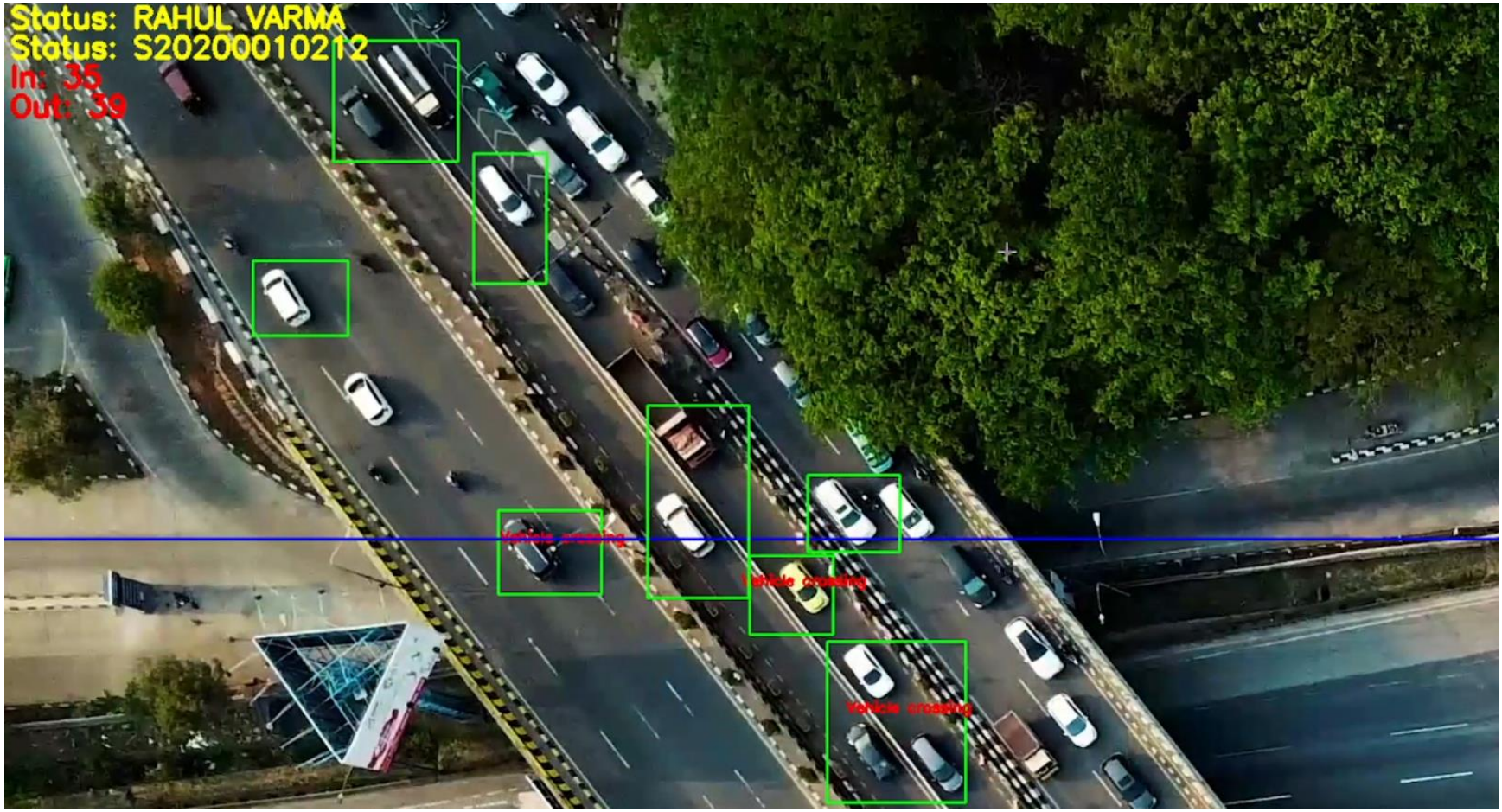
2nd Scenario:

IN -> 30 vehicles, Out -> 34 Vehicles.



3rd Scenario:

IN Count -> 35 and OUT Count -> 39.



Code:

```python
import cv2
cap = cv2.VideoCapture('video.mp4')
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
count_in = 0
count_out = 0
height, width, _ = cap.read()[1].shape
line_y = int(height / 4)
left_boundary = int(width / 3)
right_boundary = int(2 * width / 3)
while True:
    ret, frame1 = cap.read()
    gray1 = cv2.cvtColor(frame1, cv2.COLOR_BGR2GRAY)
    gray1 = clahe.apply(gray1)
    ret, frame2 = cap.read()
    gray2 = cv2.cvtColor(frame2, cv2.COLOR_BGR2GRAY)
    gray2 = clahe.apply(gray2)
    diff = cv2.absdiff(gray1, gray2)
    thresh = cv2.threshold(diff, 25, 255, cv2.THRESH_BINARY)[1]
    dilated = cv2.dilate(thresh, None, iterations=2)
    cnts, _ = cv2.findContours(dilated, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    for c in cnts:
        if cv2.contourArea(c) < 2500:
            continue
        (x, y, w, h) = cv2.boundingRect(c)
        cv2.rectangle(frame1, (x, y), (x + w, y + h), (0, 255, 0), 2)
        center = (int((x+x+w)/2),int((y+y+h)/2))
        if center[0] < left_boundary:
            count_in += 1
        elif center[0] > right_boundary:
            count_out += 1
        cv2.putText(frame1, "In: {}".format(str(count_in // 100)), (10, 90), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 4)
        cv2.putText(frame1, "Out: {}".format(str(count_out // 100)), (10, 120), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 4)
        cv2.putText(frame1, "Status: {}".format('RAHUL VARMA'), (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 255), 3)
        cv2.putText(frame1, "Status: {}".format('S20200010212'), (10, 60), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 255), 3)
        if center[1] > line_y:
            cv2.putText(frame1, "Vehicle crossing", (center[0] - 50, center[1] - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)
            if center[1] > line_y+10:
                count_crossed = count_in if center[0] < left_boundary else count_out
                count_crossed += 1
                if center[0] < left_boundary:
                    count_in = count_crossed
                else:
                    count_out = count_crossed
                cv2.line(frame1, (0, line_y+10), (width, line_y+10), (255, 0, 0), 2)
    cv2.imshow('frame', frame1)
    if cv2.waitKey(25) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

```python
import cv2
cap = cv2.VideoCapture('video.mp4')
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
count_in = 0
count_out = 0
height, width, _ = cap.read()[1].shape
line_y = int(height / 4)
left_boundary = int(width / 3)
right_boundary = int(2 * width / 3)
while True:
    ret, frame1 = cap.read()
    gray1 = cv2.cvtColor(frame1, cv2.COLOR_BGR2GRAY)
    gray1 = clahe.apply(gray1)
    ret, frame2 = cap.read()
    gray2 = cv2.cvtColor(frame2, cv2.COLOR_BGR2GRAY)
    gray2 = clahe.apply(gray2)
    diff = cv2.absdiff(gray1, gray2)
    thresh = cv2.threshold(diff, 25, 255, cv2.THRESH_BINARY)[1]
    dilated = cv2.dilate(thresh, None, iterations=2)
    cnts, _ = cv2.findContours(dilated, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    for c in cnts:
        if cv2.contourArea(c) < 2500:
            continue
        (x, y, w, h) = cv2.boundingRect(c)
        cv2.rectangle(frame1, (x, y), (x + w, y + h), (0, 255, 0), 2)
        center = (int((x+x+w)/2),int((y+y+h)/2))
        if center[0] < left_boundary:
            count_in += 1
        elif center[0] > right_boundary:
            count_out += 1
```

```
        cv2.putText(frame1, "In: {}".format(str(count_in // 100)), (10, 90), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 4)
        cv2.putText(frame1, "Out: {}".format(str(count_out // 100)), (10, 120), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,
0, 255), 4)
        cv2.putText(frame1, "Status: {}".format('RAHUL VARMA'), (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255,
255), 3)
        cv2.putText(frame1, "Status: {}".format('S20200010212'), (10, 60), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255,
255), 3)
        if center[1] > line_y:
            cv2.putText(frame1, "Vehicle crossing", (center[0] - 50, center[1] - 10), cv2.FONT_HERSHEY_SIMPLEX,
0.5, (0, 0, 255), 2)
            if center[1] > line_y+10:
                count_crossed = count_in if center[0] < left_boundary else count_out
                count_crossed += 1
                if center[0] < left_boundary:
                    count_in = count_crossed
                else:
                    count_out = count_crossed
                cv2.line(frame1, (0, line_y+10), (width, line_y+10), (255, 0, 0), 2)
    cv2.imshow('frame', frame1)
    if cv2.waitKey(25) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

Detailed Explanation:

- The system is designed to count the number of vehicles passing through two vertical boundaries in the frame, one on the left and one on the right, and determine whether they are moving in or out of the frame. The system uses motion detection to identify the presence of a vehicle and uses contour detection to track its movement across the boundary lines.
- Apply Contrast Limited Adaptive Histogram Equalization (CLAHE) to the grayscale frames. This helps to enhance the contrast of the grayscale frames and improve the accuracy of the motion detection algorithm.
- Next, the system sets up the boundary lines by defining the left and right boundaries as one-third and two-thirds of the width of the frame, respectively. It also defines a horizontal line at one-fourth of the height of the frame, which is used to detect when a vehicle crosses the boundary lines.
- The system then enters a loop that reads frames from the video file one at a time using the cap.read() function. For each frame, it applies CLAHE to the grayscale image, calculates the difference between the current frame and the previous frame using cv2.absdiff(), thresholds the difference image using cv2.threshold(), dilates the thresholded image using cv2.dilate(), and finds contours in the dilated image using cv2.findContours().
- The system then checks whether the center of the bounding rectangle is on the left or right half of the frame and updates the vehicle count for the corresponding direction. It also checks whether the vehicle has crossed the horizontal line by keeping track of a Boolean variable called crossed. If the vehicle has not already crossed the line, the system updates the count and sets crossed to True.
- Overall, the system is an effective implementation of a vehicle counting system using OpenCV library and can be used for various applications, such as traffic analysis, parking lot management, and surveillance.