**Report of Hardware Mini-project**

**1.Estimated System Performance**

The recognition region has a width of 20 pixels. If a car drives too slow so that it enters the region at frame 0 and stays in the region at frame 1, it will be counted twice. The other possibility is that if a car drives too fast so that it enters the region at frame 0 and stays in the region at frame 1, it will be counted twice.

Furthermore, currently, the system can only detect the cars at night because the algorithm detects cars by capturing the headlight of car. We are not sure how the system works during daytimes because we are running out of time.

The starting point of this project is from the ground: knowing nothing about the python, raspberry pi and ssh operating system, etc.

The raspberry pi encounter serious internet connection problems. Therefore the pi didn’t get required python libraries until two days before the submission. In addition to that, the lack of python language skills makes the code comparatively lengthy and not optimized.

The primary reason to choose python cv2 library is because of the search of internet. Web informations to the “car detection and counting” is highly composed by the opencv library. Through a short time I understand the library that I used in my code.

**2. Hardware System Deficiencies:**

The system can be improved largely in hardware part:

1. The pi zero w has comparatively fragile input ports.
2. The very low frame rate cause low resolution of video.
3. When the pi connect to the monitor, all the screen process use pi’s own CPU.
4. The pi frequently goes very hot in the process of use.

**3. Basic Algorithms:**

The code that we used are two separated python files. The motdet.py is the video recording file. This file is from Prof. Hirsch’s github senior design page:

<https://github.com/BostonUniversitySeniorDesign/hardware-project-2018/wiki>

The car\_count.py file is the file used for vehicle detection and counting. The basic algorithm for this project can be called the “dialate special feature point”:

Delta frame difference----> Dilate feature points (car light) -------> find contour for dilated feature -----> if the feature is in the region of detection ----->detection reached and car number +1

**Content of library cv2. :**

cv2.cvtColor: conversion of BGR to gray.

cv2.GaussianBlur:works for Gaussian Filtering

cv2.rectangle: draw a straight bounding rectangle

cv2.absdiff: difference between the pixel values of the current frame and the frame right before. The change of pixel value make the system to detect object in motion, i.e. cars in this project.

cv2.threshold: minimum value of difference between the pixel values of the current frame and the frame right before for the system to recognize it.

cv2.dilate:increasing the size of car light to make detection easier

cv2.findContours: find the contour of a car

(\_, cnts, \_) = cv2.findContours(thresh.copy(), cv2.RETR\_CCOMP, cv2.CHAIN\_APPROX\_SIMPLE)

cv2.contourArea(c): find contour area of of detected object C

cv2.boundingRect(c): smallest rectangle that can fit in the detected object C

cv2.circle: draw circle

cv2.putText:add text

camera.read(): read mp4 file

cv2.imshow: show windows

cv2.waitKey: wait for key press

camera.release(): close camera

cv2.destroyAllWindows(): close all windows

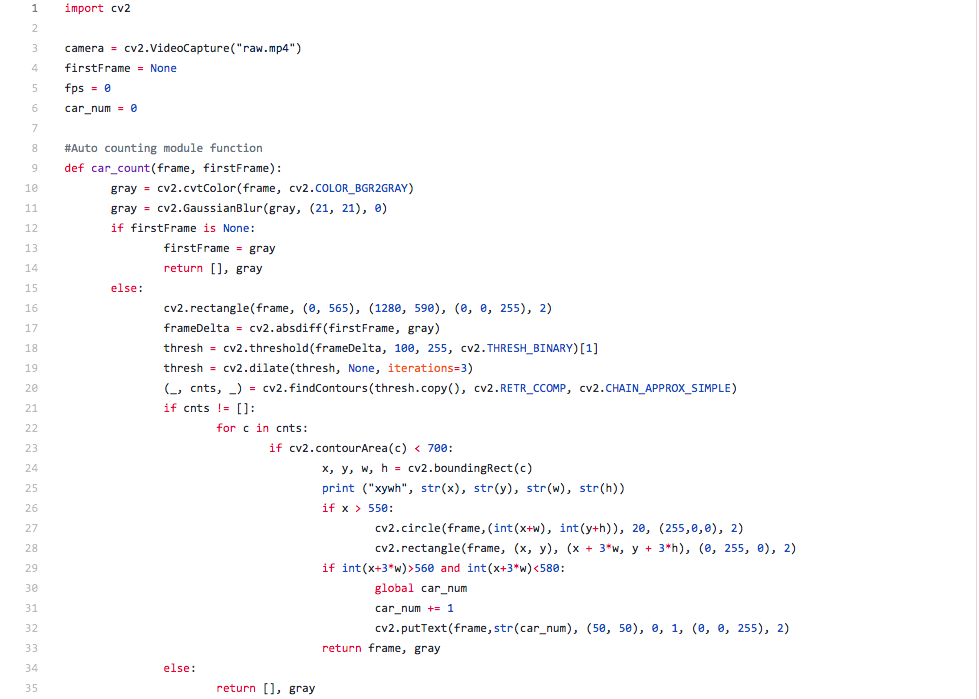
The three cv library functions that labeled in red above represents the first three steps in the process shown above. The primary reason for the chosen algorithms is because: motion cause frame difference, and frame difference can be detected by digital difference.

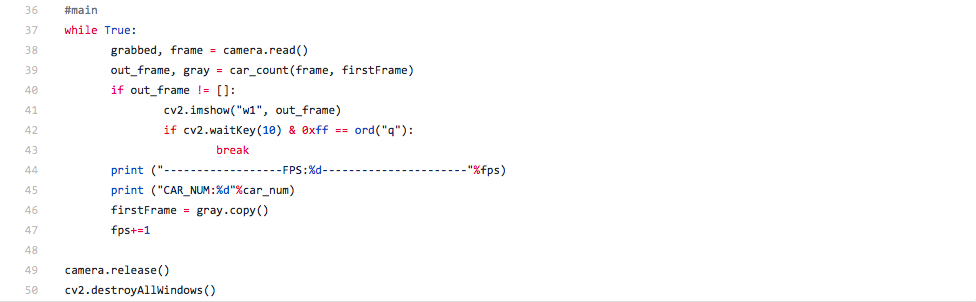
Advantage:

1. easier structure.
2. Fast enough.

Disadvantage:

1. can only capturing light.
2. More time is needed to combine two files together that make one file directly reach all goals.





**4. User defined interfaces:**

We used one user-defined function: def car\_count(frame, firstFrame) in our system.

We find it advantageous because, instead of writing the same function every single time, we just use one line and reuse the same function for multiple times.