EMBEDDED MACHINE VISION AND INTELLIGENT AUTOMATION

EXERCISE 5

NIKHIL DIVEKAR

Question1.

This article presents the story of Stanley robot which won DARPA Grand challenge in 2005. Grand challenge refers to the competition of the robots to achieve the high-speed desert driving without human intervention. The robot's software system depended predominantly on machine learning and probabilistic behavior. The goal of the challenge was to develop autonomous robot which can traverse through unrehearsed terrain. The first competition had 142-mile long course in the Mojave Desert within 10 hours. In first competition. 15 teams raced but none of the teams covered even 5% of the course. Stanley won this challenge in 2005 while competing with 22 other teams with excellent timing of 6 hours 53 min and 52 seconds. The development of Stanley was much contributed by Stanford University and experts from Volkswagen, Mohr Davidow Ventures, Intel and other such entities. Stanley has 6 outfitted processing platforms by Intel and a suite of sensors and actuators. The most important challenges in design was to achieve high reliability, speed and precision. The design of Stanley inspired many of the self-driving automobile designs in the future. The venues of this challenge varied from high-quality graded dirt road to winding rocky mountain passes. The race course venue and mappings are hidden from teams until 2 hours for competition. When a faster robot overtakes the slower one, the slower robot was paused by DARPA officials as if it were static obstacle.

Stanford racing team was subdivided into 4 groups. Vehicle group oversaw all the modification to the core vehicle. Software team was responsible for software development for navigation to safety. Testing group tests the robot in all the possible conditions. While communications group handled all the media activities and fund-raising.

Stanley is based on diesel-powered Volkswagen Touareg which has four-wheel drive, variable-height air suspension, automatic electronic locking differentials. Individual wheel speed and steering angle can be sensed automatically and sent to the computer system. Stanley's roof has two 24 GHz RADAR sensors. The GPS module of this car is responsible for estimation of location and velocity of the vehicle relative to the external coordinate system. It also has the E-stop feature which allows wireless link for the chase vehicle to stop vehicle in case of emergency. The operating system being used is Linux due to its networking and time-sharing capabilities. In the race, Stanley executed race software on three machines, fourth was used for logging purposes. Stanley's custom user interface allows user to safely operate the robot as conventional passenger. Driver can gain access to the manual control any time.

Number of principles were involved with Stanley's design. Some of them were control and data pipeline, state management, reliability, and development support. Race software can be broken down into six layers. Sensor interface layer responsible for collection and time stamping of data from various sensors, Perception layer maps sensor data to the internal modules, Control layer controls the unit like steering, throttle, brakes, etc. Vehicle interface layer is more of like user's

application layer. Global services provide many pre-requisite and smaller services to the vehicle. In Stanley, the vehicle state comprises a total of 15 variables.

For the obstacle detection, Lasers are used as the basis in Stanley's design. It is equipped with 5 single-scan laser range finders for precise object detection. Stanley uses discriminative learning algorithm for locally optimizing parameters in a way that maximizes the accuracy of the terrain analysis. Stanley uses color camera to find drivable surfaces at ranges exceeding that of the laser analysis. The vision module classifies image into drivable and non-drivable region. Adaptive approach is necessary for this classification as the quality of road depends on the number of factors. Stanley finds drivable surfaces by projecting drivable area from laser analysis into the camera image. An adaptive computer vision uses the pixels in this image as training example for drivable surfaces. The learned gaussian is used to analyze the image. The remaining pixels are then classified using learned mixture of Gaussian. Pixels whose RGB values are nearer to one or more of learned Gaussians are classified as drivable. When no drivable area is detected within 40 feet, robot is slowed down, at which point laser navigation comes into picture.

One of the most important component of Stanley is a method that makes Stanley stay in the center of the road. Another feature is ability to choose the velocity after taking terrain on which it is travelling into consideration. Stanley also has the component called online path planning. The goal is to complete the course as fast as possible following the minimum distance and while avoiding the obstacle. The path finder is basically a search algorithm that minimizes linear combination of continuous cost functions. The first dimension describes the amount of lateral offset to be added to the current trajectory. The second parameter in the path search allows the planner to control the urgency of obstacle avoidance. All of the above features depict how machine vision was used in Stanley.

Real-time control is very important for higher reliability and higher precision. It includes velocity control and steering control. Lots of preparation and testing was done with Stanley before participating in the competition. Stanley was the only vehicle to clear all the 50 gates in every run in the national qualification event. While DARPA's grand challenge was success for Stanley. Stanley was still unable to navigate in the traffic. So there is still much scope in the improvement.

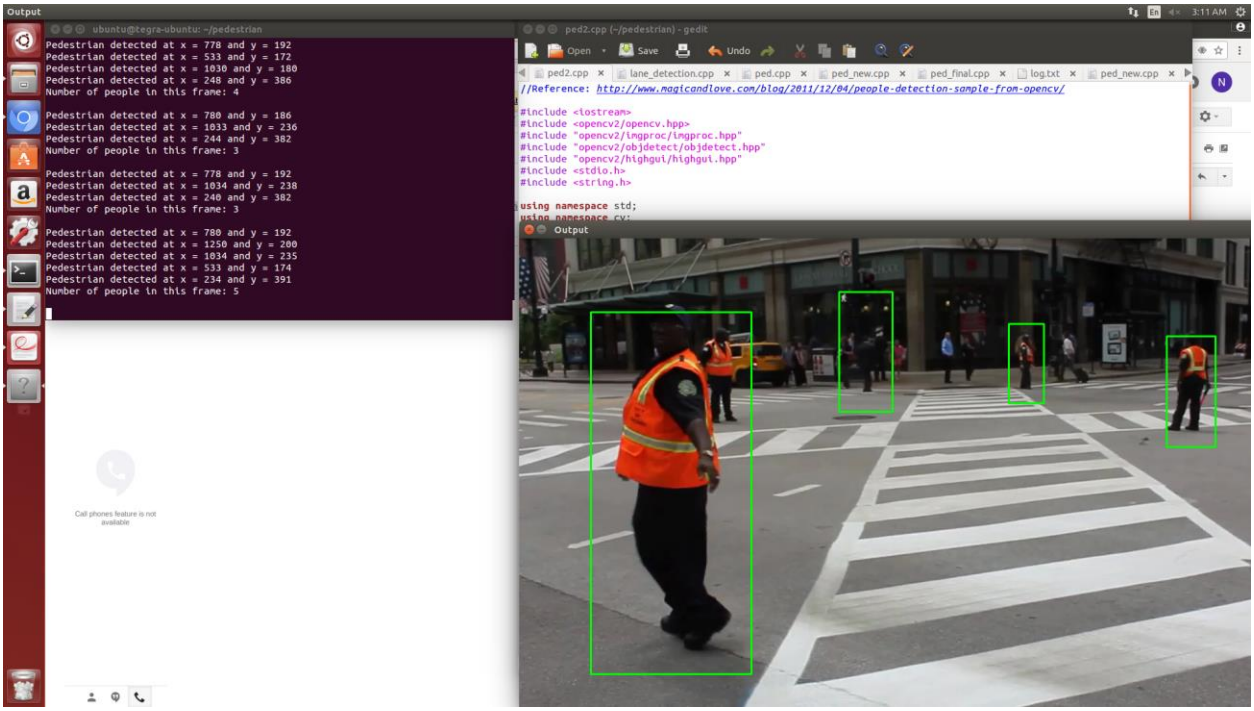
Question 2.

I attempted the pedestrian detection part of the Question 2. I was able to detect the pedestrians in the video and mark them with the rectangle. I used the HOG descriptor code for the same. Once the pedestrians were detected, I recorded the number of pedestrians and co-ordinates of the pedestrians detected and logged this data to some txt file.

Make command

```
ubuntu@tegra-ubuntu:~/pedestrian$ make
g++ -O0 -g   -c ped2.cpp
g++  -O0 -g   -o ped2 ped2.o `pkg-config --libs opencv` -L/usr/lib -lopencv_core
 -lopencv_flann -lopencv_video
ubuntu@tegra-ubuntu:~/pedestrian$
```

Screenshot of the pedestrian detection code running and video



Log file:

```
Frame number: 1
Pedestrian detected at x = 1030 and y = 180
Frame number: 1
Pedestrian detected at x = 250 and y = 381
Number of people in the frame: 2

Frame number: 2
Pedestrian detected at x = 778 and y = 192
Frame number: 2
Pedestrian detected at x = 533 and y = 172
Frame number: 2
Pedestrian detected at x = 1030 and y = 180
Frame number: 2
Pedestrian detected at x = 248 and y = 386
Number of people in the frame: 4

Frame number: 3
Pedestrian detected at x = 780 and y = 186
Frame number: 3
Pedestrian detected at x = 1033 and y = 236
Frame number: 3
Pedestrian detected at x = 244 and y = 382
Number of people in the frame: 3

Frame number: 4
Pedestrian detected at x = 778 and y = 192
Frame number: 4
Pedestrian detected at x = 1034 and y = 238
Frame number: 4
Pedestrian detected at x = 240 and y = 382
Number of people in the frame: 3
```

Link to the output video:

https://drive.google.com/file/d/1FTUxBci40BrpOOCpuWKSh_eF9757yD9j/view

Question 3.

We are currently deciding among 3 ideas so far this exercise, we are submitting 3 different proposals, titled Option 1, Option 2, Option 3.

Option 1:

Self-driving car:

Idea and Implementation:

- Self-driving car is the vehicle with an autopilot system and capable of driving from one point to another without human intervention.
- We are planning to implement the self-driving car system from the image processing point of view. The major modules which would be included in the project are lane detection, vehicle and pedestrian detection along with road sign detections.

- We plan to control the steering module and perform the speed control according to the lanes detected and other vehicles and road signs.
- Lane detection will be helpful in deciding the movement of steering wheel in right or left to keep the vehicle in the same lane.
- Vehicles in the front are detected and speed will be reduced if there is vehicle stopped right in the front or if the vehicle ahead of us is decelerating. Vehicle will also stop if stop sign or red traffic signal is detected.
- We also plan to integrate this module with Google maps APIs so that self-driving car should be able to go from one point to other following the directions provided by the google maps.
- Also, lane detection module is used to understand whether it is valid to overtake the vehicle right in front of us. Lane departure warning system can also be implemented for safety purposes.
- Rear camera is also integrated to confirm the safety while lane changing or turning onto the different road.

Relevance of selection:

- Many big company's like Google and Tesla are prominent leaders in self-driving car. We plan to implement the self-driving car module prototype with different modules. These modules will collectively achieve target of automatic steering and speed control, reacting according to the road signs and finding route from source to destination.
- OpenCV can be used to detect the lane (hough transform), vehicles (haar cascade) and pedestrians (HOG descriptors), Also machine learning can be integrated to identify the road signs and also for affiliation with Google map APIs.

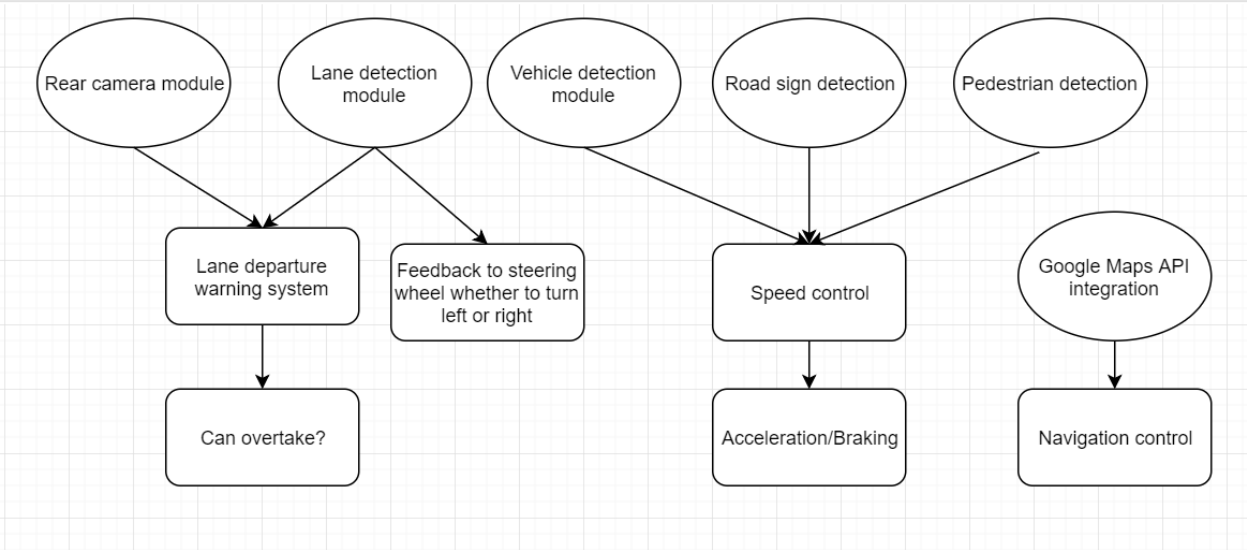
Challenge:

- The biggest challenge with this problem statement should be the integration of all the above-mentioned modules.
- The efficiency and speed should be kept optimal even after integrating the different modules. We are planning to use multi-threading to minimize the speed concern.
- Also, exact identification of road signs and integration with google map APIs provides challenge to some extent.

Rationale of selection:

1) Minimization of human intervention and thereby reducing the chances of human mistakes and thus fewer accidents.
2) Traffic management will be improved to a greater extent and thus decreased traffic congestion.
3) Lower fuel consumption due to minimization of traffic jams.
4) This will also minimize the incident of speeding since car will be programmed to follow the road signs and thus also maintain speed according to the limit.

Block diagram:



Option 2

Vehicle speed detection and license plate extraction using OpenCV:

Idea and Implementation:

- Speeding endangers everyone on the road: In 2016, speeding killed 10,111 people, accounting for more than a quarter (27%) of all traffic fatalities that year. Thus, the automatic system which can help keep check on speeding should be implemented.
- The basic idea of this project is the speed detection of the vehicles on the road and license plate extraction of the vehicle which is navigating above the speed limit.
- We are planning to use two cameras. One camera will be at certain height so that it should be able to capture the video of the traffic on the road. The main functionality of this camera is to detect the vehicles moving in one direction and track their speed.
- We will also capture the image of the vehicle and track the count of vehicles according to their size whether its light weight or heavy weight vehicle. Now, if certain vehicle is observed to go at the speed higher than the speed limit, then the second camera placed at certain distance from the first one and at the lower height will be triggered.
- According to the speed and distance between two cameras, second camera will properly capture the image of the speeding vehicle. Since this camera is at the lower altitude, it will be helpful in getting proper look at the number plate.
- Once the image is captured, we plan on marking the number plate of that vehicle as the region of interest and extract it from the image of the car. Further, image enhancement and sharpening will be used to detect the characters and numbers from the number plate.

- The ultimate target will be exact determination of the plate number of the vehicles and then logging the speeding vehicle's license plate number along with recorded speed and date and time to a certain log file.

Relevance of selection:

- The OpenCV can be utilized in vehicle speed tracking and pinpointing of speeding vehicles. Vehicle detection can be achieved by Haar cascade.
- Speed can be detected by following the line once vehicle is tracked. This implementation can be integrated with machine learning for exact determination of the number plates.

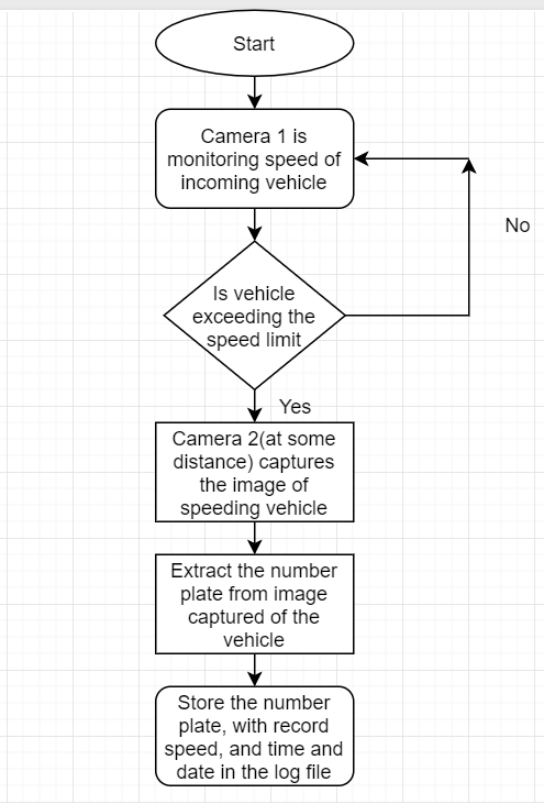
Challenge:

- The utmost challenge in the implementation is the synchronization of the two cameras. One which records the speed of all the vehicles and other one which captures the image of the speeding vehicle.
- Also, another challenge includes the exact determination of the number plate of the speeding car when zoomed-in.

Rationale of the selection:

1) This implementation helps to eliminate the efforts of police patrolling the major roads. It reduced the human labor and thereby reduces the probability of mistakes.
2) The implementation provides with very important details of speed tracking and catching the speeding vehicle which can be stored in database for future data analytics.
3) Further, the number plate can be tracked back to owner and speeding ticket can be directly issued online using certain database. (Future Scope).

Flow-chart:



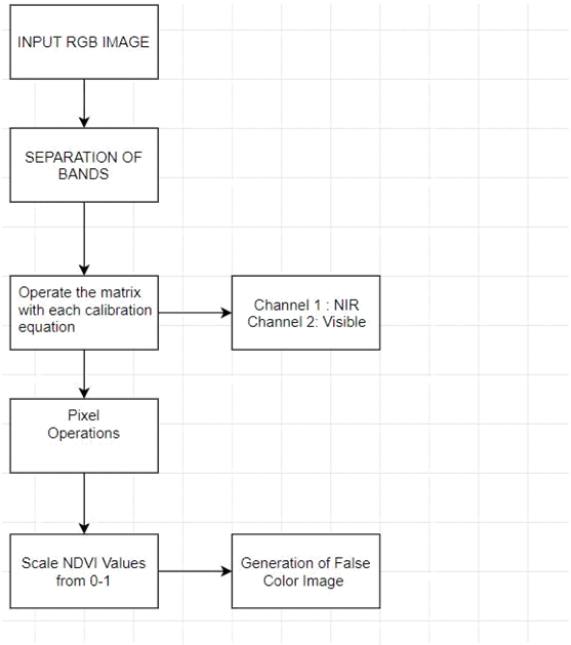
Option 3:

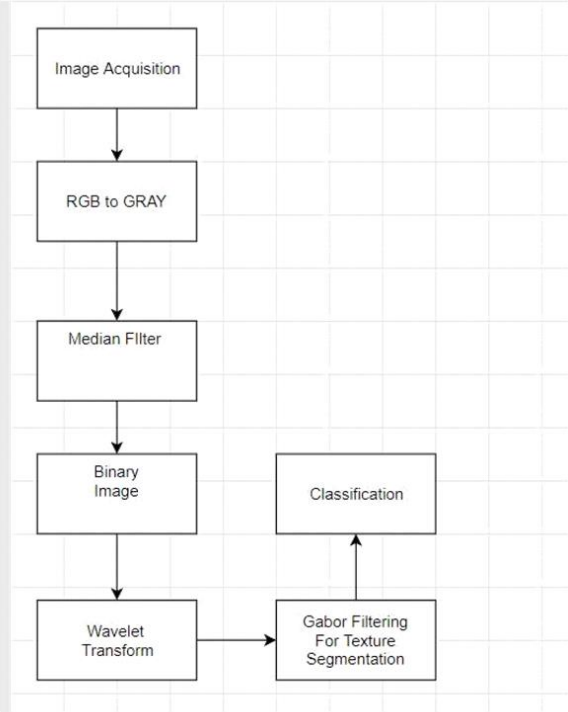
Precision Agriculture using Image processing

- Precision agriculture is the method where you analyze certain large-scale farm by dividing it into small parts to precisely find out the requirement of necessary factors like lime and other fertilizers and water based on co-ordinates of those areas.
- It's hard to inspect large-scale agricultural land manually and it's prone to human errors since we can't analyze these requirements precisely with manual inspection.
- Precision agriculture is the technique where you analyze satellite images of the farm and based on that you get to know about different factors like vegetation index, water clogging due to faults in irrigation system, presence of weed and pests present in different areas of your farm.
- Thus, using geographical information system (GIS) and GPS we can farmers to locate affected areas and take necessary action.
- Though these techniques were used since long time these are not precise since getting and processing images from satellites is time consuming and costly process. [1]

- Also, it’s not precise since it's taken from distance.
- Thus, to make this process efficient we would try to implement it using pictures taken by cameras mounted on farm monitoring UAVs
- Also, we will try to calculate get perpendicular vegetation index (PVI) instead of normalized difference vegetation index (NDVI) since it's more accurate measure of healthiness of the plant and which requires different image processing technique than what's traditionally used. [2]

Flowcharts:





Question 4.

Option 1:

Self-driving Car:

Minimum:

1) Lane detection, vehicle and pedestrian detection.
2) Road sign detection.
3) Speed control according to vehicle and road signs.

Target:

1) Steering wheel direction feedback according to the lane detection system.
2) Properly identifying the road signs and reacting appropriately according to identified signs.
3) Lane departure warning system.

Optimal:

1) Integrating google map APIs

2) Rear camera module to detect the vehicles behind and detect whether it's ok to change the lane.

Option 2:

Vehicle speed detection and license plate extraction using OpenCV

Minimum:

1) Keeping count of vehicles.
2) Tracking the speed of vehicles.

Target:

1) Capturing the image of the vehicle exceeding the speed limit.
2) Triggering the second camera to focus and capture number plate of the vehicle of interest.
3) Picking number plate region and extracting numbers and characters from number plate.

Optimal:

1) Use of character recognition and machine learning to exactly identify number plate.
2) Logging the number plate, speed of speeding vehicle along with date and time and store it in certain file or database.

Option 3:

Precision Agriculture using Image processing

Minimum:

1) Getting RGB images from satellite and using band separation and pixel manipulation techniques calculate approximate NDVI or use IR images and do the same to get accurate results. [1] [3] . Satellite options are AVHRR or Landsat 8.[5]

2) Feature extraction using NDVI to create false color composite image which would be spectral signature(geographical) of vegetation. [3]

Target:

1) Identify and locate water clogged areas using image segmentation and spectral analysis methods on resulting false color image.
2) Segregate different areas based on healthiness of the plants and identify their co-ordinates.
3) Differentiate between weed and plants using texture segmentation and classification based on training data-sets.

Optimal:

1) Image processing on images captured by drone instead of satellites which would increase the accuracy of the system, this would require stitching frames in the captured video and based on the path and speed of the drone and frame-rate of the camera to recreate the top view of the farm to do further image processing to get NDVI. (Drone Flying Site: Flatiron/Longmont Golf Course)

2) Write algorithm to calculate PVI instead of NDVI which would give more precise results.[2]

References:

1. https://www.agritechtomorrow.com/article/2018/01/ndvi-vs-false-ndvi--whats-better-for-analyzing-crop-health/10434
2. http://www.mdpi.com/2072-4292/2/3/673
3. https://ac.els-cdn.com/S2212017312006196/1-s2.0-S2212017312006196-main.pdf?_tid=90b17072-583d-4fbd-9a50-a61434a37da6&acdnat=1532834305_79e74a4c38bc4c0fa32af96c2a70f37f
4. http://www.microimages.com/sml/ParisScripts/PrecisionRemoteSensingSlides.pdf
5. http://proceedings.esri.com/library/userconf/proc17/papers/1981_658.pdf

6. http://www.magicandlove.com/blog/2011/12/04/people-detection-sample-from-opencv/