Atmiya University of Technology And Science

Rajkot – 360005

A Project Report On

"Computer Vision Road – Analysis Tool"

Submitted in the partial fulfilment of the requirements for the award of the Degree of

Bachelor of Science in Information Technology

Submitted By:

Drashty Ranpara	**(180802084)**
Unnati Pandya	**(180802070)**

Under the support and guidance of :

Dr. Ripal Ranpara

Assistant Professor,
Department of CS

Mr. Deepan Parikh

ML/AI Engineer,
nViso

Acknowledgement

Knowledge in itself is a continuous process. At this moment of our substantial enhancement, we rarely find words to express our gratitude towards those who were constantly involved with us.

The completion of any inter disciplinary project depends upon coordination, cooperation and combined efforts of several resources of knowledge, creativity, skill, energy and time. The work being accomplished now, we feel our sincerest urge to recall and knowledge through these lines, trying our best to give full credit wherever it deserves.

We would like to thank our project guide **Dr. Ripal D. Ranpara** who advised and gave us moral support through the duration of our project. Without their constant encouragement we could not have been able to achieve what we have.

It’s our good fortune that we had support and well wishes of many. We are thankful to all and those names which have been forgotten to acknowledge here but contributions have not gone unnoticed.

With Sincere Regards,
Drashty Ranpara (180802084)
Unnati Pandya (180802070)

Table of Contents

Project Profile

- **Project Name**: CV Road – Analysis Tool

- **Type of Project**: Computer Vision based SDK

- **Project Description**

- **Computer Vision based solution for better Road Analysis**

Project uses state-of-the-art computer vison technology to effectively and efficiently process vehicle dashboard videos and perform road analysis in order to detect and have a better understand of driving sense, road hazards which can cause a major impact for insurance companies in order to have an automated decisional platform.

- **Team Size**: 2
- **Front End**: PySimpleGui
- **Back End**: Python, OPEN CV, Yolo-V3
- **IDE Used**: PyCharm

Introduction to Tools

Introduction to Tools

➢ Front End Tools:

PYSIMPLEGUI:

- PySimpleGUI is a Python package that enables Python programmers of all levels to create GUIs. You specify your GUI window using a "layout" which contains widgets (they're called "Elements" in PySimpleGUI). Your layout is used to create a window using one of the 4 supported frameworks to display and interact with your window. Supported frameworks include tkinter, Qt, WxPython, or Remi. The term "wrapper" is sometimes used for these kinds of packages.
- Your PySimpleGUI code is simpler and shorter than writing directly using the underlying framework because PySimpleGUI implements much of the "boilerplate code" for you. Additionally, interfaces are simplified to require as little code as possible to get the desired result. Depending on the program and framework used, a PySimpleGUI program may require 1/2 to 1/10th amount of code to create an identical window using one of the frameworks directly.
- While the goal is to encapsulate/hide the specific objects and code used by the GUI framework you are running on top of, if needed you can access the frameworks' dependent widgets and windows directly. If a setting or feature is not yet exposed or accessible using the PySimpleGUI APIs, you are not walled off from the framework. You can expand capabilities without directly modifying the PySimpleGUI package itself.

➢ Back End Tools :

- **Python : Python** is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming.

- **Open CV** : **Open CV** is a cross-platform library using which we can develop real-time **computer vision applications**. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

- **YOLO** : **You only look once (YOLO)** is a state-of-the-art, real-time object detection system. A single neural network to the full image is applied. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. It has several advantages over classifier-based systems. It looks at the whole image at test time so its predictions are informed by global context in the image. It also makes predictions with a single network evaluation unlike systems like R-CNN which require thousands for a single image. This makes it extremely fast, more than 1000x faster than R-CNN and 100x faster than Fast R-CNN.

System Study

1. Existing System

- There are a lot of manual assessments involved nowadays in the present system in order to process a vehicle claim request for the insurance companies.
- In order to which insurance has enough chances to miss some of very important aspects.
- This results to un-reliable and inaccurate means of planning which occurs a significant loss to the client.
- This loses both efficiency and effectivity in order to achieve better claim decisions.

2. Proposed System

- CV Road-Analysis Tool helps insurance companies better understand the claim request with an automated approach which can help them provide much more accurate decisions over the claims.
- Clients can be justified with much better reasonability with respect to their claim requests.
- Enables better understanding of people for the better driving skills road hazards.

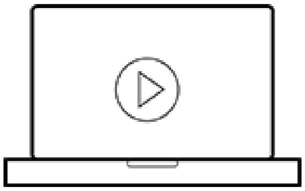
3. Scope of the Proposed System

➢ Insurance

- Project provides innovative solutions that help insurance companies provide accurate and trustworthy claim reasonability for their customer satisfaction and detect unrecognized responses for various risks. This allows insurance providers to take customer interaction to the next level.

4. Project Working Chronology ➢

Dashboard Experience



- Unique assessment solution enhancing the entire accident claim journey made available via a an engaging computer vision based application.

- Insurance companies clients can have a dashcam recording and provide is for assessment to the insurance agent of their loss to get their claims.

- A dashboard is present with the insurance agent where they can just input the video for better analysis of the situation.

- Various functions such as potential collision detection, collision detection, directional visibility, distance based filters, and distance based lines can be added for a better visualization of the scenario and to better recognize the situation.

- After selecting the appropriate visualization and hazard detecting functionalities, the dashboard shows the real time analysis at the original fps, going through processing each and every frame of the video.

➢ Real-time Personalized Processed Video

- Once entire video is analysed, an interactive processed video illustrating clients driving scenario profile is generated and available in real-time .

5. Aim and Objective of the proposed System

- Insurance companies can make a more accurate decision on claim requests.

- Enables understanding of people in order to better understand the driving sense and scenarios and how to best apply for the claim.

- Better understanding of the clients also helps the insurance companies to better communicate their own decision-making process.

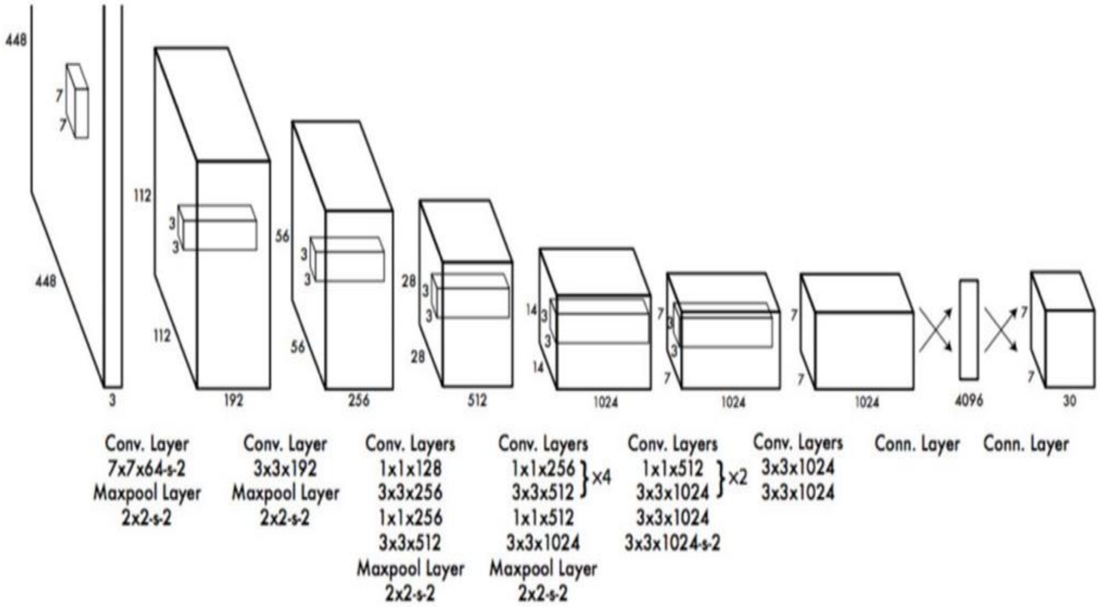
- Customers and regulators can have proof of insurance companies to act honestly, fairly and professionally in the client's best interest.

- Computer Vision powered processing enhances traditional risk profiling questionnaires providing more comprehensive understanding of the scenarios.

System Analysis

➢ System Modules

1. Yolo Architecture:



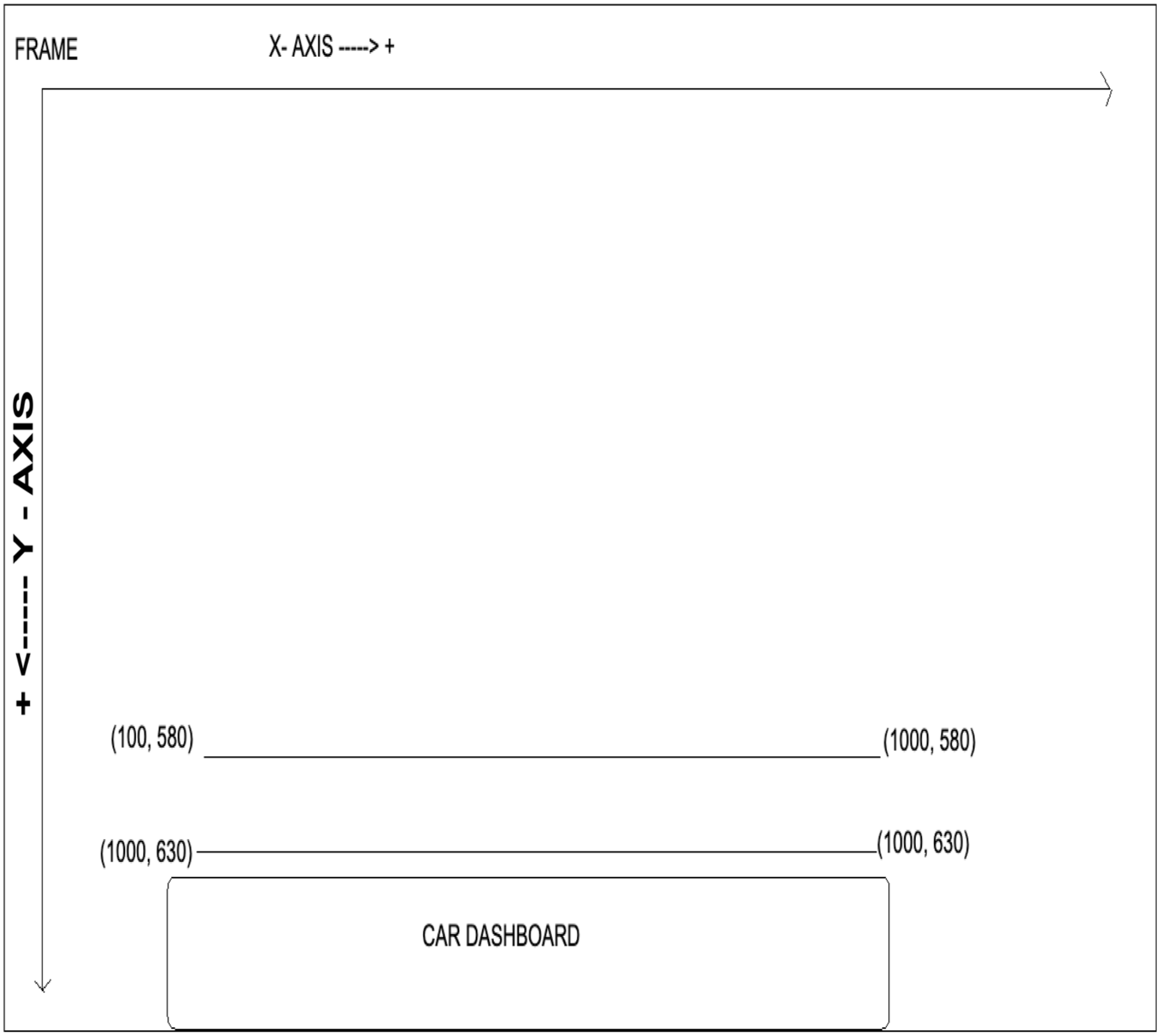
The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1 × 1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224 × 224 input image) and then double the resolution for detection.

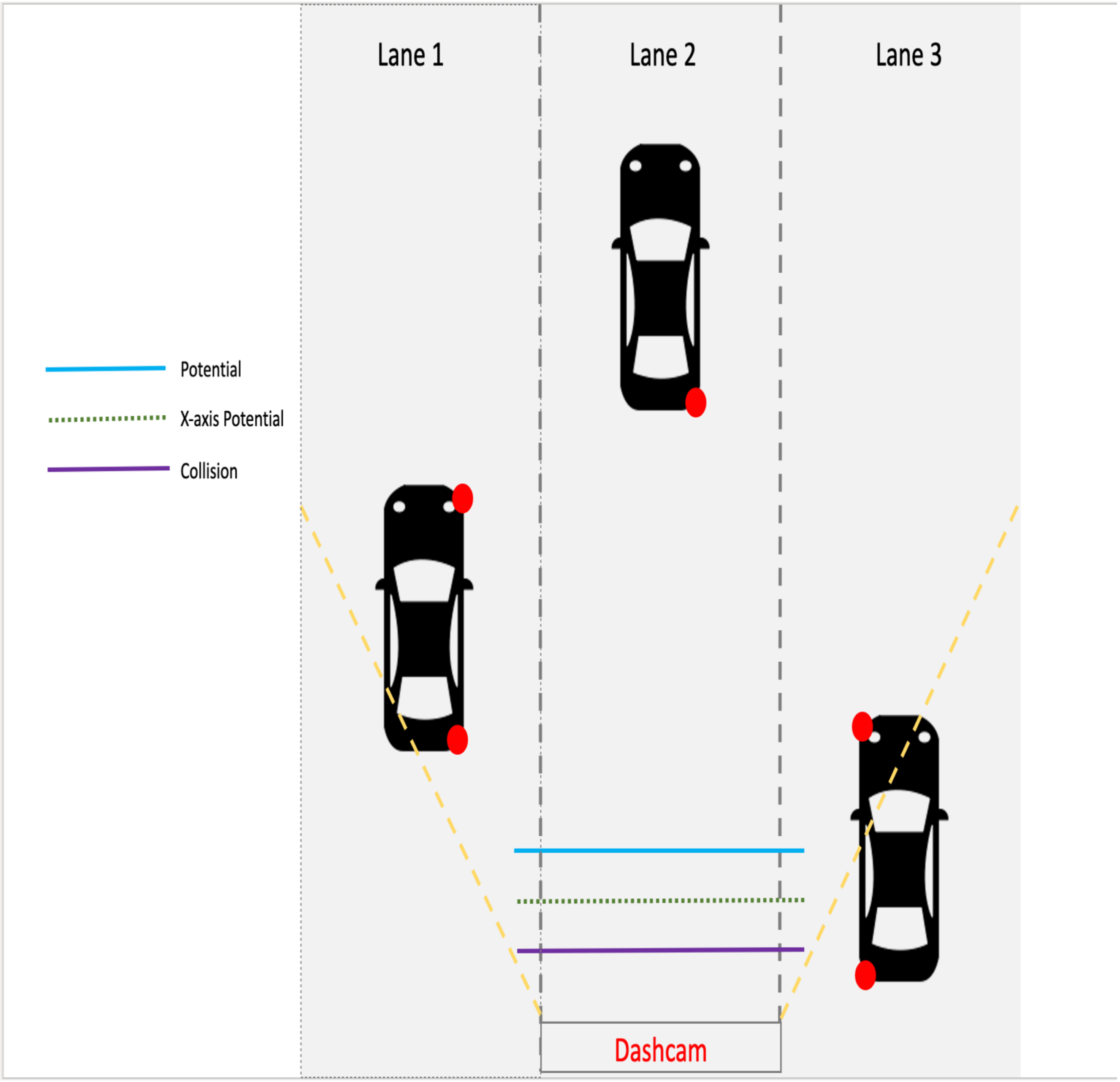
2. Dashboard Components:

- **Path to input video and Path to output video** : Can choose a video to process from the system and save the processed video at any specific location.
- **Training Type** : Can select an appropriate YOLO model to process the video. Once can have multiple custom/pretrained model from various open source datasets such as COCO or from Open Image dataset.
- **Confidence** : Can select confidence level at which the convolutional neural network should make its predictions. The range of it varies from 0 – 100 which is represented as 0 – 10 in the dashboard UI.
- **Collision Detection Config** :
  - Every video is being calculated on a pixel based graph where every pixel can be altered as a coordinate.
  - The X-Axis of the graph tends to increase as it goes right side of the frame while Y-Axis tends to increase in the downward direction of the frame.
  - Both X and Y Axis have been set to a configurable threshold coordinate with respect to car dashboard visibility.

- The default threshold range for X axis is from 100 – 100 pixel coordinate while the threshold value for Y axis differs for both potential collision detection and collision detection. The default threshold value for Y axis is 580 and 630 respectively for potential collision detection and collision detection.

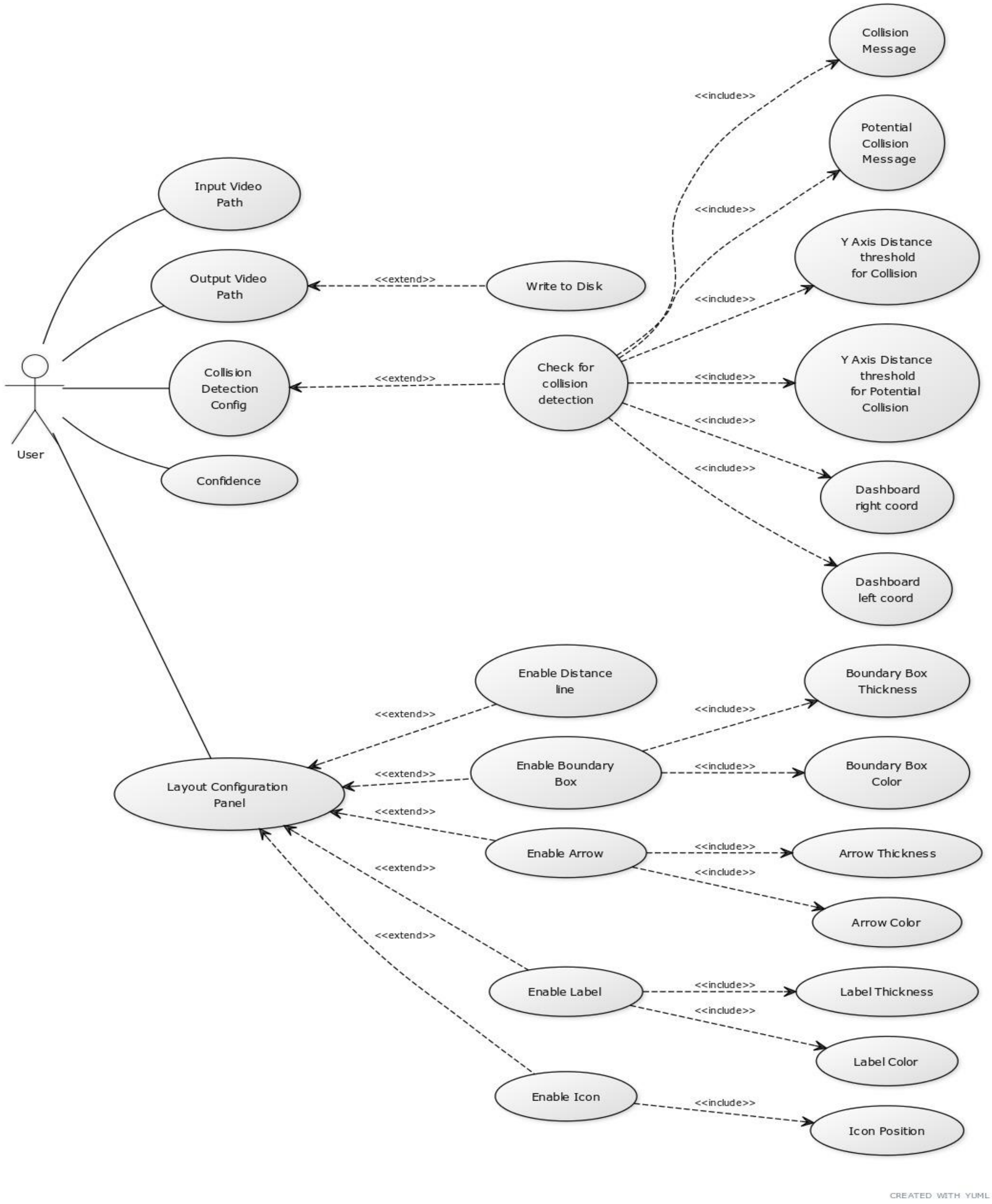
- Below is the diagram for the Collision Detection configs.


FRAME
X- AXIS -----> +
+ <------ Y - AXIS
(100, 580)
(1000, 580)
(1000, 630)
(1000, 630)
CAR DASHBOARD



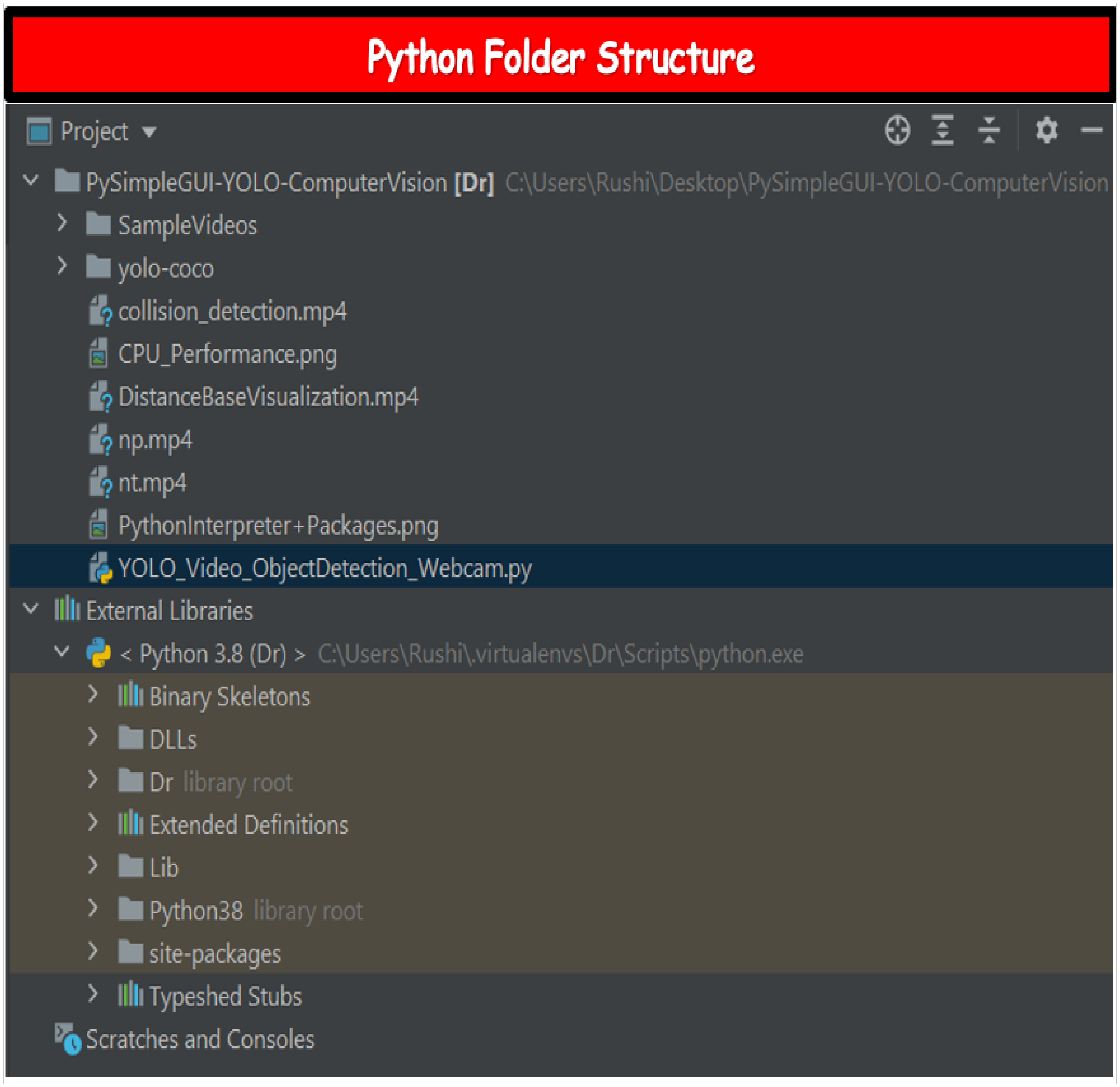
- **Layout Configuration Panel** : Consists of various manipulation parameters for boundary box, distance based filters as well distance based lines. It also gives the ability for directional visibility control.

- **Use Case Diagram**



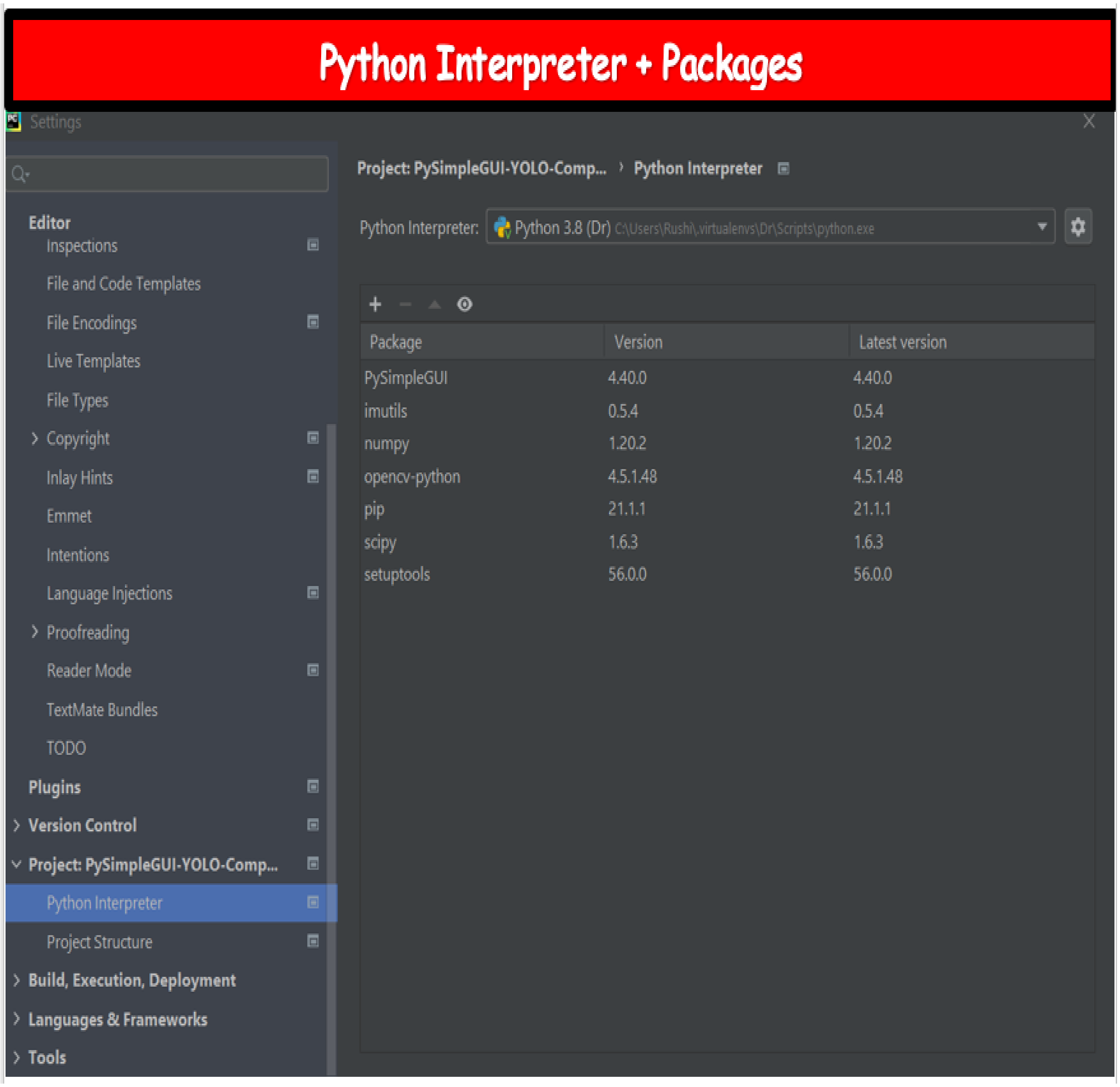
System Design

- **Screen Layouts**



Python Folder Structure

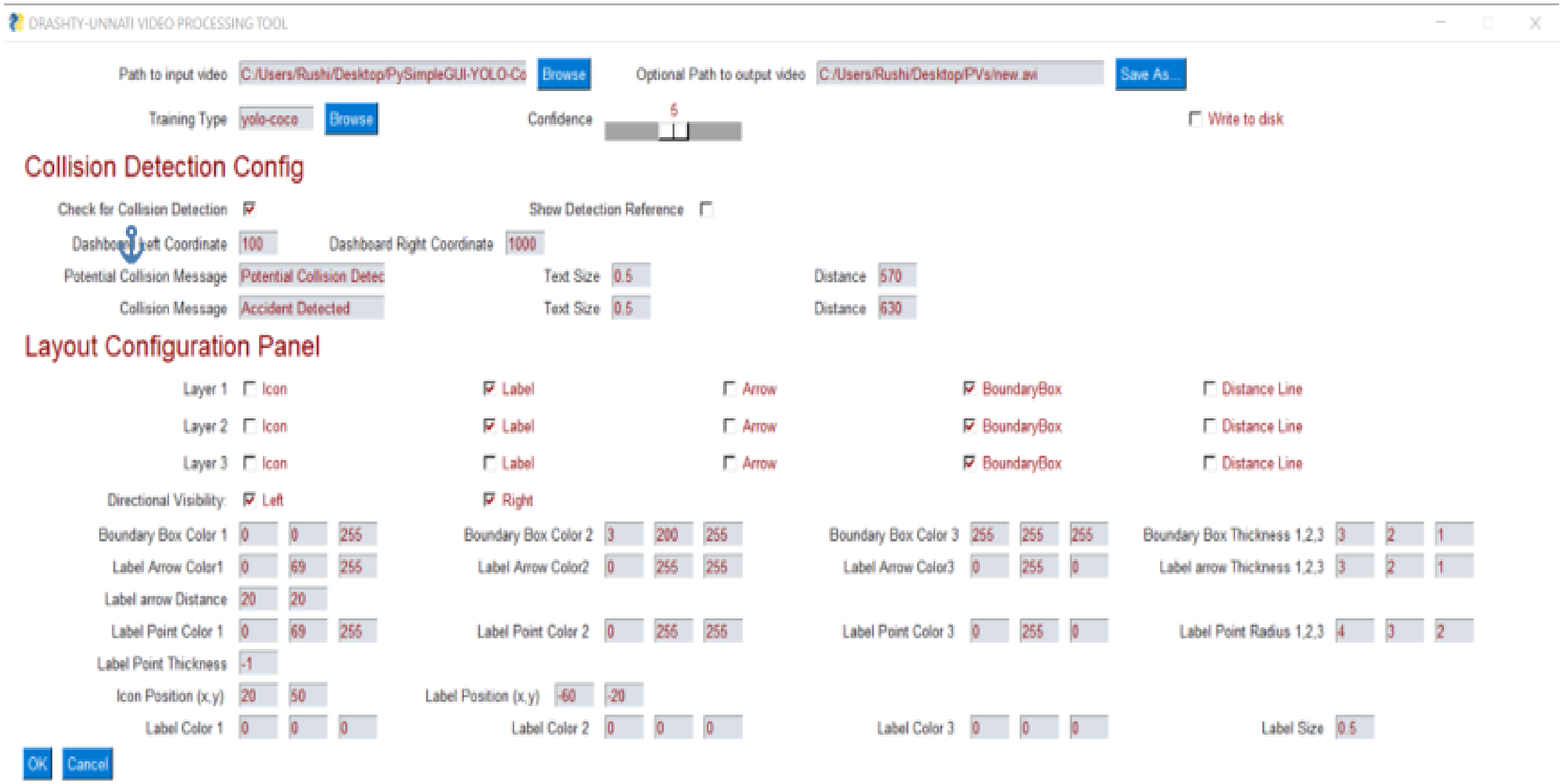
```
PySimpleGUI-YOLO-ComputerVision [Dr] C:\Users\Rushi\Desktop\PySimpleGUI-YOLO-ComputerVision
    SampleVideos
    yolo-coco
    collision_detection.mp4
    CPU_Performance.png
    DistanceBaseVisualization.mp4
    np.mp4
    nt.mp4
    PythonInterpreter+Packages.png
    YOLO_Video_ObjectDetection_Webcam.py
External Libraries
    < Python 3.8 (Dr) > C:\Users\Rushi\.virtualenvs\Dr\Scripts\python.exe
        Binary Skeletons
        DLLs
        Dr library root
        Extended Definitions
        Lib
        Python38 library root
        site-packages
        Typeshed Stubs
Scratches and Consoles
```



Python Interpreter + Packages

Python Interpreter: Python 3.8 (Dr) C:\Users\Rushi\.virtualenvs\Dr\Scripts\python.exe

Package	Version	Latest version
PySimpleGUI	4.40.0	4.40.0
imutils	0.5.4	0.5.4
numpy	1.20.2	1.20.2
opencv-python	4.5.1.48	4.5.1.48
pip	21.1.1	21.1.1
scipy	1.6.3	1.6.3
setuptools	56.0.0	56.0.0



DRASHTY-UNNATI VIDEO PROCESSING TOOL

Path to input video C:/Users/Rushi/Desktop/PySimpleGUI-YOLO-Co Optional Path to output video C:/Users/Rushi/Desktop/PVs/new.avi

Training Type yolo-coco Confidence 5 ☐ Write to disk

Collision Detection Config

Check for Collision Detection ☑ Show Detection Reference ☐

Dashboard Left Coordinate 100 Dashboard Right Coordinate 1000

Potential Collision Message Potential Collision Detec Text Size 0.5 Distance 570

Collision Message Accident Detected Text Size 0.5 Distance 630

Layout Configuration Panel

Layer 1 ☐ Icon ☑ Label ☐ Arrow ☑ BoundaryBox ☐ Distance Line

Layer 2 ☐ Icon ☑ Label ☐ Arrow ☑ BoundaryBox ☐ Distance Line

Layer 3 ☐ Icon ☐ Label ☐ Arrow ☑ BoundaryBox ☐ Distance Line

Directional Visibility: ☑ Left ☑ Right

Boundary Box Color 1 0 0 255 Boundary Box Color 2 3 200 255 Boundary Box Color 3 255 255 255 Boundary Box Thickness 1,2,3 3 2 1

Label Arrow Color1 0 69 255 Label Arrow Color2 0 255 255 Label Arrow Color3 0 255 0 Label arrow Thickness 1,2,3 3 2 1

Label arrow Distance 20 20

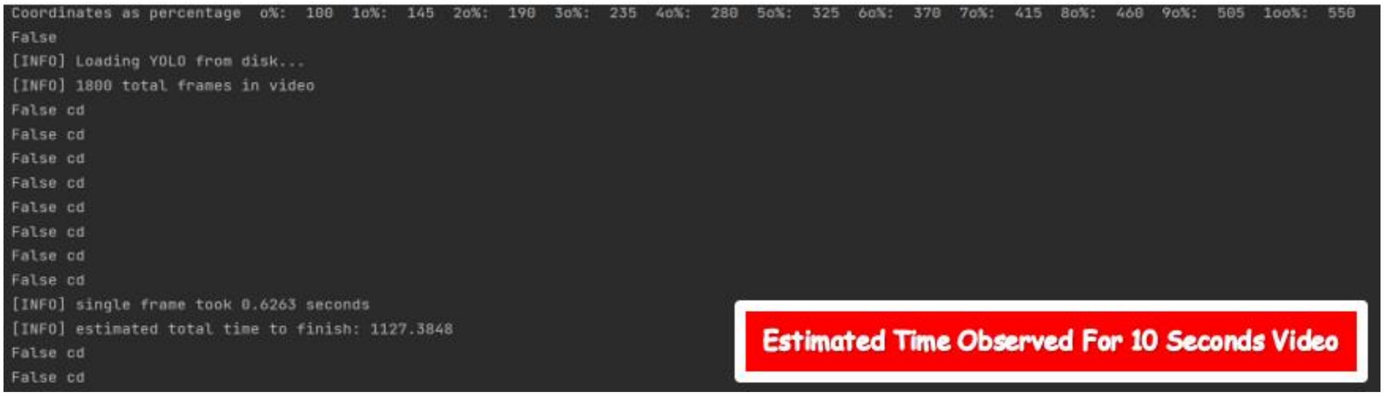
Label Point Color 1 0 69 255 Label Point Color 2 0 255 255 Label Point Color 3 0 255 0 Label Point Radius 1,2,3 4 3 2

Label Point Thickness -1

Icon Position (x,y) 20 50 Label Position (x,y) -60 -20

Label Color 1 0 0 0 Label Color 2 0 0 0 Label Color 3 0 0 0 Label Size 0.5

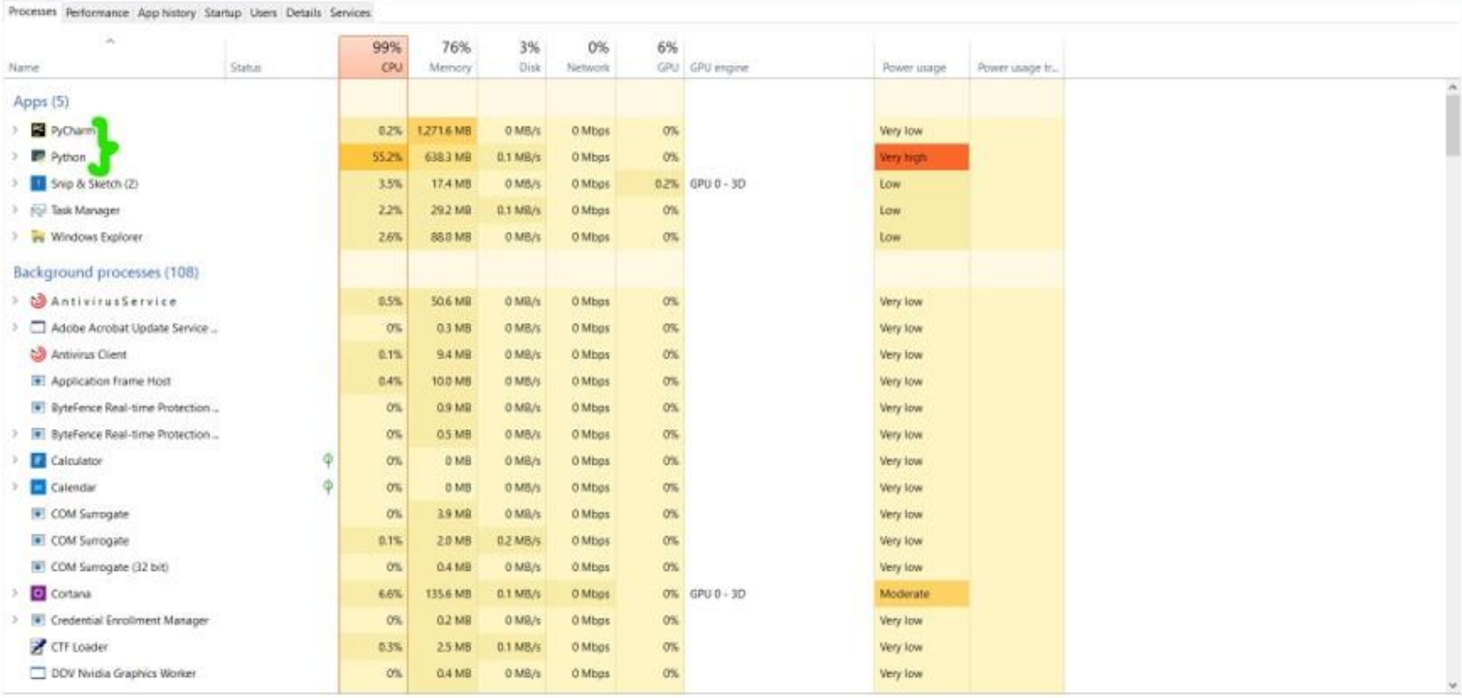
Dashboard - GUI



```
Coordinates as percentage  o%:  100  1o%:  145  2o%:  190  3o%:  235  4o%:  280  5o%:  325  6o%:  370  7o%:  415  8o%:  460  9o%:  505  1oo%:  550
False
[INFO] Loading YOLO from disk...
[INFO] 1800 total frames in video
False cd
False cd
False cd
False cd
False cd
False cd
False cd
False cd
[INFO] single frame took 0.6263 seconds
[INFO] estimated total time to finish: 1127.3848
False cd
False cd
```

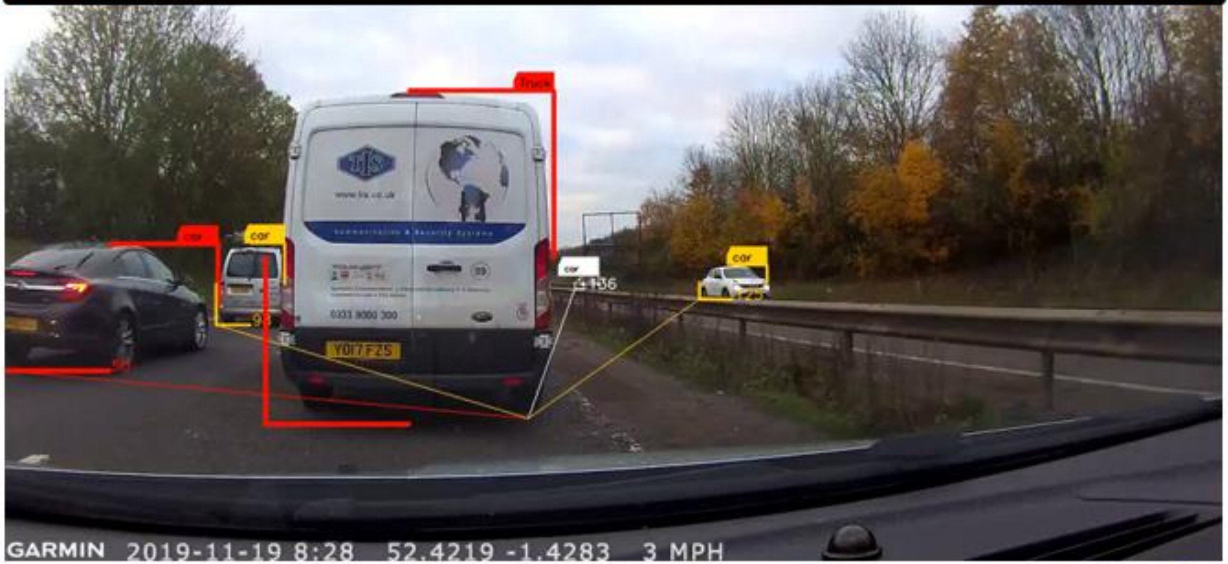
Estimated Time Observed For 10 Seconds Video

C.P.U. Performance And Utilization + Ram Useage + Power Consumption Status

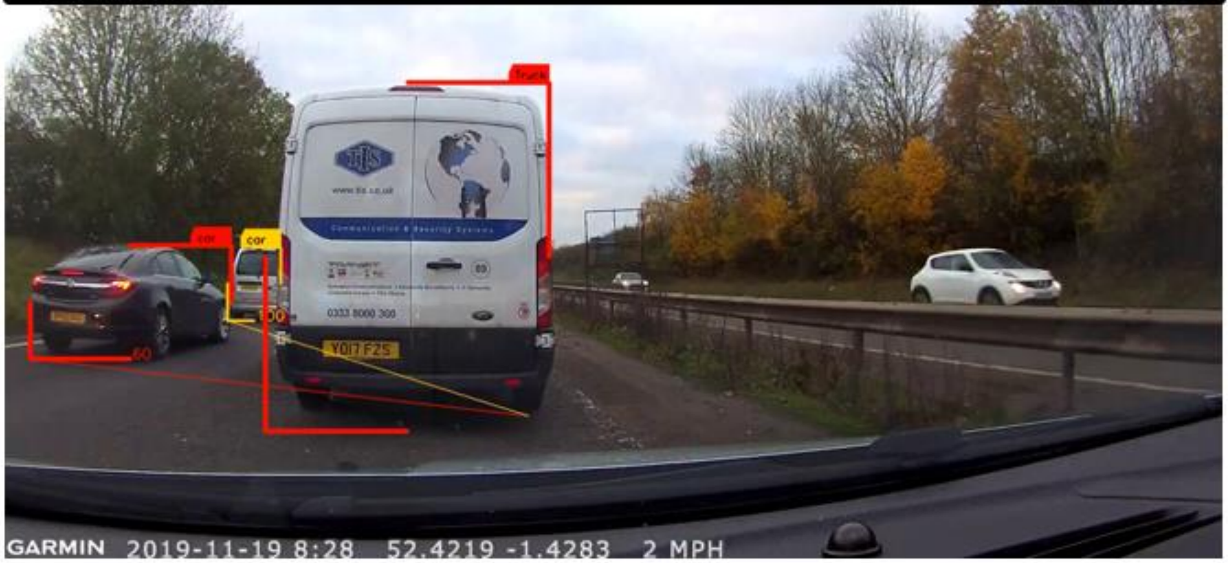


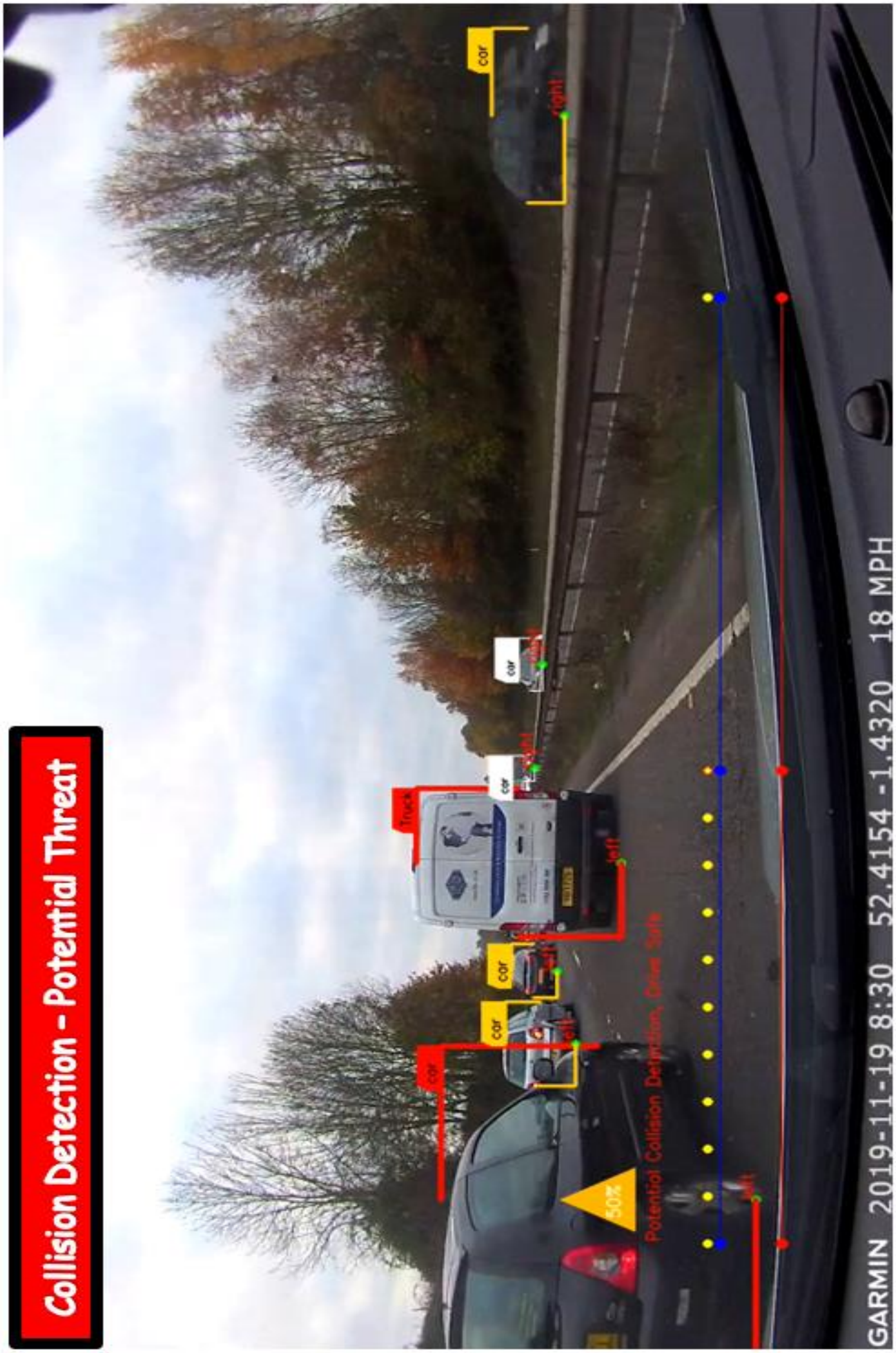
		99%	76%	3%	0%	6%			
Name	Status	CPU	Memory	Disk	Network	GPU	GPU engine	Power usage	Power usage tr...
Apps (5)									
PyCharm		0.2%	1,271.6 MB	0 MB/s	0 Mbps	0%		Very low	
Python		55.2%	638.3 MB	0.1 MB/s	0 Mbps	0%		Very high	
Snip & Sketch (2)		3.5%	17.4 MB	0 MB/s	0 Mbps	0.2%	GPU 0 - 3D	Low	
Task Manager		2.2%	29.2 MB	0.1 MB/s	0 Mbps	0%		Low	
Windows Explorer		2.6%	88.0 MB	0 MB/s	0 Mbps	0%		Low	
Background processes (108)									
AntivirusService		0.5%	50.6 MB	0 MB/s	0 Mbps	0%		Very low	
Adobe Acrobat Update Service ...		0%	0.3 MB	0 MB/s	0 Mbps	0%		Very low	
Antivirus Client		0.1%	9.4 MB	0 MB/s	0 Mbps	0%		Very low	
Application Frame Host		0.4%	10.0 MB	0 MB/s	0 Mbps	0%		Very low	
ByteFence Real-time Protection ...		0%	0.9 MB	0 MB/s	0 Mbps	0%		Very low	
ByteFence Real-time Protection ...		0%	0.5 MB	0 MB/s	0 Mbps	0%		Very low	
Calculator	φ	0%	0 MB	0 MB/s	0 Mbps	0%		Very low	
Calendar	φ	0%	0 MB	0 MB/s	0 Mbps	0%		Very low	
COM Surrogate		0%	3.9 MB	0 MB/s	0 Mbps	0%		Very low	
COM Surrogate		0.1%	2.0 MB	0.2 MB/s	0 Mbps	0%		Very low	
COM Surrogate (32 bit)		0%	0.4 MB	0 MB/s	0 Mbps	0%		Very low	
Cortana		6.6%	135.6 MB	0.1 MB/s	0 Mbps	0%	GPU 0 - 3D	Moderate	
Credential Enrollment Manager		0%	0.2 MB	0 MB/s	0 Mbps	0%		Very low	
CTF Loader		0.3%	2.5 MB	0.1 MB/s	0 Mbps	0%		Very low	
DDV Nvidia Graphics Worker		0%	0.4 MB	0 MB/s	0 Mbps	0%		Very low	

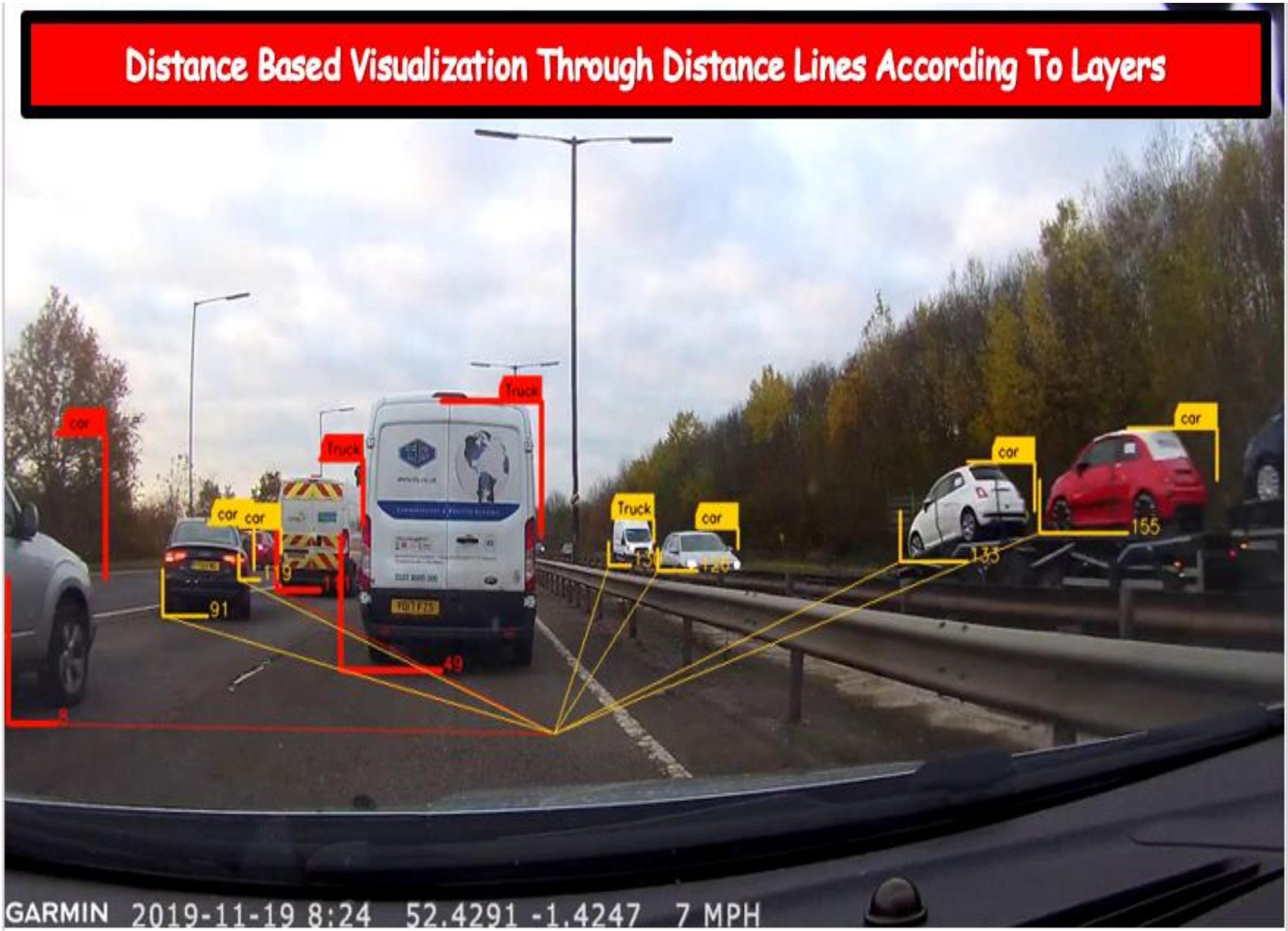
Directional Visibility - Both : Left,Right



Directional Visibility - Left




Collision Detection - Potential Threat
car
right
car
right
car
right
Truck
left
car
left
car
left
car
left
50%
Potential Collision Detection, Drive Safe
GARMIN 2019-11-19 8:30 52.4154 -1.4320 18 MPH



Distance Based Visualization Through Distance Lines According To Layers

System Testing

- **Testing Strategies :**
  - **Methodology used :** Agile Methodology

  - **Brief about Agile Methodology** : Agile testing is a software testing process that follows the principles of agile software development. Agile testing aligns with iterative development methodology in which requirements develop gradually from customers and testing teams. The development is aligned with customer requirements.

  - **Principles of Agile** :

    - **Testing is continuous:** Agile team tests continuously because it is the only way to ensure continuous progress of the product.
    - **Continuous feedback-** Agile testing provides feedback on an ongoing basis and this is how your product meets the business needs.
    - **Tests performed by the whole team:** In a traditional software development life cycle, only the test team is responsible for testing but in agile testing, the developers and the business analysts also test the application.
    - **Decrease time of feedback response:** The business team is involved in each iteration in agile testing & continuous feedback shortens the time of feedback response.

- **Simplified & clean code:** All the defects which are raised by the agile team are fixed within the same iteration and it helps in keeping the code clean and simplified.
- **Less documentation:** Agile teams use a reusable checklist, the team focuses on the test instead of the incidental details.
- **Test Driven:** In agile methods, testing is performed at the time of implementation whereas, in the traditional process, the testing is performed after implementation.

- **Advantages of Agile** : In agile testing, the test plan is written as well as updated for every release. A test plan in agile includes:

    - The scope of the testing
    - Consolidating new functionalities to be tested
    - Performance & load testing
    - Consideration of infrastructure
    - Risks Plan

Future Enhancements

1. Get the analysis output in JSON format.
2. Add custom CNN instead of YOLO.
3. Use transfer learning instead for pre trained data
4. Attach GRPC to Python APIs to the SDK for data manipulation.

Bibliography/References

- For the Successful working of our Project we have referred many sources for the code snippets; tips n tricks from the various applications as well as websites.

■ https://pjreddie.com/darknet/yolo/
■ www.stackoverflow.com
■ https://arxiv.org/pdf/1506.02640.pdf
■ https://dev.to/