**Object Detection and Tracking using Deep Learning and Artificial Intelligence for Video Surveillance Applications**

1. **INTRODUCTION**

Over the past years domains like image analysis and video analysis has gained a wide scope of applications. CV and AI are two main technologies dominating technical society. Technologies try to depict the biology of human. Human vision is the sense through which a perception of outer 3D world is perceived. Human Intelligence is trained over years to distinguish and process scene captured by eyes. These intuitions acts as a crux to budding new technologies. Rich resource is now accelerating researchers to excavate more details form the images. These developments are due to state-of the-art methods like CNN. Applications from Google, Facebook, Microsoft, and Snapchat are all results of tremendous improvement in Computer vision and Deep learning. During time, the vision-based technology has transformed from just a sensing modality to intelligent computing systems which can understand the real world. Computer vision applications like vehicle navigation, surveillance and autonomous robot navigation find Object detection and tracking as important challenges. For tracking vehicles and other real word objects, video surveillance is a dynamic environment. In this paper, efficient algorithm is designed for object detection and tracking for video Surveillance in complex environment.

Object detection and tracking goes hand in hand for computer vision applications. Object detection is identifying object or locating the instance of interest in-group of suspected frames. Object tracking is identifying trajectory or path; object takes in the concurrent frames. Image obtained from dataset is, collection of frames. Basic block diagram of object detection and tracking is shown in Fig. 1. Data set is divided into two parts. 80 % of images in dataset are used for training and 20 % for testing. Image is considered to find objects in it by using algorithms CNN and YOLOv3. A bounding box is formed across object with Intersection over union (IoU) > 0.5. Detected bounding box is sent as references for neural networks aiding them to perform Tracking. Bounded box is tracked in concurrent frames using Multi Object Tracking (MOT). Importance of this research work is used to estimate traffic density in traffic junctions, in autonomous vehicles to detect various kinds of objects with varying illumination, smart city development and intelligent transport systems.

**Objective of the project:**

Data is the new oil in current technological society. The impact of efficient data has changed benchmarks of performance in terms of speed and accuracy. The enhancement is visualizable because the processing of data is performed by two buzzwords in industry called Computer Vision (CV) and Artificial Intelligence (AI). Two technologies have empowered major tasks such as object detection and tracking for traffic vigilance systems. As the features in image increases demand for efficient algorithm to excavate hidden features increases. Convolution Neural Network (CNN) model is designed for urban vehicle dataset for single object detection and YOLOv3 for multiple object detection on KITTI and COCO dataset. Model performance is analyzed, evaluated and tabulated using performance metrics such as True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN), Accuracy, Precision, confusion matrix and mean Average Precession (mAP). Objects are tracked across the frames using YOLOv3 and Simple Online Real Time Tracking (SORT) on traffic surveillance video. This paper upholds the uniqueness of the state of the art networks like DarkNet. The efficient detection and tracking on urban vehicle dataset is witnessed. The algorithms give real-time, accurate, precise identifications suitable for real-time traffic applications**.**

**­­­2. LITERATURE SURVEY**

**Learning Framework for Robust Obstacle Detection, Recognition, and Tracking**

This paper introduces a general framework for detection, recognition, and tracking preceding vehicles and pedestrians based on a deep learning approach. The proposed framework combines a novel deep learning approach with the use of multiple sources of local patterns and depth information to yield robust on-road vehicle and pedestrian detection, recognition, and tracking. The proposed system is first based on robust obstacle detection to identify obstacles appearing along the road that are likely to be vehicles and pedestrians, implemented as an efficient adaptive U-V disparity algorithm. Second, the results from the obstacle detection stage are input into a novel vehicle and pedestrian recognition system based on a deep learning model that processes multiple sources of depth information and local patterns. Finally, the results from the recognition stage are used to track detected vehicles or pedestrians in the next frame by means of a proposed tracking and validation model. The proposed framework has been thoroughly evaluated by inputting several vehicle and pedestrian data sets that were collected under various driving conditions. Experimental results show that this framework provides robust vehicle and pedestrian detection, recognition, and tracking with high accuracy, and also satisfies the real-time requirements of driver assistance systems.

**Detecting Abnormal Events in University Areas**

This paper presents a distinct video surveillance system which took place in the Lebanese International University Saida- Campus., which is considered as a very crowded environment., and reveals if there is an unusual event. Our main target is to apply simple procedures that will be present as a future”s benchmark. The work is split into three major parts., starting by dividing the video frame into zones., then to compute the magnitude of optical flow in each., and finally to analyze these data and classify it., based on a logical threshold., as normal or abnormal events. We implement our results based on Histogram of Magnitudes for each zone (HOM) and the outcome met our expectations.

**Detection of unwanted traffic congestion based on existing surveillance system using in freeway via a CNN-architecture trafficnet**

Detection of traffic congestion is important for route guidance using in intelligent transport system (ITS) to prevent jam escalation. Although the surveillance system has been used in freeway for years, it is hard to automatically identify and report traffic congestion in complicated transportation scene according to various illumination, weather and other disturbances. The detection process based on human eye is time-consuming and tedious as the machine detection accuracy is not high enough to meet the requirements of practical applications. In this paper, a new classifier is proposed using convolutional neural networks (CNN) to generate four TrafficNet based on two championships of ILSVRC including AlexNet and VGGNet. Instead of using fully-connected layers in AlexNet and VGGNet, a support vector machine (SVM) are used after CNN architecture. Congestion and non-congestion images are trained and tested through this new structure. Image database with more than 30000 images are extracted from existing traffic surveillance video and corresponding labels are added manually. With database, those TrafficNet are trained and tested, detection accuracy and training time of those TrafficNet are compared. The experimental results show that the accuracy of proposed method can reach up to 90%, which is much higher than traditional method based on feature extraction without deep learning.

**The Research of Target Tracking Algorithm Based on an Improved PCANet**

The feature extraction method will greatly affect the performance of the target tracking algorithm. In traditional feature extraction methods, feature descriptors are manually designed, such as HOG features, etc. which does not express spatial information very well. In this paper, a novel target tracking method is proposed. This method uses a lightweight deep learning model, called PCANet network, to extract features. In the previous work, the number of PCA layer filters is determined through a large number of experiments. In this paper, the number of filters in the PCA layer is determined by the cumulative contribution rate, and which achieves the adaptive adjustment of the network parameters. Firstly, the region of interest is acquired by particle filtering; Secondly, the depth characteristics of the image are extracted via PCANet; Finally, the target is determined by the SVM classifier. The results of the experiment show that this algorithm has strong robustness, because the target tracking can be tracked accurately under the condition of illumination change, occlusion and rapid movement.

**Real-Time Detection, Tracking and Classification of Multiple Moving Objects in UAV Videos**

Unnamed Aerial Vehicles (UAVs) are becoming increasingly popular and widely used for surveillance and reconnaissance. There are some recent studies regarding moving object detection, tracking, and classification from UAV videos. A unifying study, which also extends the application scope of such previous works and provides real-time results, is absent from the literature. This paper aims to fill this gap by presenting a framework that can robustly detect, track and classify multiple moving objects in real-time, using commercially available UAV systems and a common laptop computer. The framework can additionally deliver practical information about the detected objects, such as their coordinates and velocities. The performance of the proposed framework, which surpasses human capabilities for moving object detection, is reported and discussed.

**Object Tracking with Shallow Convolution Feature**

Traditional target tracking algorithm uses manual extraction of features, which is difficult to cope with the challenges of rotation and occlusion and deformation. Based on the deep learning method, the convolution neural network is used to extract the features. Because of lost a lot of spatial information in the convolution process, it's easy to make the tracking target drift. In this paper, we use a shallow convolutional network without second training to extract features for tracking, which combines hard negative mining technology and bounding box regression to refine the target location. We have compared our tracker performance with others state-of-the-art tracker. The obtained experimental results in OTB dataset demonstrate the effectiveness of our proposed tracker has outperformed the compared tracking algorithms.

**Convolutional Neural Networks Based Fire Detection in Surveillance Videos**

The recent advances in embedded processing have enabled the vision based systems to detect fire during surveillance using convolutional neural networks (CNNs). However, such methods generally need more computational time and memory, restricting its implementation in surveillance networks. In this research paper, we propose a cost-effective fire detection CNN architecture for surveillance videos. The model is inspired from GoogleNet architecture, considering its reasonable computational complexity and suitability for the intended problem compared to other computationally expensive networks such as AlexNet. To balance the efficiency and accuracy, the model is fine-tuned considering the nature of the target problem and fire data. Experimental results on benchmark fire datasets reveal the effectiveness of the proposed framework and validate its suitability for fire detection in CCTV surveillance systems compared to state-of-the-art methods.

**Cell Tracking with Deep Learning and the Viterbi Algorithm**

We present a cell tracking pipeline that combines deep cell segmentation with a Viterbi algorithm tracker to accurately detect and track cells in microscopy videos. Our pipeline handles large illumination shifts, large appearance variability in the cells, and heavy occlusion from other cells and debris. We first train a Fully Convolutional Network (FCN) to detect the cells, then track the cells across frames using a tracker based on the Viterbi algorithm. We evaluate our algorithm on a dataset featuring Escherichia coli (E. coli) where the experimental goal is to immobilize the E. coli using blue light, thus making the dataset especially challenging due to large illumination shifts. Our results demonstrate that despite these challenges, our pipeline is able to accurately detect and track the cells.

**An object tracking method using deep learning and adaptive particle filter for night fusion image**

In this paper, we propose an online visual tracking algorithm for fused sequences via deep learning and adaptive Particle filter (PF). Our algorithm pretrains a simplified Convolution Neural Network (CNN) to obtain a generic target representation. The outputs from the hidden layers of the network help to form the tracking model for an online PF. During tracking, the moving information guides the distribution of particle samples. The tests illustrate competitive performance compared to the state-of-art tracking algorithms especially when the target or camera moves quickly.

**Online Multi-Object Tracking Using Selective Deep Appearance Matching**

In this paper, we focus on designing appearance matching network and solving computational bottleneck problem of it. From the development of deep neural network and graphic device(GPU), many research topics in computer vision (e.g. detection, classification) achieved state-of-the-art performance using convolutional neural network(CNN). In multi-object tracking, also, there have been several works which used CNN for extracting appearance feature of targets. Although, deep appearance feature improved an accuracy of tracking, it increased processing time and made an algorithm hard to be applied in real-world situation. So, we propose a simple technique to improve speed by removing redundant appearance matchings. Also, we propose a structure of joint-input siamese network and method to train it. We verify the performance of our work by comparison with recent online trackers.

**Existing System**

Object tracking is a very challenging task in the presence of variability Illumination condition, background motion, complex object shape partial and full object occlusions. Object detection and location in digital images has become one of the most important applications for industries to ease user, save time and to achieve parallelism. This is not a new technique but improvement in object detection is still required in order to achieve the targeted objective more efficiently and accurately. The main aim of studying and researching computer vision is to simulate the behavior and manner of human eyes directly by using a computer and later on develop a system that reduces human efforts.

**Disadvantage**

1. Objects are not easily Tracking & Detected.
2. Less Accuracy.
3. Human effort is more.
4. More Time Tacking process.

**Proposed System**

In order to overcome the issue of detection, tracking related to object movement and appearance. Most of the algorithm focuses on the tracking algorithm to smoothen the video sequence. In this project using python and OPENCV module we are detecting objects from videos and webcam. This application consists of two modules such as ‘Browse System Videos’ and ‘Start Webcam Video Tracking’.

Browse System Videos Using this module application allow user to upload any video from his system and application will connect to that video and start playing it, while playing if application detect any object then it will mark that object with bounding boxes, while playing video if user wants to stop tracking then he need to press ‘q’ key from keyboard to stop video playing.

Start Webcam Video Tracking: Using this module application connect itself with inbuilt system webcam and start video streaming, while streaming if application detect any object then it will surround that object with bounding boxes, while playing press ‘q’ to stop web cam streaming.

**Advantage**

1. Objects are Tracking & Detected Easily.

**Modules:**

This application consists of two modules such as ‘Browse System Videos’ and ‘Start Webcam Video Tracking’.

**Browse System Videos:** Using this module application allow user to upload any video from his system and application will connect to that video and start playing it, while playing if application detect any object then it will mark that object with bounding boxes, while playing video if user wants to stop tracking then he need to press ‘q’ key from keyboard to stop video playing.

**Start Webcam Video Tracking:** Using this module application connect itself with inbuilt system webcam and start video streaming, while streaming if application detect any object then it will surround that object with bounding boxes, while playing press ‘q’ to stop web cam streaming.

To implement this project we are using object tracking algorithms from OPENCV python API.

**3.3. PROCESS MODEL USED WITH JUSTIFICATION**

**SDLC (Umbrella Model):**

**Umbrella Activity**

**Umbrella Activity**

**Umbrella Activity**

1. Feasibility Study
2. TEAM FORMATION
3. Project Specification PREPARATION

Business Requirement Documentation

ANALYSIS & DESIGN

CODE

UNIT TEST

DOCUMENT CONTROL

ASSESSMENT

TRAINING

INTEGRATION & SYSTEM TESTING

DELIVERY/INSTALLATION

ACCEPTANCE TEST

Requirements Gathering

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

* Requirement Gathering
* Analysis
* Designing
* Coding
* Testing
* Maintenance

**Requirements Gathering** **stage:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

* Feasibility study is all about identification of problems in a project.
* No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
* Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage:**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

  
When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

* **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artefacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labour data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

**3.4. Software Requirement Specification**

**3.4.1. Overall Description**

A Software Requirements Specification (SRS) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) is a complete description of the behaviour of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development lifecycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [Business requirements](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms *what* must be delivered or accomplished to provide value.
* Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
* Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:
* **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

* **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

* **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

**3.4.2. External Interface Requirements**

**User Interface**

The user interface of this system is a user friendly python Graphical User Interface.

**Hardware Interfaces**

The interaction between the user and the console is achieved through python capabilities.

**Software Interfaces**

The required software is python.

**Operating Environment**

Windows XP.

**HARDWARE REQUIREMENTS:**

# Processor - Pentium –IV

* Speed - 1.1 Ghz
* RAM - 256 MB(min)
* Hard Disk - 20 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA

**SOFTWARE REQUIREMENTS:**

* Operating System - Windows7/8
* Programming Language - Python

**4. SYSTEM DESIGN**

**UML Diagram:**

**Class Diagram:**

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake

**Class Diagram:**



**Use case Diagram:**

A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

**Use case Diagram:**



**Sequence diagram:**

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.



**Collaboration diagram:**

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system.



**5. IMPLEMETATION**

**5.1 Python**

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

**History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

**Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

**Features of Python:**

**A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

**Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software’s written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

**Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

**Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

**A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

**Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

**Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

**Applications of Python:**

**1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

a = 2

b = 3

sum = a + b

print(sum)

**2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

**3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

**4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

**5.2 Sample Code:**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

from tkinter.filedialog import askopenfilename

import imutils

import time

import cv2

import numpy as np

main = tkinter.Tk()

main.title("Object Tracking Using Python")

main.geometry("1300x1200")

net = cv2.dnn.readNetFromCaffe("MobileNetSSD\_deploy.prototxt.txt","MobileNetSSD\_deploy.caffemodel")

global filename

global train

global ga\_acc, bat\_acc, bee\_acc

global classifier

CLASSES = ["background", "aeroplane", "bicycle", "bird", "boat",

"bottle", "bus", "car", "cat", "chair", "cow", "diningtable",

"dog", "horse", "motorbike", "person", "pottedplant", "sheep",

"sofa", "train", "tvmonitor"]

COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))

def uploadVideo():

global filename

filename = filedialog.askopenfilename(initialdir="videos")

pathlabel.config(text=filename)

text.delete('1.0', END)

text.insert(END,filename+" loaded\n");

vc = cv2.VideoCapture(filename)

while True:

frame = vc.read()

frame = frame if filename is None else frame[1]

if frame is None:

break

frame = imutils.resize(frame, width=500)

(h, w) = frame.shape[:2]

blob = cv2.dnn.blobFromImage(cv2.resize(frame, (300, 300)),0.007843, (300, 300), 127.5)

net.setInput(blob)

detections = net.forward()

for i in np.arange(0, detections.shape[2]):

confidence = detections[0, 0, i, 2]

if confidence > 0.2:

idx = int(detections[0, 0, i, 1])

box = detections[0, 0, i, 3:7] \* np.array([w, h, w, h])

(startX, startY, endX, endY) = box.astype("int")

if (confidence \* 100) > 50:

label = "{}: {:.2f}%".format(CLASSES[idx],confidence \* 100)

cv2.rectangle(frame, (startX, startY), (endX, endY),COLORS[idx], 2)

y = startY - 15 if startY - 15 > 15 else startY + 15

cv2.putText(frame, "Object detected in video", (startX, y),cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, COLORS[idx], 2)

text.insert(END,"Object detected in video"+"\n")

cv2.imshow("Frame", frame)

key = cv2.waitKey(1) & 0xFF

if key == ord("q"):

break

vc.stop() if filename is None else vc.release()

cv2.destroyAllWindows()

def webcamVideo():

text.delete('1.0', END)

webcamera = cv2.VideoCapture(0)

time.sleep(0.25)

oldFrame = None

while True:

(grab, frame) = webcamera.read()

if not grab:

break

frame = imutils.resize(frame, width=500)

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

gray = cv2.GaussianBlur(gray, (21, 21), 0)

if oldFrame is None:

oldFrame = gray

continue

(h, w) = frame.shape[:2]

blob = cv2.dnn.blobFromImage(cv2.resize(frame, (300, 300)),0.007843, (300, 300), 127.5)

net.setInput(blob)

detections = net.forward()

for i in np.arange(0, detections.shape[2]):

confidence = detections[0, 0, i, 2]

if confidence > 0.2:

idx = int(detections[0, 0, i, 1])

box = detections[0, 0, i, 3:7] \* np.array([w, h, w, h])

(startX, startY, endX, endY) = box.astype("int")

if (confidence \* 100) > 50:

label = "{}: {:.2f}%".format(CLASSES[idx],confidence \* 100)

cv2.rectangle(frame, (startX, startY), (endX, endY),COLORS[idx], 2)

y = startY - 15 if startY - 15 > 15 else startY + 15

cv2.putText(frame, "Object detected in video", (startX, y),cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, COLORS[idx], 2)

text.insert(END,"Object detected in video"+"\n")

cv2.imshow("Frame", frame)

key = cv2.waitKey(1) & 0xFF

if key == ord("q"):

break

#webcamera.stop()

webcamera.release()

cv2.destroyAllWindows()

def exit():

main.destroy()

font = ('times', 16, 'bold')

title = Label(main, text='Object Tracking Using Python')

title.config(bg='light cyan', fg='pale violet red')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 14, 'bold')

uploadButton = Button(main, text="Browse System Videos", command=uploadVideo)

uploadButton.place(x=50,y=100)

uploadButton.config(font=font1)

pathlabel = Label(main)

pathlabel.config(bg='light cyan', fg='pale violet red')

pathlabel.config(font=font1)

pathlabel.place(x=460,y=100)

webcamButton = Button(main, text="Start Webcam Video Tracking", command=webcamVideo)

webcamButton.place(x=50,y=150)

webcamButton.config(font=font1)

exitButton = Button(main, text="Exit", command=exit)

exitButton.place(x=330,y=150)

exitButton.config(font=font1)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=150)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=250)

text.config(font=font1)

main.config(bg='snow3')

main.mainloop()

**6. TESTING**

**Implementation and Testing:**

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

**Implementation**

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

**Testing**

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

**System Testing**

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

**Module Testing**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

**Integration Testing**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

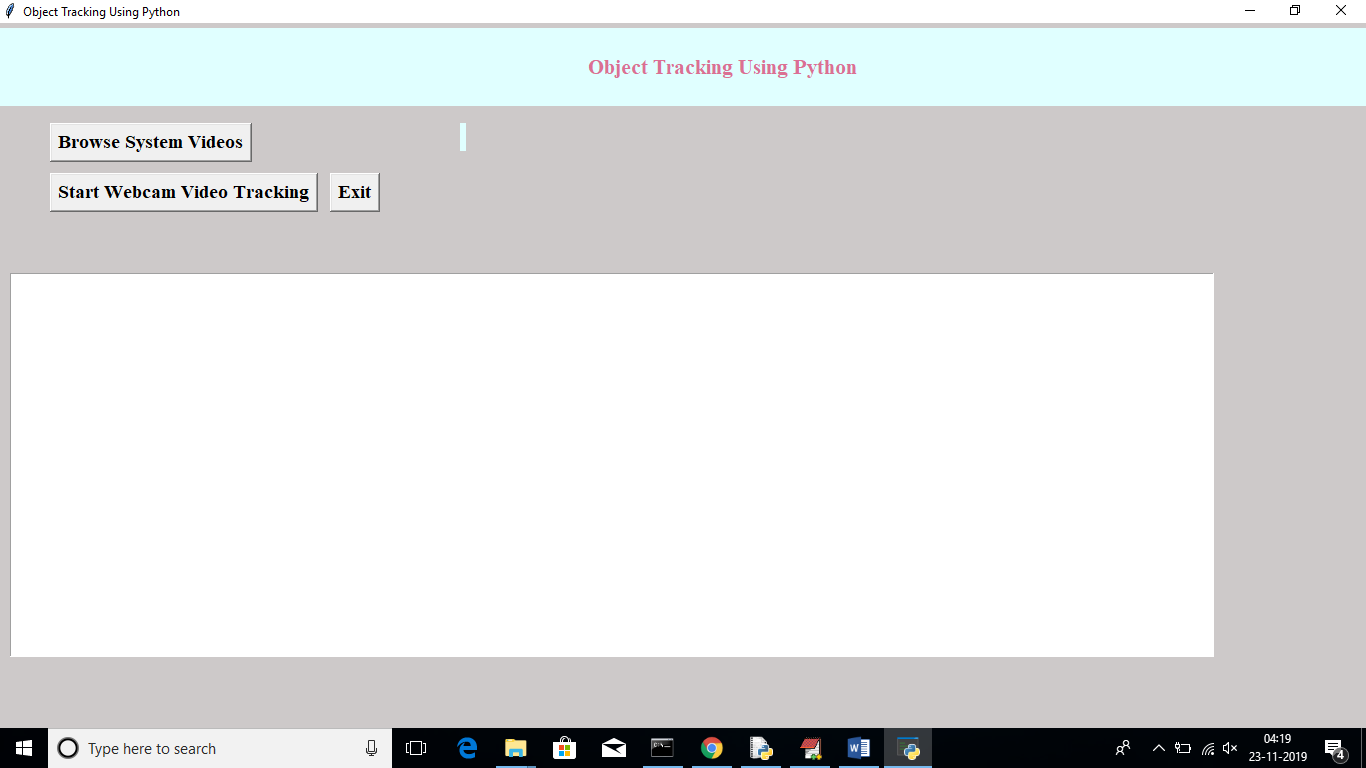
**Acceptance Testing**

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

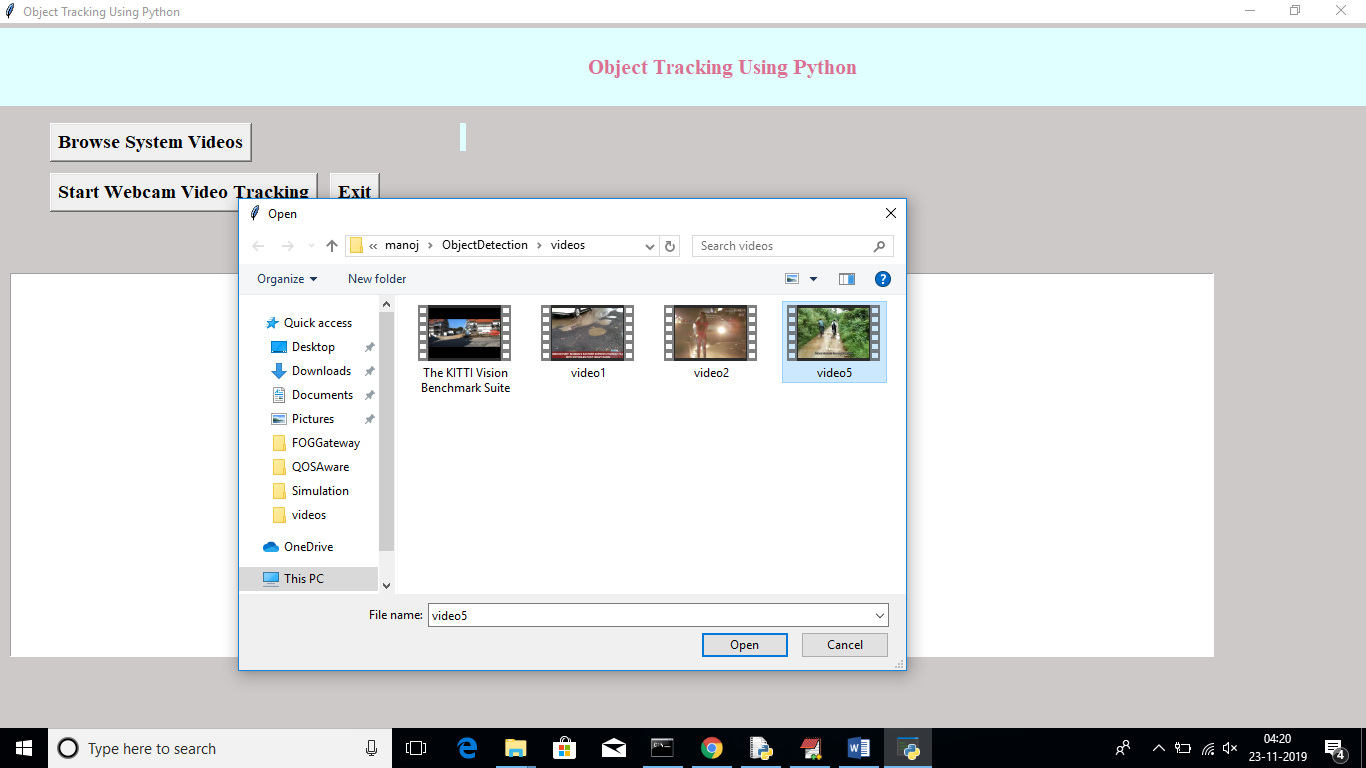
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | **Test Case Status** | **Test Priority** |
| **Step** | **Expected** | **Actual** |
| 01 | **Browse System Videos** | Test whether the video is uploading or not | If video dataset is not uploaded | we cannot do further operations | If video is Dataset uploaded we will do further operations | High | High |
| 02 | **Start Webcam Video Tracking** | Verify the video is tracking the objects or not | Without tracking the object | We cannot do further operations | We Can  do further operations | High | High |

**7. SCREENSHOTS:**

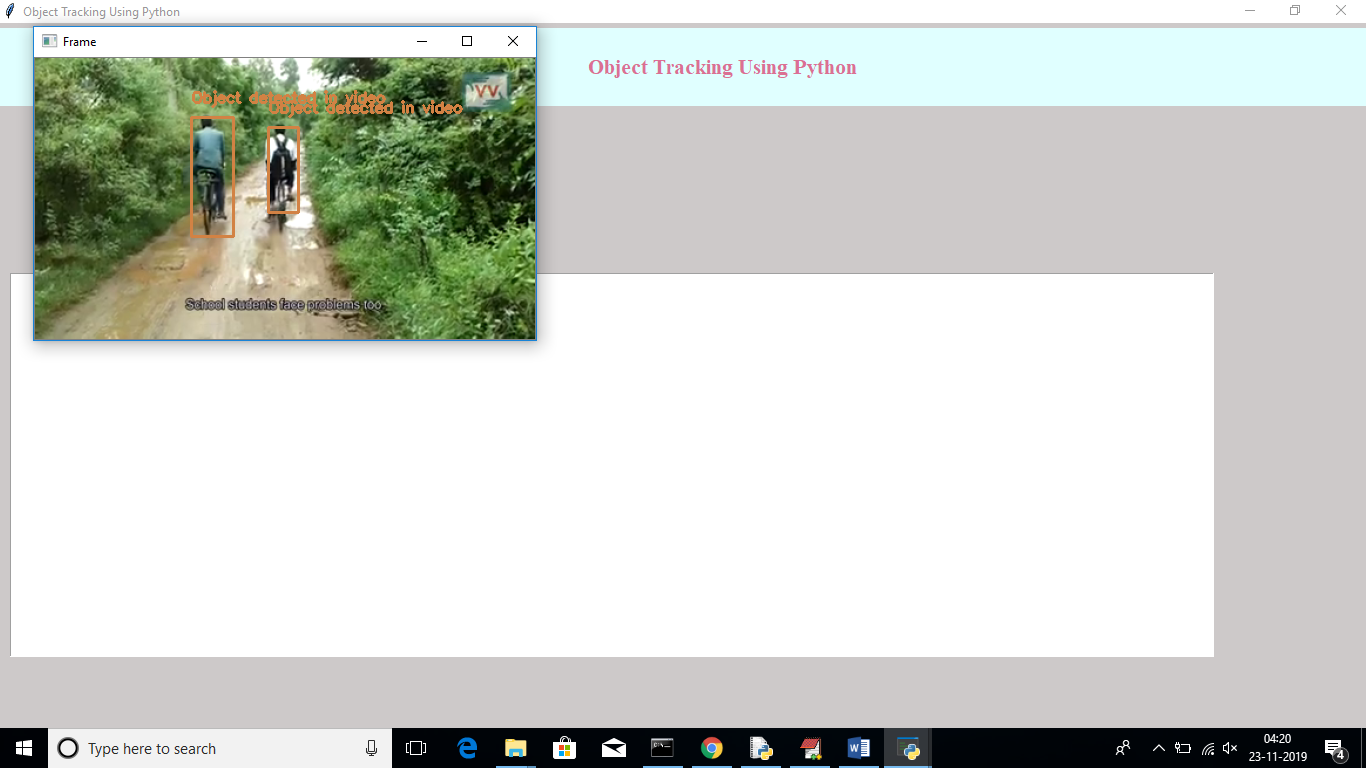
Double click on ‘run.bat’ file to get below screen



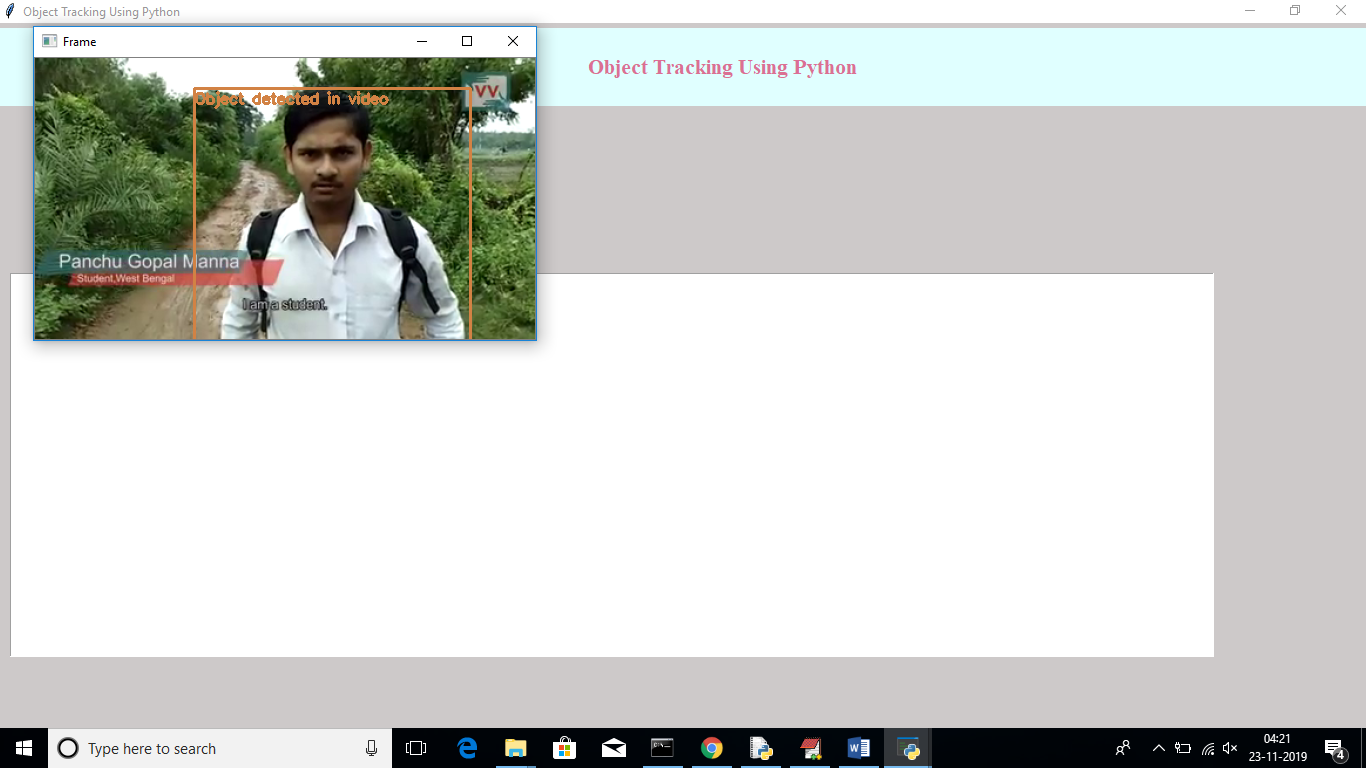
Now click on ‘Browse System Videos’ button to upload videos from system

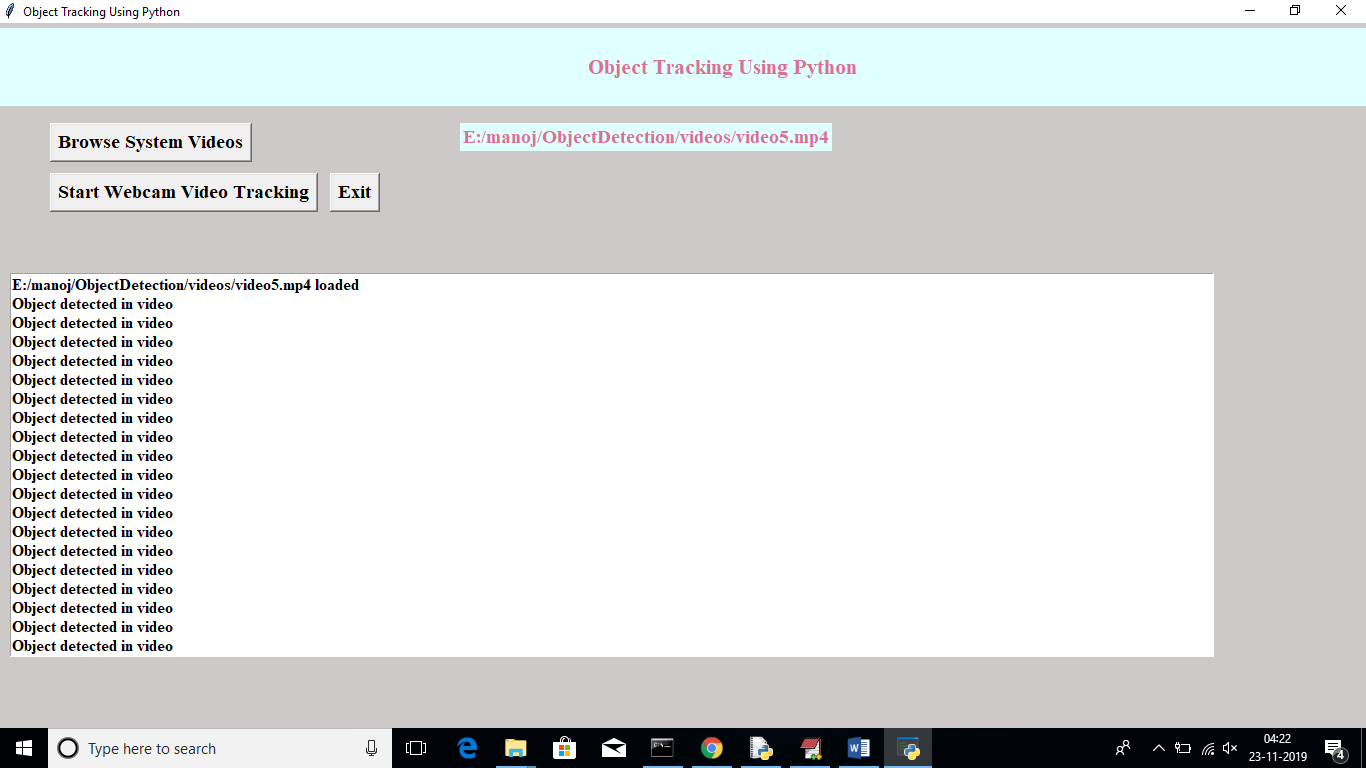


In above screen I am uploading one video, after upload will get below screen

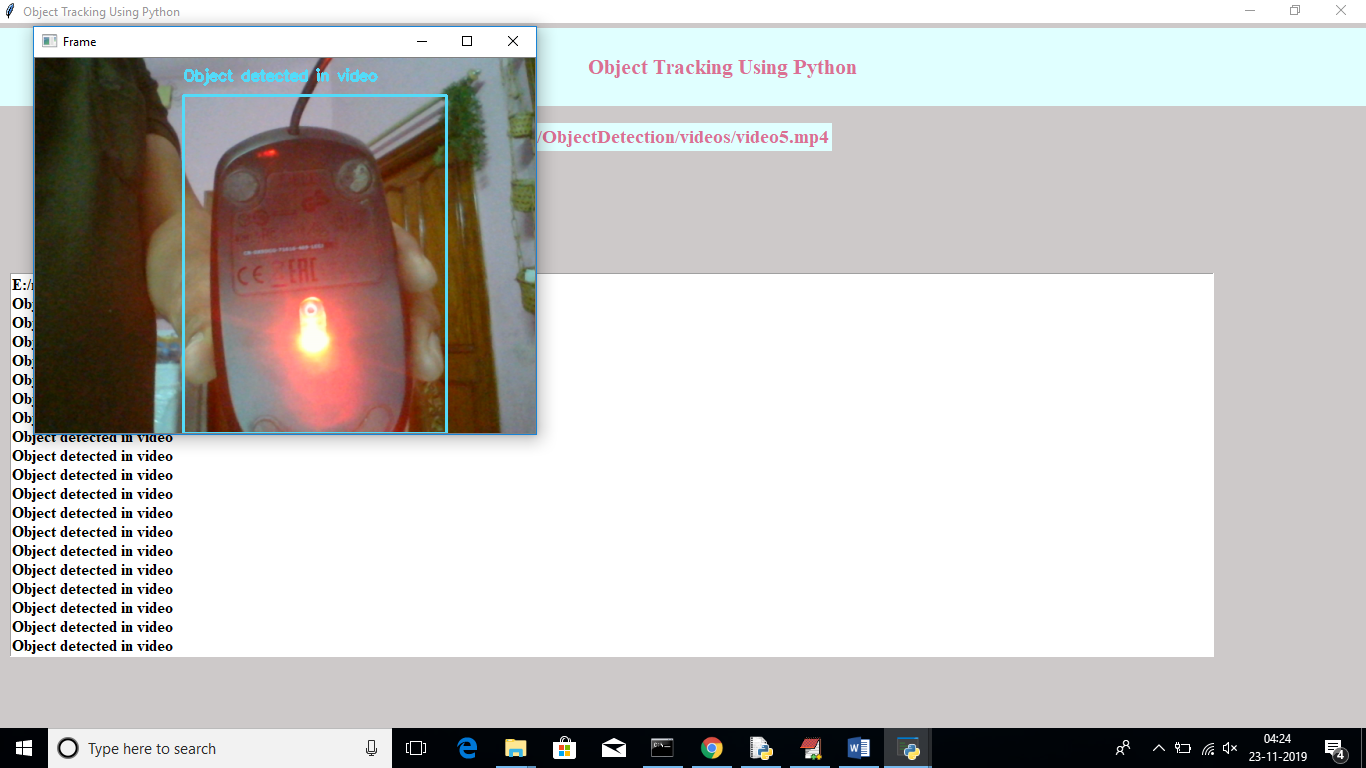


In above video we can see application start tracking objects from video and mark them with bounding boxes. Similarly we can upload any video and track objects from video

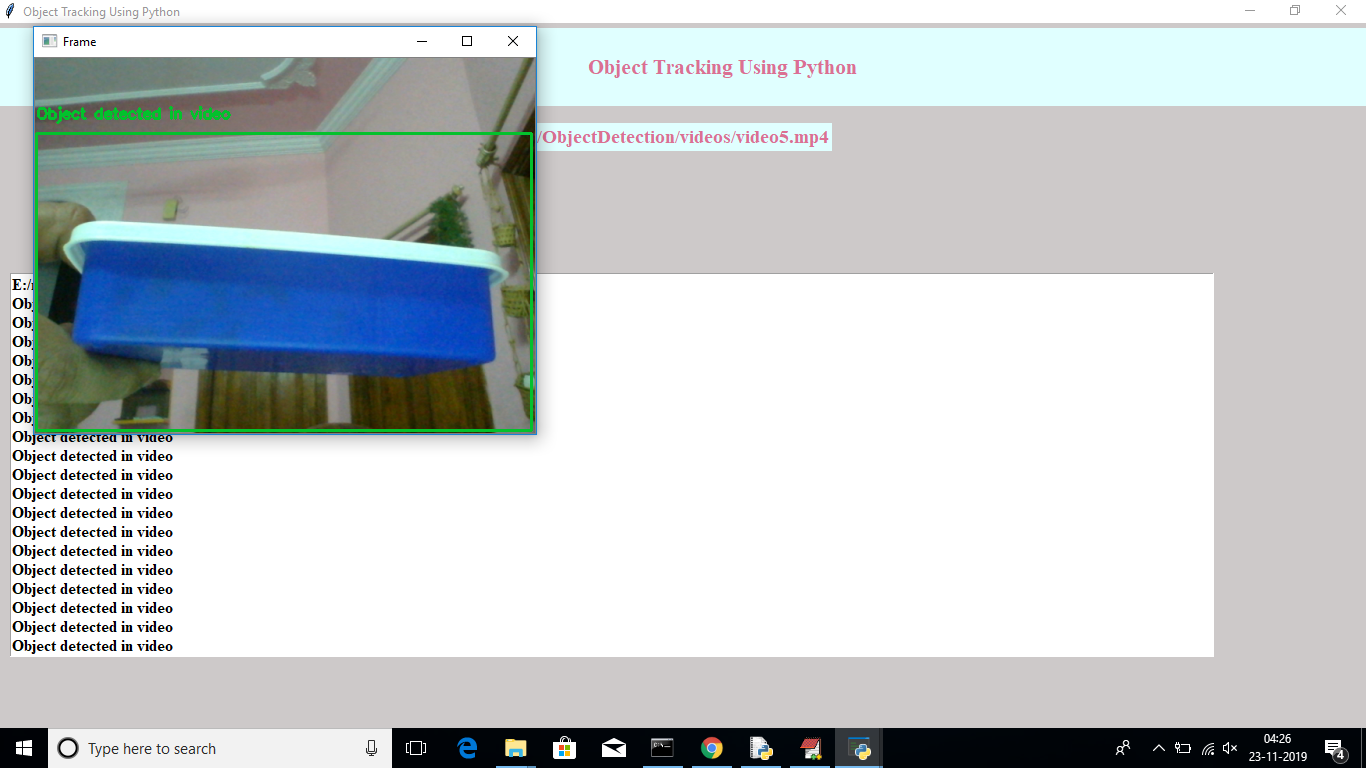




In above screen now click on another button called ‘Start Webcam Video Tracking’ to connect application to web cam and start streaming. After connecting to webcam will get below screen



In above screen we can see objects is getting tracked from webcam also. In above screen it track computer mouse from web cam video



**8. CONCLUSION**

The inclusion of Artificial Intelligence to solve Computer vision tasks has outperformed the image processing approaches of handling the tasks. The CNN model trained to on road vehicle dataset for single object detection, achieved a validation accuracy of 95.7 % for auto, 95.5% for car and 96 % for heavy vehicles for day images. The high validation accuracy is because of huge amount of data on which it is trained from each class. Performance metrics are tabulated for day, evening and NIR images. Multiple object detection is implemented using YOLOv3 for KITTI and COCO dataset. Performance metrics is tabulated for YOLOv3 on considered classes of images. Higher the precession value of class greater will be mAP value. The mAP value depends on image chosen for calculation. IoU of 0.5 is ideal for detection and tracking. mAP values can be enhanced by increasing true positive values. Results of performance metrics is totally dependent on image data set used. Further objects are detected in video based on region of interest. The performance measures measured such as speed and color of vehicle, type of vehicle, direction of vehicle movement and the number of vehicles in ROI. Multiple object tracking is implemented for traffic surveillance video using YOLOv3 and OpenCV. Multiple objects are detected and tracked on different frames of a video. Further training the models on powerful GPUs and by increasing the number of images evaluate the models on other datasets and modify the design if required to make the model more robust and suitable for real-time applications.

**9.REFERENCES**

[1] V. D. Nguyen et all., “Learning Framework for Robust Obstacle Detection, Recognition, and Tracking”, IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 6, pp. 1633-1646, June 2017.

[2] Zahraa Kain et all, “Detecting Abnormal Events in University Areas”, 2018 International conference on Computer and Applications(ICCA),pp. 260-264, 2018.

[3] P. Wang et all., “Detection of unwanted traffic congestion based on existing surveillance system using in freeway via a CNN-architecture trafficnet”, IEEE Conference on Industrial Electronics and Applications (ICIEA), Wuhan, 2018, pp. 1134-1139.

[4] Q. Mu, Y. Wei, Y. Liu and Z. Li, “The Research of Target Tracking Algorithm Based on an Improved PCANet”, 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, 2018, pp. 195-199.

[5] H. C. Baykara et all., “Real-Time Detection, Tracking and Classification of Multiple Moving Objects in UAV Videos”, 29th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), Boston, MA, 2017, pp. 945-950.

[6] W. Wang, M. Shi and W. Li, “Object Tracking with Shallow Convolution Feature”, 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, 2017, pp. 97-100.

[7] K. Muhammad et all., “Convolutional Neural Networks Based Fire Detection in Surveillance Videos”, IEEE Access, vol. 6, pp. 18174-18183, 2018.

[8] D. E. Hernandez et all., “Cell Tracking with Deep Learning and the Viterbi Algorithm”, International Conference on Manipulation, Automation and Robotics at Small Scales (MARSS), Nagoya, 2018, pp. 1-6.

[9] X. Qian et all., “An object tracking method using deep learning and adaptive particle filter for night fusion image”, 2017 International Conference on Progress in Informatics and Computing (PIC), Nanjing, 2017, pp. 138-142.

[10] Y. Yoon et all., “Online Multi-Object Tracking Using Selective Deep Appearance Matching”, IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia), Jeju, 2018, pp. 206-212.

[11] H. S. Bharadwaj, S. Biswas and K. R. Ramakrishnan. “A large scale dataset for classification of vehicles in urban traffic scenes”, Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing,ACM,2016.

[12] Mohana et.al., “Performance Evaluattion of background modeling methods for object Detection and Tracking”,International Conference on Inventive systems and Control (ICISC).

[13] G. Chandan et.al., “Real Time Object Detection and Tracking Using Deep Learning and OpenCV”, International Conference on Inventive Research in Computing Applications (ICIRCA), 2018.

[14] Mohana et.al., “Elegant and efficient algorithms for real time object detection, counting and classification for video surveillance applications from single fixed camera” International Conference on Circuits, Controls, Communications and Computing (I4C),2016.

[15] Mohana et.al., “Simulation of Object Detection Algorithms for Video Survillance Applications”, 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud),2018.