# **REAL TIME OBJECT DETECTION**

**(YOLO ALGORITHM)**

**A MINI PROJECT REPORT**

## ***Submitted by***

**SATHISH KUMAR M 212220220042**

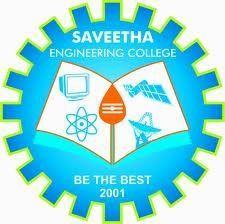
**VIGNESHKUMAR V 212220220054**

## **in partial fulfilment for the award of the degree of**

**BACHELOR OF ENGINEERING**

**IN**

**INFORMATION TECHNOLOGY**



**SAVEETHA ENGINEERING COLLEGE**

**ANNA UNIVERSITY: CHENNAI- 600 025**

**DECEMBER 2022**

## **ANNA UNIVERSITY : CHENNAI - 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report **“ REAL TIME OBJECT DETECTION”** is the bonafide work of “ **SATHISH KUMAR M -212220220042 , VIGNESHKUMAR V -212220220054** ” who carried out the project work under my supervision.

**SIGNATURE**      **SIGNATURE**

**Dr. K. Suresh Kumar,** **Dr.K.Suresh Kumar**

**Associate Professor** **Associate** **Professor**

**SUPERVISOR**  **HEAD OF THE DEPARTMENT**

Department of Information Department of Information Technology Technology

Saveetha Engineering College, Saveetha Engineering College, Thandalam, Chennai 602105. Thandalam, Chennai 602105.

DATE OF THE EXAMINATION: ……………………

**INTERNAL EXAMINER**     **EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

We express our deep sense of gratitude to our honourable and beloved Founder President **Dr. N. M. Veeraiyan**, our President **Dr. Saveetha Rajesh**, our Director **Dr. S. Rajesh** and other management members for providing the infrastructure needed.

We express our wholehearted gratitude to our **principal**, **Dr. N. Duraipandian**, for his wholehearted encouragement in completing this project.

We convey our thanks to **Dr. K .Suresh Kumar** , Professor and **Head of the Department** of Information Technology, Saveetha Engineering College, for his kind support and for providing necessary facilities to carry out the project work.

We would like to express our sincere thanks and deep sense of gratitude to our **Supervisor** **Dr. K. Suresh Kumar**, Associate Professor, Department of Information Technology , Saveetha Engineering College and our **project Coordinator**, **Dr. G . Nalini priya**, Professor, Department of Information Technology, Saveetha Engineering College for their valuable guidance, suggestions and constant encouragement that paved the way for the successful completion of the project work and for providing us necessary support and details at the right time and during the progressive reviews.

We owe our thanks to all the members of our college, faculty, staff and technicians for their kind and valuable cooperation during the course of the project. We are pleased to acknowledge our sincere thanks to our beloved parents, friends and well-wishers who encouraged us to complete this project successfully.

**ABSTRACT**

When we look at images or videos, we can easily locate and identify the objects of our interest within moments. Passing on this intelligence to computers is nothing but object detection - locating the object and identifying it. Object Detection has found its application in a wide variety of domains such as video surveillance, image retrieval systems, autonomous driving vehicles and many more. Various algorithms can be used for object detection but we will be focusing on the YoloV3 algorithm. YOLO stands for "You Only Look Once". The YOLO model is very accurate and allows us to detect the objects present in the frame. YOLO follows a completely different approach. Instead of selecting some regions, it applies a neural network to the entire image to predict bounding boxes and their probabilities. YOLO is a single deep convolutional neural network that splits the input image into a set of grid cells, so unlike image classification or face detection, each grid cell in YOLO algorithm will have an associated vector in the output that tells us if an object exists in that grid cell, the class of that object, the predicted bounding box for that object. The model here is progressive so it learns more over time, increasing its prediction accuracy over time. The way the model works is that it makes many predictions in one frame and decides to use the most accurate prediction, thus discarding the other. The predictions are made randomly, so if the model feels like there is an object in the frame which is of a very small pixel it will take that also into consideration. To make it more precise and clearer, the model simply creates bounding boxes around everything in the frame, it would make predictions for each box and pick the one with the most confidence score. All this is done in a small-time frame, thus showing why this specific model is the best to use in a real time situation.

Keywords : Object Detection, YOLO Algorithm, Prediction.

**Table of Contents**

| **Chapter Number** | **Title** | **Page Number** |
| --- | --- | --- |
|  | **Abstract** | 4 |
|  | **List of Figures** | 7 |
|  | **List of Abbreviations** | 8 |
| **1** | **INTRODUCTION** | **9** |
|  | 1.1 Overview of the project | 10 |
|  | 1.2 Motivation  1.3 Research AIM  1.4 Problem Description  1.5 Object Detection  1.6 Image Detection | 10  10  11  11  12 |
| **2** | **LITERATURE SURVEY** | **14** |
|  | 2.1 Introduction | 15 |
|  | 2.2 Literature Survey | 15 |
| **3** | **SYSTEM DESIGN** | **18** |
|  | 3.1 Feasibility Study | 19 |
|  | 3.2 Economical Feasibility | 19 |
|  | 3.3 Technical Feasibility | 19 |
|  | 3.4 Social Feasibility | 20 |
|  | 3.5 CNN | 20 |
|  | 3.6 RCNN | 21 |
|  | 3.7 fast R-CNN  3.8 Algorithm  3.9 Architecture Diagram  3.10 Data Flow Diagram  3.11 System Requirements | 21  22  24  24  25 |
| **4** | **IMPLEMENTATION AND ANALYSIS** | **26** |
|  | 4.1 Python Library | 27 |
|  | 4.2 Data | 28 |
| **5** | 4.3 Dataset Collections  4.4 Software Description  4.4.1 Python  4.4.2 Javascript  **Conclusion**  5.1 Conclusion  5.2 Future Enhancement  **Appendix**  **Sample Coding**  **Sample Output**  **References** | 29  29  29  30  31  32  32  35  35  44  46 |

**LIST OF FIGURES**

3.9. Architecture diagram

3.10. Data Flow Diagram

3.10. Activity Diagram

**LIST OF ABBREVIATIONS**

NumPy - Numerical Python

Pandas - Python Data Analysis Library

SciKit - SciPy Toolkit

AI - Artificial Intelligence

ML - Machine Learning

IDE - Integrated Development Environment

CNN - Convolutional Neural Network

RCCN - Recurrent Neural Network

YOLO - You only look once

GPU - Graphical Processing unit

**Chapter 1**

**INTRODUCTION**

## **OVERVIEW OF THE PROJECT**

In this chapter, we introduce the background and focus area of our thesis. We highlighted our motivations, scope, and thesis organization. We will explore YOLO’s potential in recognizing objects and creating bounding box around them.

**1.2 MOTIVATION**

Blind people do lead a normal life with their own style of doing things. But, they definitely face troubles due to inaccessible infrastructure and social challenges. The biggest challenge for a blind person, especially the one with the complete loss of vision, is to navigate around places. Obviously, blind people roam easily around their house without any help because they know the position of everything in the house. Blind people have a tough time finding objects around them. . So we decided to make a REAL TIME OBJECT DETECTION System. We are interested in this project after we went through few papers in this area. As a result we are highly motivated to develop a system that recognizes objects in the real time environment.

**1.3 RESEARCH AIM**

Real time object detection is a vast, vibrant and complex area of computer vision. If there is a single object to be detected in an image, it is known as Image Localization and if there are multiple objects in an image, then it is Object Detection. This detects the semantic objects of a class in digital images and videos. The applications of real time object detection include tracking objects, video surveillance, pedestrian detection, people counting, self-driving cars, face detection, ball tracking in sports and many more. Convolution Neural Networks is a representative tool of Deep learning to detect objects using Open CV (Open source Computer Vision), which is a library of programming functions mainly aimed at real time computer vision.

**1.4 PROBLEM DESCRIPTION**

The problem description of object detection is to determine where objects are located in a given image (object localization) and which category each object belongs to (object classification). Object detection is the problem of finding and classifying a variable number of objects on an image. The important difference is the “variable” part. In contrast with problems like classification, the output of object detection is variable in length, since the number of objects detected may change from image to image.

**1.5 OBJECT DETECTION**

Object detection is a computer vision implementation that makes a system (an algorithm) about to estimate the location of objects in a digitized scene such as an image or video.

Usually, a bounded box is wrapped around the detected object which helps humans locate the object quicker than unprocessed images. For this discourse, an object is the representation of a physical object (URL) in an image.

In image processing, it is an identifiable portion of an image that can be interpreted as a single unit. This creates a sharp contrast to the layman’s idea that an image or an object are interchangeable. Usually, an image may contain one or more objects, the discernibility of which is of upmost importance. For instance, in a single image the objects contained can range from a single unit to as many objects of as numbers, bordering on infinity.

Although “detection” could mean locating a hidden concealed object, detection may also mean the ability of an intelligence to signify the existence and identification of an object. The object in question does not have to be hidden. This later form is the form in which we based this thesis.

Fundamentally, two approaches to image detection exist, they are machine learning-based approaches and deep learning-based approaches [11]. In more traditional ML-based approaches, computer vision techniques are employed to analyze various features of an image, such as the color histogram or edges, to identify groups of pixels that may belong to an object. These features are then inputted into a regression model that predicts the location of the object along with its label.

Some Machine learning approaches are Viola–Jones object detection framework based on Haar features, Scale-invariant feature transform (SIFT) and Histogram of oriented gradients (HOG) features [10]. On the other hand, deep learning-based approaches employ convolutional neural networks (CNNs) to perform end-to-end, unsupervised object detection, in which features do not need to be defined and extracted separately

**1.6 IMAGE DETECTION**

Object detection is often confused with image recognition. A picture of a dog receives the label “dog”. A picture of two dogs, still receives the label “dog”. Object detection, on the other hand, draws a box around each dog and labels the box “dog”. The model predicts where each object is and what label should be applied. In that way, object detection provides more information about an image than recognition.

Recognition in this context is the ability of an intelligent system to identify an object based on certain similarities that it shares with another object that the intelligence has previously encountered. Recognition may be based on inference or relation, that is, a situation whereby an intelligence is able to recognize an object because it recognizes similarities in form and properties. Recognition may also occur because the Artificial intelligence has encountered the exact specimen at a previous instance.

In human beings’ recognition is a cognitive process that happens seamlessly and almost instantly without any hitch. The human brain is capable of learning and adapting information with minimal effort such that even humans that are still in the developmental stages of their existence can recognize objects and patterns easily. Humans can recognize a multitude of objects in images with little effort, even though the image of the objects may vary somewhat in different viewpoints, in many different sizes and scales or even when they are translated or rotated. Objects can even be recognized when they are partially obstructed from view.

Artificial intelligence, however, does not innately possess this cognitive ability. For Artificial Intelligence to acquire this level of skill they must acquire training. This training is usually acquired by ‘teaching’ the A.I. using coding, datasets, and databases. This task is still a challenge for computer vision systems given these A.I. systems need to be trained for each class of object it is meant to recognize

Object recognition is an extremely difficult computational problem. The core problem is that each object in the world can cast an infinite number of different 2-D images onto the retina as the object's position , pose, lighting, and background vary relative to the viewer. Yet the brain solves this problem effortlessly. Progress in understanding the brain's solution to object recognition requires the construction of artificial recognition systems that ultimately aim to emulate our own visual abilities, often with biological inspiration

**Chapter 2**

**LITERATURE**

**SURVEY**

# 

# **2.1 INTRODUCTION**

A literature survey or a literature review in a project report is that section which shows various analysis and research made in the field of your interest and the results already published, taking into account the various parameters of the project and the extent of project. Once the programmers start building the tool programmers need a lot of external support. This support can be obtained from senior programmers, books or from the websites. It is the most important part of your report as it gives you a direction in the area of your research. It helps you set a goal for your analysis - thus giving you your problem of statement. Literature survey is the most important sector in the software development process. Before developing the tools and the associated designing it is necessary to determine the survey the time factor, resource requirement, manpower, economy, and company strength.

**2.2. LITERATURE SURVEY**

* 2016 - joseph Redmon, Santosh Divvala, Ross Girshick, Ali FarhadI

A fast and simple approach to detecting real time images was introduced in this paper as You Only Look Once. The model was built to detect images accurately, fast and to differentiate between art and real images.In comparison with Object detection techniques that came before YOLO, like R-CNN, YOLO introduced a single unified architecture for regression go image into bounding boxes and finding class probabilities for each box. This meant that YOLO performed much faster and also provided more accuracy. It could also predict artwork correctly.

* 2018 - Chengji Liu, Yufan Tao, Jiawei Liang, Kai Li1, Yihang Chen

A generalized object detection network was developed by applying complex degradation processes on training sets like noise, blurring, rotating and cropping of images. The model was trained with the degraded training sets which resulted in better generalizing ability and higher robustness.The experiment showed that the model trained with the standard sets does not have good generalization ability for the degraded images and has poor robustness. Then the model was trained using degraded images which resulted in improved average precision. It was proved that the average precision for degraded images was better in general degenerative model compared to the standard model.

* 2018 - Wenbo Lan, Jianwu Dang, Yangping Wang, Song Wang

The network structure of YOLO algorithm is improved and a new network structure YOLO-R was proposed to increase the ability of the network to extract the information of the shallow pedestrian features by adding passthrough layers to the original YOLO network.The YOLO v2 and YOLO-R network models were tested on the test set of the INRIA data set. The experimental results show that the YOLO-R network model is superior to the original YOLO v2 network model. The number of detection frames reached 25 frames/s, basically meeting the requirement of realtime performance.

* 2018 - Rumin Zhang, Yifeng Yang

An obstacle detection algorithm in the indoor environment is proposed which combines the YOLO object detection algorithm and the light field camera and will classify objects into categories and mark them in the image.The images of the common obstacles were labeled and used for training YOLO. The object filter is applied to remove the unconcern obstacle. Different types of scene, including pedestrian, chairs, books and so on, are demonstrated to prove the effectiveness of this obstacle detection algorithm

* 2019 - Zhimin Mo1, Liding Chen1, Wen-jing You

A method for identifying the solder joints of automotive door panels based on YOLO algorithm that provides the type and location of solder joints in real time. For detecting the small solder joints more precisely, this paper adopts YOLO algorithm which adopts multi-level predictions, predicting on different size feature maps and combining the prediction results to obtain the final result.

The YOLO algorithm, proposed identifies the position of the solder joints accurately in real time. This is helpful to increase the efficiency of the production line and it has a great significance for the flexibility and real-time of the welding of automobile door panels.

**Chapter 3**

**SYSTEM DESIGN**

**3.1. Feasibility Study**

Object Detection is a study of Computer Vision Field. Object location is a huge exploration region in Computer Vision, can be applied to numerous applications, for example, Driver less vehicles, security, reconnaissance, machine examination, and so forth. Object Detection is utilized to distinguish the area of the object in a picture, Face detection, medical imaging, etc. Evolution of Deep learning have changed the old methods of object detection and tracking system. Computer Vision recognizes characteristics in pictures, Classifying Object in the picture, Classifying objects along with localization, drawing a bounding box around object Present in the picture, Object segmentation or semantic segmentation, Neural style Transfer. Deep learning strategies are the most grounded strategy for object detection.

**3.2. ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. Thus the developed system requires an IDE and person for data collection. The software is mostly open source and datasets can be collected from actors using the software.

**3.3. TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is the Technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. Thus the developed system requires an IDE such as Jupyter or google colab to evaluate datasets and collection of dataset for different environment conditions requires a person to follow the instructions.

**3.4. SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. Since the system deals with audio it might be a little difficult to make it socially acceptable and person can choose to be anonymous if they wish to.

**3.5. CNN (Convolutional Neural Network)**

The network comprises of five convolutional layers. It accepts input as a picture which is a 2D array of a pixel with RGB channel. Then Channels or elements indicator apply to the information picture and get yield highlights maps. Numerous convolutional are acted in lined up by applying the ReLU work. CNN works for just a single object at a time so it doesn't work successfully in different objects in an image. CNN turned into a decent norm for image classification after Kriszhevsky's CNN's performance. We can't recognize objects which are overlapping and various background and don't order these various objects yet in addition don't distinguish boundries, contrasts and relations in other.

**3.6. RCCN (Recurrent Neural Network)**

This network incorporates three principal parts, first is region extractor, second is feature extractor and last is classifier. It involves a selective search algorithms for object detection to create region proposal. Extricate 2000 small regions for each picture. Here 2000 convolutional networks utilized for every small regions of the pictures. So have one Convolutional network expected to handle RCNN different regions with CNN characteristics partitions the picture into a few regions. Run pictures through pre-prepared AlexNet lastly apply the SVM algorithm.

**3.7. FAST R-CNN**

This network is a superior adaptation of R-CNN which is presented by Ross Girshick. The article guarantees that Quick R-CNN multiple times quicker than past R-CNN which is nine times. Network select different sets arrangements of bounding boxes then use feature extractor by CNN network then, at that point, use classifier or regression for yield the class of each containers.

**3.8. Algorithm for Object Detection (YOLO Algorithm)**

YOLO Algorithm:YOLO is Abbrevation of (you only look once). Older Object detection algorithms utilize the districts grid parts to distinguish and identifies the objects however don't utilize the whole picture , a few regions might contain the objects . YOLO is an object detection algorithm entirely different from the district based algorithms seen previously. In YOLO a convolutional network predicts the bounding boxes and the class probabilities for these containers. It's challenging for everybody to contain the assets for the Deep Learning So that is where this Yolo came into the picture. Furthermore, bunches of pre-prepared models and datasets are accessible at this point.

YOLO Algorithm Process

YOLO stores the information in Vector Form:

YOLO = (pc, bx, by, bh, bw, c1, c2, c3),

Where pc characterizes the Probability and demonstrates in the event that object is available or not bx, by, bh, bw determines whether objects for the classes c1,c2,c3.

So on the off chance that there is any object concerning class c1, it will have the worth 1 generally 0.

It utilizes the non max suppression the bounding box with more exactness, precision is chosen and remaining are disregarded.

Equation for Non Max Suppression is :-

IoU = Area of the crossing point or interaction

**Advantages of Object Detection**

Object recognition can help you to automate the specific processes bringing competitive advantages to Retail, Healthcare, Manufacturing, Transportation and other industries. Exposit Machine Learning engineers have enhanced experience in solving complex business tasks by implementing smart software platforms.

**Disadvantage of Object Detection**

Object detection is customarily considered to be much harder than image classification, particularly because of these five challenges: dual priorities, speed, multiple scales, limited data, and class imbalance.

**PROS AND CONS**

Benefits of using YOLO :

● It is extremely fast compared to other real time detectors which came before it as it uses a Unified Model where the detection is seen as a single regression problem and there is no complex pipeline, just a neural network run on the image

. ● It makes less errors than Fast R-CNN as it can see the bigger context because YOLO, unlike Fast R-CNN, can globally reason the image when making predictions. YOLO sees the entire image and encodes some of the contextual information about all classes and their appearance.

● YOLO has learnt generalized representations of objects. YOLO successfully differentiates natural images against art work.

The limitations of YOLO are:

● Small object detection, such as a flock of birds, is a problem as there is a spatial restriction on bounding boxes with each cell being able to predict only two boxes and have one class.

● Problems when generalizing objects of abnormal aspect ratios and configurations.

● Loss function will treat the errors of small or large bounding boxes as same

**3.9. Architecture Diagram**

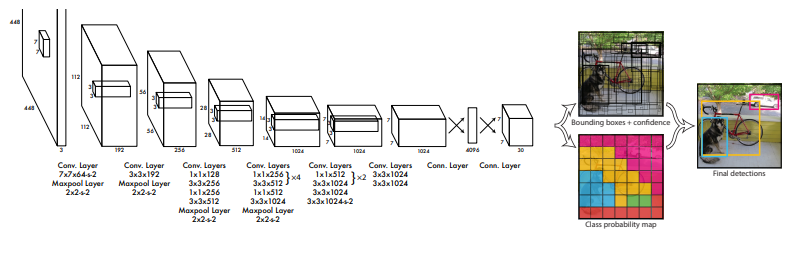
****

Figure 3.9

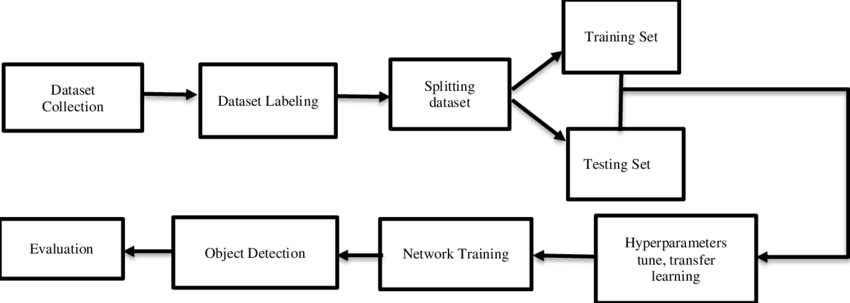
**3.10. Data Flow Diagram**

Figure 3.10

**3.10. Activity Diagram**

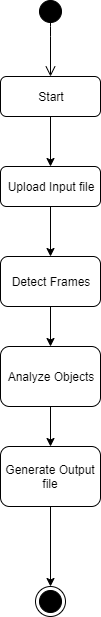
****

Figure 3.10

**3.11. System Requirements**

Software Specifications:

* Google Colab
* Google Drive

Hardware Specifications:

* Processor – Intel® Core™ i5-7200U CPU @ 2.50GHz 2.71GHz
* Hard disk drive – 238 GB
* RAM – 8.00 GB
* OS – Windows 10 Home ©2017 Microsoft corporation
* Keyboard, Mouse.

**Chapter 4**

**IMPLEMENTATION**

**AND**

**ANALYSIS**

**4.1. Python Library**

A Python library is a reusable chunk of code that you may want to include in your programs/ projects. The Python Standard Library is a collection of exact syntax, token, and semantics of Python. It comes bundled with core Python distribution. We mentioned this when we began with an introduction. For visualization of the dataset and prediction we use various python libraries.

Python libraries that are used in the project are:

* NumPy
* Pandas
* Matplotlib
* Scikit-learn

**NumPy:**

NumPy is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning. It is particularly useful for linear algebra, Fourier transform, and random number capabilities. High-end libraries like TensorFlow use NumPy internally for manipulation of Tensors.

**Pandas:**

Pandas is a popular Python library for data analysis. It is not directly related to Machine Learning. As we know that the dataset must be prepared before training. In this case, Pandas comes handy as it was developed specifically for data extraction and preparation. It provides high-level data structures and a wide variety of tools for data analysis. It provides many inbuilt methods for groping, combining and filtering data.

**Matplotlib:**

Matplotlib is used for data visualization. Like Pandas, it is not directly related to Machine Learning. It particularly comes in handy when a programmer wants to visualize the patterns in the data. It is a 2D plotting library used for creating 2D graphs and plots. A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc. It provides various kinds of graphs and plots for data visualization, viz., histogram, error charts, bar charts, etc.

**Scikit-learn:**

Scikit-learn is one of the extensively used ML libraries for classical ML algorithms. It is built on top of two basic Python libraries, viz., NumPy and SciPy. Scikit-learn supports most of the supervised and unsupervised learning algorithms. Scikit-learn can also be used for data-mining and data-analysis, which makes it a great tool for those starting out with ML.

**4.2. Data**

The crucial element in machine learning tasks for which a particular attention should be clearly taken is the data. Indeed, the results will be highly influenced by the data based on where did we find them, how are they formatted, are they consistent, is there any outlier and so on. At this step, many questions should be answered in order to guarantee that the learning algorithm will be efficient and accurate. Many sub steps are taken to get, clean and transform the data.

**4.3. Dataset Collection**

The first problem is to get the data to build a large enough dataset since we want to be able to predict and visualise. When it comes to Object detection databases it is difficult to get already done ones. Because of this we made a sample dataset just to make sure the ones we get from the web were not random. Then we did web scraping.

**4.4. Software Description**

**4.4.1. Python**

Python has huge set libraries which can be easily used for machine learning. Python is one of the languages which can be used to write codes in the Map-Reduce model while working in the Hadoop Ecosystem. Also, Spark, which is one of the new technologies for scalable big-data analysis, has a machine learning library in python. So, simplicity and wider applicability goes hand-in-hand to make it the so-called machine learning language. Python is an interpreter, high-level, general-purpose programming language. Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, it also has a comprehensive standard library. Python interpreters are available for many operating systems.

**4.4.2 JavaScript**

JavaScript is a programming language primarily used to create interactive webpages and applications. It is a scripting language that is used to add dynamic and interactive elements to webpages. JavaScript enables developers to create complex web applications that are more interactive and engaging. JavaScript can also be used to create mobile applications and desktop applications. JavaScript is a powerful and versatile language that can be used for a wide variety of tasks.

**Chapter 5**

**CONCLUSION**

**5.1. Conclusion**

As Object detection and recognition in todays world can be considered as one of the most challenging, complex and most important task in computer vision Field. As we know that this project is been developed with the underlying purpose of real-time object in pictures, videos captured streaming cameras or web cams.

**5.2. FUTURE ENHANCEMENT**

The object recognition system can be applied in the area of surveillance system, face recognition, fault detection, character recognition etc. The objective of this thesis is to develop an object recognition system to recognize the 2D and 3D objects in the image. The performance of the object recognition system depends on the features used and the classifier employed for recognition. This research work attempts to propose a novel feature extraction method for extracting global features and and obtaining local features from the region of interest. Also the research work attempts to hybrid the traditional classifiers to recognize the object. The object recognition system developed in this research was tested with the benchmark datasets like COIL100, Caltech 101, ETH80 and MNIST.

The object recognition system is implemented in MATLAB 7.5 It is important to mention the difficulties observed during the experimentation of the object recognition system due to several features present in the image. The research work suggests that the image is to be preprocessed and reduced to a size of 128 x 128. The proposed feature extraction method helps to select the important feature. To improve the efficiency of the classifier, the number of features should be less in number. Specifically, the contributions towards this research work are as follows,

An object recognition system is developed, that recognizes the two-dimensional and three dimensional objects.

* The feature extracted is sufficient for recognizing the object and marking the location of the object. x The proposed classifier is able to recognize the object in less computational cost.
* The proposed global feature extraction requires less time, compared to the traditional feature extraction method.
* The performance of the SVM-kNN is greater and promising when compared with the BPN and SVM.
* The performance of the One-against-One classifier is efficient.
* Global feature extracted from the local parts of the image.
* Local feature PCA-SIFT is computed from the blobs detected by the Hessian-Laplace detector.
* Along with the local features, the width and height of the object computed through projection method is used.

The methods presented for feature extraction and recognition are common and can be applied to any application that is relevant to object recognition. The proposed object recognition method combines the state-of-art classifier SVM and k-NN to recognize the objects in the image. The multiclass SVM is used to hybridize with the k-NN for the recognition. The feature extraction method proposed in this research work is efficient and provides unique information for the classifier.

The image is segmented into 16 parts, from each part the Hu’s Moment invariant is computed and it is converted into Eigen component. The local feature of the image is obtained by using the Hessian-Laplace detector. This helps to obtain the objects feature easily and mark the object location without much difficulty.

As a scope for future enhancement,

Features either the local or global used for recognition can be increased, to increase the efficiency of the object recognition system.

Geometric properties of the image can be included in the feature vector for recognition. 150

Using unsupervised classifier instead of a supervised classifier for recognition of the object.

The proposed object recognition system uses grey-scale image and discards the color information. The colour information in the image can be used for recognition of the object. Colour based object recognition plays vital role in Robotics Although the visual tracking algorithm proposed here is robust in many of the conditions, it can be made more robust by eliminating some of the limitations as listed below:

In the Single Visual tracking, the size of the template remains fixed for tracking. If the size of the object reduces with the time, the background becomes more dominant than the object being tracked. In this case the object may not be tracked.

Fully occluded object cannot be tracked and considered as a new object in the next frame.

Foreground object extraction depends on the binary segmentation which is carried out by applying threshold techniques. So blob extraction and tracking depends on the threshold value.

**APPENDIX**

**Sample Coding**

from IPython.display import display, Javascript, Image

from google.colab.output import eval\_js

from google.colab.patches import cv2\_imshow

from base64 import b64decode, b64encode

import cv2

import numpy as np

import PIL

import io

import html

import time

import matplotlib.pyplot as plt

%matplotlib inline

from darknet import \*

network, class\_names, class\_colors = load\_network("cfg/yolov4-csp.cfg", "cfg/coco.data", "yolov4-csp.weights")

width = network\_width(network)

height = network\_height(network)

def darknet\_helper(img, width, height):

darknet\_image = make\_image(width, height, 3)

img\_rgb = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

img\_resized = cv2.resize(img\_rgb, (width, height), interpolation=cv2.INTER\_LINEAR)

img\_height, img\_width, \_ = img.shape

width\_ratio = img\_width/width

height\_ratio = img\_height/height

copy\_image\_from\_bytes(darknet\_image, img\_resized.tobytes())

detections = detect\_image(network, class\_names, darknet\_image)

free\_image(darknet\_image)

return detections, width\_ratio, height\_ratio

image = cv2.imread("data/person.jpg")

detections, width\_ratio, height\_ratio = darknet\_helper(image, width, height)

for label, confidence, bbox in detections:

left, top, right, bottom = bbox2points(bbox)

left, top, right, bottom = int(left \* width\_ratio), int(top \* height\_ratio), int(right \* width\_ratio), int(bottom \* height\_ratio)

cv2.rectangle(image, (left, top), (right, bottom), class\_colors[label], 2)

cv2.putText(image, "{} [{:.2f}]".format(label, float(confidence)),

(left, top - 5), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5,

class\_colors[label], 2)

cv2\_imshow(image)

def js\_to\_image(js\_reply):

image\_bytes = b64decode(js\_reply.split(',')[1])

jpg\_as\_np = np.frombuffer(image\_bytes, dtype=np.uint8)

img = cv2.imdecode(jpg\_as\_np, flags=1)

return img

def bbox\_to\_bytes(bbox\_array):

bbox\_PIL = PIL.Image.fromarray(bbox\_array, 'RGBA')

iobuf = io.BytesIO()

bbox\_PIL.save(iobuf, format='png')

bbox\_bytes = 'data:image/png;base64,{}'.format((str(b64encode(iobuf.getvalue()), 'utf-8')))

return bbox\_bytes

def take\_photo(filename='photo.jpg', quality=0.8):

js = Javascript('''

async function takePhoto(quality) {

const div = document.createElement('div');

const capture = document.createElement('button');

capture.textContent = 'Capture';

div.appendChild(capture);

const video = document.createElement('video');

video.style.display = 'block';

const stream = await navigator.mediaDevices.getUserMedia({video: true});

document.body.appendChild(div);

div.appendChild(video);

video.srcObject = stream;

await video.play();

// Resize the output to fit the video element.

google.colab.output.setIframeHeight(document.documentElement.scrollHeight, true);

// Wait for Capture to be clicked.

await new Promise((resolve) => capture.onclick = resolve);

const canvas = document.createElement('canvas');

canvas.width = video.videoWidth;

canvas.height = video.videoHeight;

canvas.getContext('2d').drawImage(video, 0, 0);

stream.getVideoTracks()[0].stop();

div.remove();

return canvas.toDataURL('image/jpeg', quality);

}

''')

display(js)

data = eval\_js('takePhoto({})'.format(quality))

img = js\_to\_image(data)

detections, width\_ratio, height\_ratio = darknet\_helper(img, width, height)

for label, confidence, bbox in detections:

left, top, right, bottom = bbox2points(bbox)

left, top, right, bottom = int(left \* width\_ratio), int(top \* height\_ratio), int(right \* width\_ratio), int(bottom \* height\_ratio)

cv2.rectangle(img, (left, top), (right, bottom), class\_colors[label], 2)

cv2.putText(img, "{} [{:.2f}]".format(label, float(confidence)),

(left, top - 5), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5,

class\_colors[label], 2)

cv2.imwrite(filename, img)

return filename

try:

filename = take\_photo('photo.jpg')

print('Saved to {}'.format(filename))

display(Image(filename))

except Exception as err:

print(str(err))

def video\_stream():

js = Javascript('''

var video;

var div = null;

var stream;

var captureCanvas;

var imgElement;

var labelElement;

var pendingResolve = null;

var shutdown = false;

function removeDom() {

stream.getVideoTracks()[0].stop();

video.remove();

div.remove();

video = null;

div = null;

stream = null;

imgElement = null;

captureCanvas = null;

labelElement = null;

}

function onAnimationFrame() {

if (!shutdown) {

window.requestAnimationFrame(onAnimationFrame);

}

if (pendingResolve) {

var result = "";

if (!shutdown) {

captureCanvas.getContext('2d').drawImage(video, 0, 0, 640, 480);

result = captureCanvas.toDataURL('image/jpeg', 0.8)

}

var lp = pendingResolve;

pendingResolve = null;

lp(result);

}

}

async function createDom() {

if (div !== null) {

return stream;

}

div = document.createElement('div');

div.style.border = '2px solid black';

div.style.padding = '3px';

div.style.width = '100%';

div.style.maxWidth = '600px';

document.body.appendChild(div);

const modelOut = document.createElement('div');

modelOut.innerHTML = "<span>Status:</span>";

labelElement = document.createElement('span');

labelElement.innerText = 'No data';

labelElement.style.fontWeight = 'bold';

modelOut.appendChild(labelElement);

div.appendChild(modelOut);

video = document.createElement('video');

video.style.display = 'block';

video.width = div.clientWidth - 6;

video.setAttribute('playsinline', '');

video.onclick = () => { shutdown = true; };

stream = await navigator.mediaDevices.getUserMedia(

{video: { facingMode: "environment"}});

div.appendChild(video);

imgElement = document.createElement('img');

imgElement.style.position = 'absolute';

imgElement.style.zIndex = 1;

imgElement.onclick = () => { shutdown = true; };

div.appendChild(imgElement);

const instruction = document.createElement('div');

instruction.innerHTML =

'<span style="color: red; font-weight: bold;">' +

'When finished, click here or on the video to stop this demo</span>';

div.appendChild(instruction);

instruction.onclick = () => { shutdown = true; };

video.srcObject = stream;

await video.play();

captureCanvas = document.createElement('canvas');

captureCanvas.width = 640; //video.videoWidth;

captureCanvas.height = 480; //video.videoHeight;

window.requestAnimationFrame(onAnimationFrame);

return stream;

}

async function stream\_frame(label, imgData) {

if (shutdown) {

removeDom();

shutdown = false;

return '';

}

var preCreate = Date.now();

stream = await createDom();

var preShow = Date.now();

if (label != "") {

labelElement.innerHTML = label;

}

if (imgData != "") {

var videoRect = video.getClientRects()[0];

imgElement.style.top = videoRect.top + "px";

imgElement.style.left = videoRect.left + "px";

imgElement.style.width = videoRect.width + "px";

imgElement.style.height = videoRect.height + "px";

imgElement.src = imgData;

}

var preCapture = Date.now();

var result = await new Promise(function(resolve, reject) {

pendingResolve = resolve;

});

shutdown = false;

return {'create': preShow - preCreate,

'show': preCapture - preShow,

'capture': Date.now() - preCapture,

'img': result};

}

''')

display(js)

def video\_frame(label, bbox):

data = eval\_js('stream\_frame("{}", "{}")'.format(label, bbox))

return data

video\_stream()

label\_html = 'Capturing...'

bbox = ''

count = 0

while True:

js\_reply = video\_frame(label\_html, bbox)

if not js\_reply:

break

frame = js\_to\_image(js\_reply["img"])

bbox\_array = np.zeros([480,640,4], dtype=np.uint8)

detections, width\_ratio, height\_ratio = darknet\_helper(frame, width, height)

for label, confidence, bbox in detections:

left, top, right, bottom = bbox2points(bbox)

left, top, right, bottom = int(left \* width\_ratio), int(top \* height\_ratio), int(right \* width\_ratio), int(bottom \* height\_ratio)

bbox\_array = cv2.rectangle(bbox\_array, (left, top), (right, bottom), class\_colors[label], 2)

bbox\_array = cv2.putText(bbox\_array, "{} [{:.2f}]".format(label, float(confidence)),

(left, top - 5), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5,

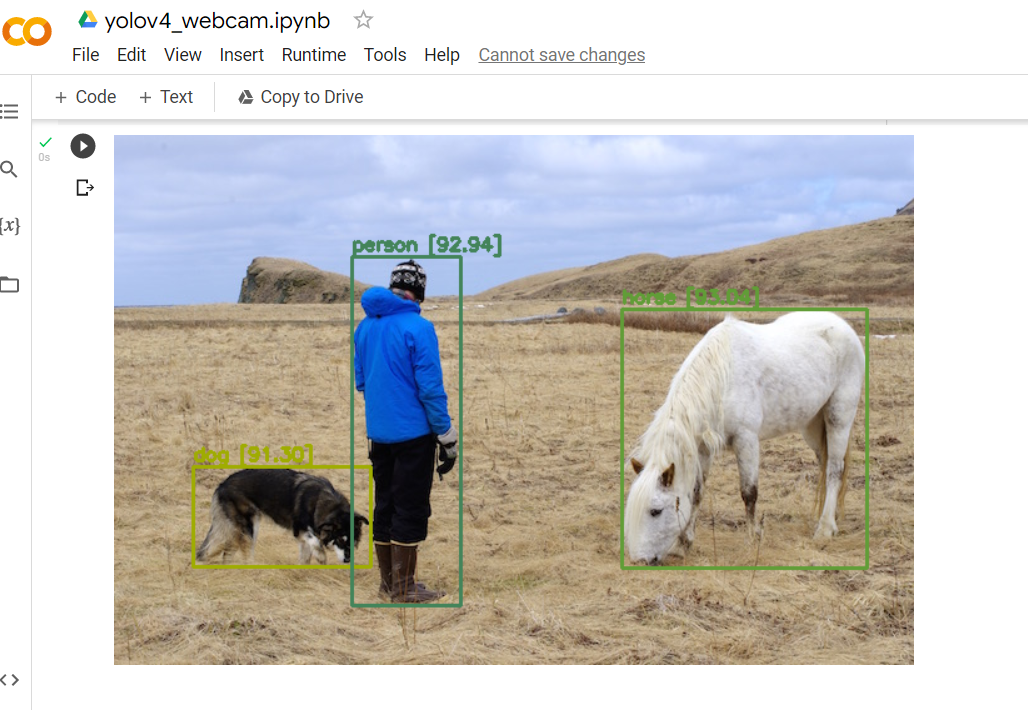
class\_colors[label], 2)

bbox\_array[:,:,3] = (bbox\_array.max(axis = 2) > 0 ).astype(int) \* 255

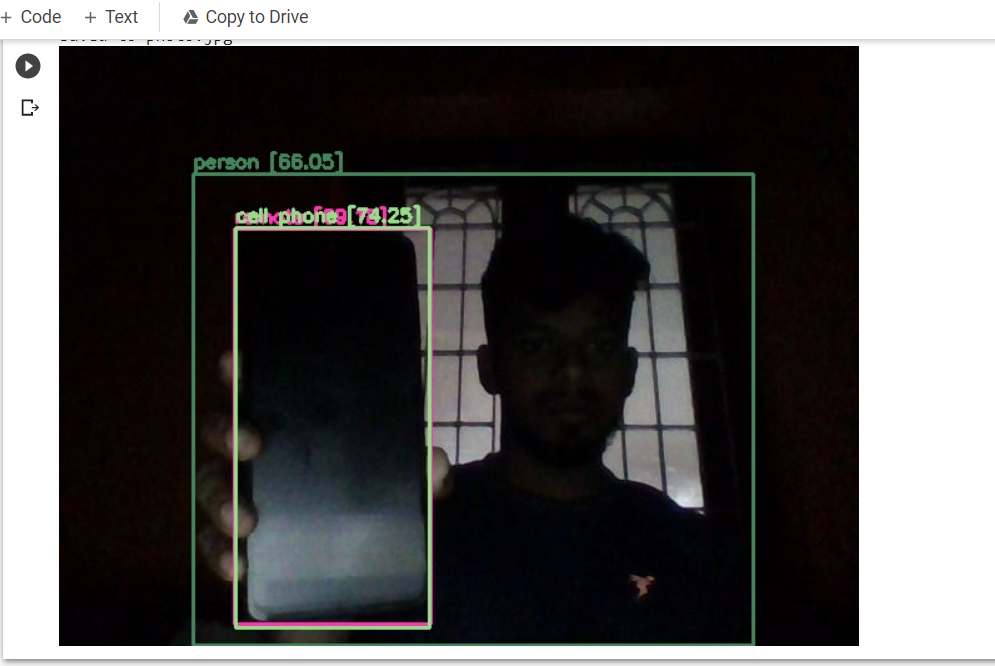
bbox\_bytes = bbox\_to\_bytes(bbox\_array)

bbox = bbox\_bytes

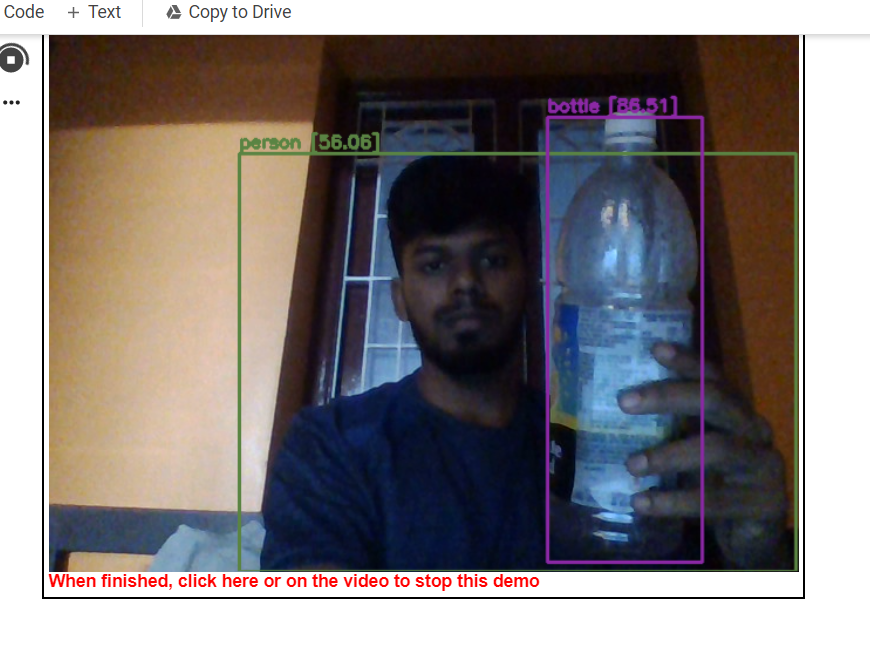
**SAMPLE OUTPUT**

****

**1. saved Image**

****

**2.captured object detection**

****

**3. Live object detection**

**REFERENCES**

1. Joseph Redmon,Santosh Divvala,Ross Girshick,Ali Farhadi. ”You Only Look Once: Uniﬁed, Real-Time Object Detection” [J].2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),2016:779788.
2. Chengji Liu ,Yufan Tao ,Jiawei Liang ,Kai Li1 ,Yihang Chen, ”Object Detection Based on YOLO Network” 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC 2018)
3. Wenbo Lan, Jianwu Dang, Yangping Wang and Song Wang, “Pedestrian Detection Based on YOLO Network Model” 978-1-5386-60751/18/$31.00 ©2018 IEEE
4. Rumin Zhang, Yifeng Yang, ”An Algorithm for Obstacle Detection based on YOLO and Light Filed Camera”, 2018 Twelfth International Conference on Sensing Technology (ICST)
5. Zhimin Mo1, Liding Chen1, Wenjing You1 ”Identification and Detection of Automotive Door Panel Solder Joints based on YOLO” 978-128101064/19$31.00 ©2019IEEE
6. M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision, 111(1):98–136, Jan. 2015.
7. Software Architecture in Practice, 3rd Addison-Wesley

Professional ©2012, ISBN:0321815734 9780321815736

1. Joseph Redmond, Ali Farhadi, ”YOLOv3: An Incremental

Improvement”, University of Washington