Real Time Object Detection Using Yolo: CNN Model

Benedict Joseph Johnson
Presidency College, Bengaluru, India

Veera Nagaiah Maddikayala
Presidency College, Bengaluru, India

Abstract

Object detection and depth estimation have been among the core objects that are often employed for application in autonomous navigation, and scene understanding, but the techniques available suffer from issues such as occlusion and accuracy problems, which make this infeasible. In attempt to resolve these issues, a new approach is thus proposed wherein deep learning architectures are integrated with efficient computation methods. Through the fusion of the state-of-the-art object detection model with a sophisticated depth estimation network, our proposal brings both accuracy and precision improvements radically. It would then be that while taking care of all the existing shortcomings, our proposed model should have furthered the envelope from being purely real-time while creating even more capabilities in the fields of object detection and depth estimation. It is used in a better version of YOLO, specifically the version 4 algorithm, for better object identification and detection purposes in creating a distance estimation model based upon pixel data through a linear regression approach.. Simulations have shown the effectiveness of integrating YOLOV4 with the distance estimation model, which provides promising results in terms of distance estimation for the detected objects. It started by applying the You Only Look Once (YOLO) model towards real-time detection of objects, which was achieved through division of the grid and generation of the bounding box regarding class probabilities.

Keywords
Object detection Convolutionalneuralnetworks YOLO
Deep learning Computer vision

I. INTRODUCTION (*HEADING 1*)

INTRODUCTION TO OBJECT DETECTION AND DEPTH ESTIMATION

Recently, the field of computer vision has undergone a dramatic development, and one of the most significant fields within that research domain is object detection and recognition. Knowing how to detect and recognize objects automatically in images and videos has important implications for many industries, including autonomous systems, surveillance, and human-computer interaction. This will be an integration work of You Only Look Once (YOLO) v4 into a pixel-based distance estimation model utilizing linear regression methodology. This project title, "Integrating You Only Look Once version 4 (YOLOV4) and Distance Estimation for Enhanced Object Detection," aptly describes the work that wants to focus on improving object detection systems in terms of accuracy, efficiency, and situational awareness.,

II. LITERARTURE REVIEW

The first method applied to measure distance is Distance Estimation Network, also known as DisNet. It is a type of machine learning setup, and it provides the obstacle detection system with a method to estimate how far away in real-life measurements the object viewed with the camera is from the monocular camera. In particular, preliminary results of a still ongoing research are reported, making it possible to allow the on-board multisensory system being developed within the framework of H2020 Shift2Rail project Synchronized Multi-Antenna Radar and 3D imaging Technology (SMART) to autonomously learn distances to objects, possible obstacles on rail tracks ahead of the locomotive. A distance estimation system is proposed based on the Multi Hidden-Layer Neural Network. It was named DisNet and learned to predict the distance of an object from the camera sensor. The DisNet was trained through the supervised learning method, which made use of

III. METHODOLOGY:

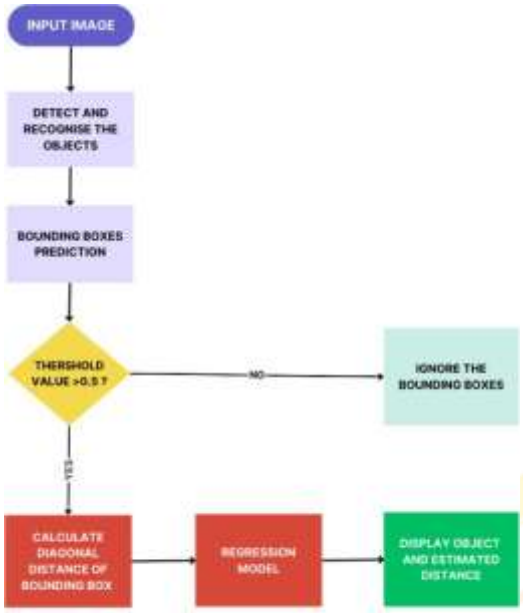


Figure 1.1: Methodology

As seen in Fig.1.1, the methodology adopted to perform this project is systematic; it involves data collection of different image datasets with humans, motorbikes, cars, and buses. YOLOV4 algorithm deep learning framework would be used in the system, and the model will be trained on the gathered datasets for efficient object detection and recognition. Concurrently, a pixel data-based and linear regression-based distance estimation model will be developed. Finally, the YOLOV4 model and the distance estimation model will be interfaced together to form an integrated system. Extensive simulations over the datasets and the real-world test cases will be conducted to measure the performance of the integrated system in terms of accuracy, efficiency, and within real time.

I. Model Building

1. Implementation of YOLO v4 model:

n this chapter, details are to be mentioned on how the implementation of a YOLO model, that is to be deployed for real-time object detection and distance estimation purposes, has been done using Python and OpenCV. The YOLO model YOLOV4-Tiny has been utilized to detect objects

2. Convolutional Neural Networks (CNNs) for Depth Prediction:

CNNs proved to be one of the most dominant architectures for most computer vision tasks, such as depth estimation. CNNs were built upon the backbone idea of parsing through grid-like data such as images and learned to extract hierarchical features with fantastic performance. In the depth map, each pixel indicates the distance that the point is from the camera in the scene. To train a CNN on depth estimation, one needs to have many image-depth pairs. In these pairs, ground truth depth maps are provided with images. In training, a CNN learns to minimize the difference between its predicted depth maps and those of the ground truth depth maps using stochastic gradient descent and other optimization algorithms. One of the most prevailing schemes applying CNNs toward depth estimation is monocular depth estimation, where just one image is used during prediction in the presence of depth. In such a case, the learning by the CNNs toward inferring depth from monocular cues is a deeply challenging task due to ambiguity in such monocular settings. However, through learning from a rich and massive dataset, CNNs can discern complex depth patterns and generalize well to unknown scenes. Another type of the application of stereo images in depth estimation is the use of Siamese CNN architectures on both left and right images simultaneously to predict disparities, which are further converted to depth values. Applications of depth estimation from CNN have achieved very promising results: from robots to autonomous driving, augmented reality, and scene understanding.

3. Depth Estimation from Stereo Vision:

Stereo vision is a key technique in computer vision-analogous to human binocular vision-that employs two cameras in order to estimate depth from images. The technique exploits epipolar geometry that inter-links camera views and calculates corresponding points in images. Disparities between corresponding points yield depth cues, subsequently resolved into 3D data through triangulation using camera baseline and into two types: local and global. The local methods include SAD, SSD, which estimates the pixel intensity around each pixel and determines the

correlation coefficient in the case of Normalized Cross-Correlation. Global methods take disparity information into account. Optimization-based Semi-Global Matching handles smoothness and occlusions carefully by maintaining proper control over boundaries. Graph Cuts and Belief Propagation minimize energy and further refine the disparities. Feature-based methods like SIFT or SURF provide distinctive keypoints. Block matching divides the images into small segments and compares them. An algorithm selection depends on needs, resources, and what kind of accuracy is required.

4.Monocular Depth Estimation:

Monocular depth estimation is the process of inferring depth information from a single image, which is a challenging task due to the lack of stereo or multiple viewpoints. Traditional computer vision techniques, as well as more recent deep learning approaches, have been used for monocular depth estimation. In this section, the exploration of depth estimation from single images will take a closer look at feature-based techniques like structure from motion and visual odometry.

5. Linear Regression Model for Depth Estimation:

Linear Regression is one of the most fundamental and widely used statistical methods for modeling any variable based on several variables. In connection with depth estimation, it can be utilized in some image features or cues to predict the depth of any scene. The idea is to establish a linear relation between features generated from the input image and their corresponding depth values. As in the case of linear regression, this model of regression tries to make the best fitting line possible by minimizing the difference between the predicted values and the actual ones for the values of depth. It can be represented by the following linear equation:

Depth = w0 + w1 · Feature1 + w2 · Feature2 +. .. + wn · FeatureN\

Here:

• Depth is the predicted value of depth. Feature1, Feature2, .FeaturesN: image features; w0, w1, w2, .wn: coefficients to learn through training process.

Application of this linear equation 3.1 to depth estimation requires an earlier step of extracting informative image features that would probably reveal insights into depth. These are gradients, textures, colors, among others with which there might be a correlation to depth. Taking these features as inputs, the linear regression model is trained using a set of paired images and their corresponding depths for learning the best coefficients, w0, w1, w2, ., wn

Hardware utilization:

1.Camera

In executing the project, a camera was used as theprimary tool for taking detailed 2D images of the scene to be targeted. More specifically, it uses a high-resolution 64-megapixel mobile camera. Thecamera plays two roles in the workflow of the project: first, in generating a captured dataset **2.Computer**

The core of the computer or laptop is considered to be the heart of the working module of the project; it serves as a processor and a computing center. This unit mainly executes several very important functions in relation to accepting the YOLOV4 algorithm as an input for proper objectdetection, intricate depth estimation calculations, and the generation of the final results.

3.DisplayInterface

The manner in which the processed image will appear is assisted by what can be best termed as a display interface. Such an interface forms part of what is considered the architecture of the project. The form of display interface that can take the form of monitors or screens is more critical given the fact that it provides for fine outputs from the processes involved in object detection and depth estimation visualization of the scene, thus allowing a better understanding and analysis of the system's performance and result.

Software Utilization:

The software elements, therefore, are used in the project in implementing the basic functions of object detection and depth estimation. The following provides an overview of the major software elements used and the role they play in the project.

1.iVCamSoftware

The project takes advantage of the skills of iVCamsoftware, which is a crucial part in bridging the gap

between the camera and the interface of the display. This enables the live feed taken by the camera to be transferred smoothly to the display interface on which they are expected to appear through a Wi-Fi connection. This software provides for real-time visualization of the images processed and plays a critical role in dynamic monitoring and assessment during testing and deployment. The successful setting up of a wireless link between the mobile camera and the display enables the achievement of the project's central objective, that is, the efficient and comprehensive integration of the system.

2.VisualStudioCode:

Visual Studio Code - The primary IDE for the project is Visual Studio Code. It has provided a very extensive, flexible platform for coding, debugging, and managing the system. Along with code editing, version control, and extensibility through a huge range of extensions, it makes the entire development and maintenance of the project very smooth. The project team collaboratively writes, optimizes, and debugs the responsible codebase for the execution of YOLOV4 algorithm, depth estimation calculations, and the orchestration of the whole system using VS Code. The UI and robust functionality of the IDE play a very significant role in the effective processing of the project. This brings about streamlined software development and continuous refinement. suite of extensions into smooth development and maintenance of the project.

2.Implementation of the YOLO v4 model:

In this section, detailed description of the real-time object detection and distance estimation implementation using Python, OpenCV is explained for the YOLO model. The YOLO model employed is YOLOV4-tiny, specifically designed to detect objects in a video stream, and the steps outlined below detail the implementation process: YOLO Model Setup and Configuration: The YOLOV4-tiny model is selected in the implementation of this project due to its efficiency in real-time object detection. The following components are needed to be loaded into the script: model weights, the configuration file, and class labels to obtain the YOLO model initializing the object detection.

3.Object Detection and Distance Estimation: Another function is specifically designed to detect objects and compute distances for the detected objects. The function takes in the YOLO model output stream, calculates the diagonal in pixels distance, and then, by use of pre-implemented class-specific equations, estimates more precise distances.

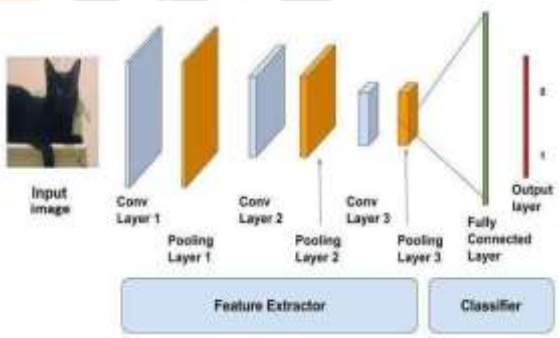
Real-time Video Feed and Processing The script starts capturing the video feed coming from the camera and starts processing every frame by calling the function. Real time processing permits the system to track objects and come up with their distances as it demonstrates a live graphical output.

4.Regressionmodels:

A two-step process is undertaken to implement a regression model for purposes of distance estimation. First, the model is trained whereby data will be used to facilitate the acquisition of an equation. The acquired equation will now be used in estimating the distances by using the length of the pixel of the diagonal box that was detected. This regression model provides a model relating pixel length to distance whereby distances can be estimated.

5.Generationofdataset:

Upon reaching this point, the process involves taking images to obtain a dataset of images. The dataset comprised a range of 3 meters to 15 meters that was made up of photographs taken at regular intervals of 1 meter.



III. PROBLEM STATEMENT

object detection estimation this includes specifying a dedication function for object detection and estimating distance for objects detected. when the appropriate output is retrieved from the YOLO model, the function will calculate the diagonal distance in pixels and further estimate the distance with predefined equations based on class.

Real-time video feed and processing :

The script then starts video capture from the camera feed, with each frame being processes by the function invocation. Real-time processing enable the system to detect objects and estimate distances base on class.

Figure 4.1 depicts a car positioned at a distance of 10 meters from the camera. Images of the car were taken at every meter from3 to 15m.

Figure 4.2: Person at 5m distance

Bounding box detected using YOLO is shown in Figure 4.3. The diagonal length of the bounding box (in pixels) is noted down. This systematic procedure is applied to each image in the dataset,

Implementing Linear Regression model:

The process of curve fitting is employed to obtain linear equations for each object.

$$y = (-0.0443 * x) + 15.71 \quad (4.1)$$

$$y = (-0.07415 * x) + 15.27 \quad (4.2)$$

$$y = (-0.05782 * x) + 15.75 \quad (4.3)$$

$$y = (-0.021 * x) + 13.3 \quad (4.4)$$

Table 4.1: Data Samples

Distance (in m)	Car (In pixels)	Motorbike (in pixels)	Person(in pixels)	Bus(in pixels)
3	330	193	300	580
4	274	165	210	403
5	216	126	160	310
6	182	98	140	267
7	152	86	128	230
8	136	76	110	200
9	127	67	100	175
10	110	60	86	165
11	102	55	75	155
12	93	48	71	145
13	85	40	65	127
14	81	43	60	120
15	74	40	53	105

Implementing Polynomial Regression model:

The process of curve fitting is employed to obtain polynomial equations for each object.

$$y = 1045.48 \cdot x - 0.97 - 0.47 \quad (4.5)$$

$$y = 421.59 \cdot x - 0.90 - 0.51 \quad (4.6)$$

$$y = 231.47 \cdot x - 0.62 - 4.39 \quad (4.7)$$

$$y = 25045.2 \cdot x - 1.57 + 2.02 \quad (4.8)$$

Using the given data in Table 4.1 ,the process of curve fitting is performed to derive the polynomial equations for the subsequent objects: car(4.5), motorbike(4.6), person(4.7), and bus(4.8).

The polynomial equation for the respective object detected is used by the program. The pixel length of the bounding box is taken as input for the equation and distance is estimated in real time

V. RESULT:

This chapter will proceed to closely scrutinize a comprehensive result of the outcome obtained from both the Linear regression model and Polynomial regression model, which was implemented above. This will give insights into how well and practicallyeach model can be used. Advantages of polynomial regression model over linear regression model:

1. Average Improvement in Accuracy Polynomial regression fits the curve much more flexibly than linear regression. This means that there is a greater ability to capture more meaningful portions of non-

linear relationships between variables. Which, in turn may improve average accuracy if relationships between such variables aren't linear.
2. Lower Error Rate Polynomial regression is preferable when the relationship between the y and x variables is complex and nonlinear. Therefore, it is likely to help decrease high error rates that may result from linear models that do not accurately represent the pattern of behavior in the data.
3. Distance-related errors: Polynomial regression can cope with non-stationary rates of change in data for varying distances. If the rate of errors deteriorates sharply towards smaller distances or is smaller towards mid-distance values, the flexibility of polynomial regression may enable it to fit the data in these specific regions.

VI. CONCLUSION

The project involves improvement in object recognition through the latest computer vision methods. The main aim here is to use YOLOV4 Algorithm for real-time object detection, recognizing, and locating the objects within a chosen environment. In addition to that, the project deals with depth estimation. For this purpose, the proposed approach will involve different Regression models to achieve an accurate measurement of the distance between objects at the camera., which enabled real-time object detection by splitting images into the grid and generating rectangles with the related class probabilities. It was further enhanced with an inventive depth estimation methodology. After such object detection, the diagonal pixel lengths of bounding boxes were extracted very cautiously and combined with the actual depths from the dataset. Detailed analysis has been made, using linear and polynomial regression techniques.
The project clearly demonstrated the capability of YOLOV4 to accurately recognize various objects and hence it is a good foundation for real-time object identification. Linear regression model resulted in accuracy of 80.96%, but an average accuracy of 98% for polynomial regression; hence the comparison between the two regressions shows that accuracy is better in polynomial regression than in linear regression, where it appropriately captures the high-order nonlinear relations of pixel length with the depth of an object. This discovery served to emphasize the importance of accounting for complex interactions when making depth estimates from vision.

This project starts on generic object detection pipelines which provide base architectures for other related tasks. With that in mind, the three most common tasks are usually done by aid provided by it: object detection, face detection, and pedestrian detection. Authors achieved this by combing 2 things: Object detection with deep learning and OpenCV and Efficient, threaded video streams with OpenCV. The camera sensor noise and lightening condition can change the result because it can create problem in recognizing the objects. generally, this whole process requires GPU's rather than CPU's. But we've done using CPU's and executes in much less time, making it efficient. Object Detection algorithms is a combination of both the image classification and object localization. It receives input in the form of image while its output is the bounding boxes that are as precise as the number of objects in the actual image along with the category label associated with every bounding box at the top. It tends to describe the case of the bounding box up the shape of position, height, and width.

VII. REFERENCES

[1] Geethapriya S, N. Duraimurugan, S.P. Chokkalingam, "Real-Time Object Detection with Yolo", International Journal of Engineering and Advanced Technology (IJEAT)
[2] Abdul Vahab, Maruti S Naik, Prasanna G Raikar an Prasad S R4, "Applications of Object Detection System", International Research Journal of Engineering and Technology (IRJET)
[3] Hammad Naeem, Jawad Ahmad and Muhammad Tayyab, "Real-Time Object Detection and Tracking", IEEE
[4] Meera M K, & Shajee Mohan B S. 2016, "Object recognition in images", International Conference on Information Science (ICIS)
[5] Astha Gautam, Anjana Kumari, Pankaj Singh: "The Concept of Object Recognition", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 3, March 2015
[6] V. Gajjar, A. Gurnani and Y. Khandhediya, "Human Detection and Tracking for Video