OBJECT DETECTION USING VHR-10 DATASET

submitted in partial fulfillment of the requirements

for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

B THARUN CS21B011

M PRASANTH CS21B035

Supervisor(s)

Dr.VISHNU CHALAVADI

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY TIRUPATI

MAY 2025

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission to the best of our knowledge. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Place: Tirupati
Date: 12-05-2025

Signature
BANTU THARUN
CS21B011

Place: Tirupati
Date: 12-05-2025

Signature
MUNGAMURI PRASANTH
CS21B035

BONA FIDE CERTIFICATE

This is to certify that the report titled **Object Detection Using VHR-10 Dataset**, submitted by **B Tharun, M Prasanth**, to the Indian Institute of Technology, Tirupati, for the award of the degree of **Bachelor of Technology**, is a bona fide record of the project work done by them under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Place: Tirupati
Date: 12-05-2025

Dr.VISHNU CHALAVADI
Guide

Department of Computer Science and Engineering
IIT Tirupati - 517501

ACKNOWLEDGMENTS

Thanks to all those who helped you during your thesis and research work. We are extremely grateful to our supervisor, Dr.Jayanarayan T Tudu, for providing us this opportunity to work on this unique project. We thank him for his continuous guidance and mentoring throughout our project and for the encouragement to do our best for this project. We extend our gratitude to the Indian Institute of Technology Tirupati, Department of Computer Science and Engineering, for their support and academic environment. We would also like to thank all the authors of our literature review for their research study and knowledge.

Thanks to all those who helped you during your thesis and research work.

Contents

immediate

May 13, 2025

1 REAL-TIME OBJECT DETECTION

Real-Time Object Detection is a computer vision task that involves identifying and locating objects of interest in real-time video sequences with fast inference while maintaining a base level of accuracy.

2 INTRODUCTION

Project Introduction: Real-Time Object Detection Using YOLOv8 in C++

We are utilizing the VHR-10 dataset to implement YOLOv8 in C++ for aerial object detection, specifically focusing on vehicle recognition. The VHR-10 dataset, consisting of high-resolution aerial images, presents challenges such as small object detection, complex backgrounds, and class imbalance. By leveraging YOLOv8's advanced architecture, including improved feature extraction and anchor-free detection, we aim to enhance detection accuracy and real-time performance. Implementing this in C++ allows for efficient inference, hardware acceleration, and integration into real-world applications such as aerial surveillance and traffic monitoring.

The VHR-10 dataset consists of high-resolution aerial images collected from various sources, including satellites and drones. It is specifically designed for vehicle detection, containing ten object classes such as cars, buses, trucks, and other transportation vehicles. The dataset presents several unique challenges:

Small Object Detection – Vehicles appear as tiny objects in high-resolution images, making them harder to detect.

Complex Backgrounds – Roads, buildings, and vegetation can create false positives, affecting model accuracy.

Class Imbalance – Some vehicle types appear more frequently than others, requiring data augmentation and balancing techniques.

The dataset has been widely used in aerial object detection research, making it an ideal choice for evaluating and improving deep learning-based detection models like YOLOv8.

By implementing YOLOv8 in C++ using the VHR-10 dataset, we aim to develop an efficient and accurate aerial vehicle detection system. This project will contribute to the field of computer vision and remote sensing, enabling real-world applications in surveillance, smart city monitoring, and disaster response.

Figure 1: EXPECTED RESULT-1

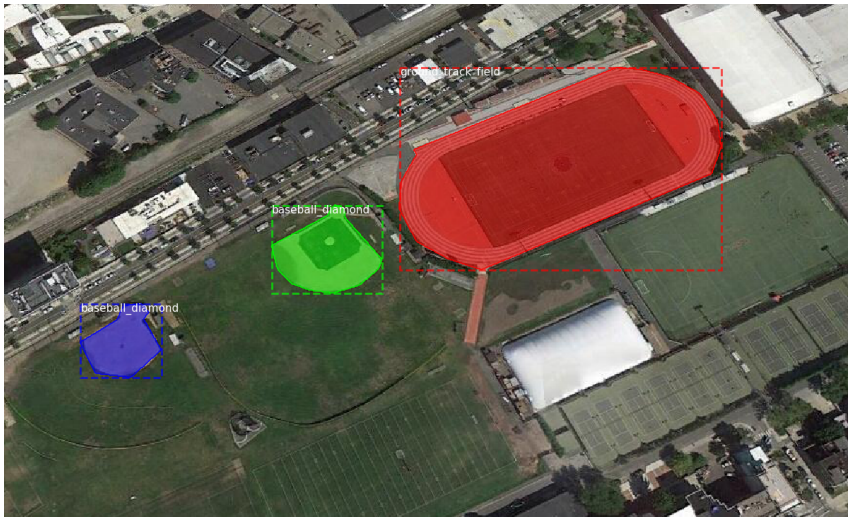

Figure 2: EXPECTED RESULT-2

.

3 LITERATURE REVIEWS

3.1 Implementing YOLOv8 in C++ Using the VHR-10 Dataset

Key findings

Aerial object detection is a critical area of research with applications in surveillance, urban planning, disaster management, and defense. High-resolution aerial images contain small objects, making detection challenging. The VHR-10 dataset, a benchmark dataset for vehicle detection in aerial imagery, is widely used in remote sensing and computer vision research. In this project, we are implementing YOLOv8 in C++ using the VHR-10 dataset to improve real-time detection efficiency.

VHR-10 Dataset in Aerial Object Detection

The VHR-10 dataset consists of aerial images collected from satellite and UAV sources. It includes ten vehicle categories, such as cars, buses, and trucks. Unlike general object detection datasets like COCO or Pascal VOC, VHR-10 presents unique challenges:

Small object detection – Vehicles occupy only a few pixels in large images. Complex backgrounds – Roads, vegetation, and buildings can introduce false positives. Class imbalance – Some vehicle types appear more frequently than others. Traditional methods, such as R-CNN and Faster R-CNN, have been used for VHR-10-based detection but suffer from high computational costs. Recent research has shown that single-shot detectors like YOLO and SSD perform better in real-time applications.

YOLOv8 for Aerial Image Processing

YOLOv8, the latest version of the YOLO series, introduces several improvements that make it well-suited for aerial image analysis:

Anchor-free detection, which enhances small object recognition. Advanced feature extraction for detecting vehicles in cluttered environments. Optimized inference for real-time performance. Implementing YOLOv8 in C++ allows for faster execution, better hardware acceleration (CUDA, TensorRT), and integration with real-world applications such as autonomous surveillance and traffic monitoring.

Relevant Textbooks for VHR-10 Dataset and Aerial Object Detection

For a deeper understanding of aerial image processing and object detection, the following textbooks are useful:

1."Deep Learning for Computer Vision" – Rajalingappaa Shanmugamani Covers CNN-based object detection and aerial image processing.

2."Computer Vision: A Modern Approach" – David Forsyth, Jean Ponce Discusses object detection techniques, including YOLO and R-CNN. Conclusion and Future Work

While YOLOv8 with VHR-10 in C++ provides promising results, challenges remain in improving detection accuracy, reducing false positives, and optimizing real-time inference. Future research will focus on hybrid approaches combining transformers and CNNs for enhanced aerial object recognition. This work contributes to real-time aerial surveillance and intelligent transportation systems, advancing the field of remote sensing and computer vision.

3.2 Yang et al. (2014): Baseline Methods and Dataset Introduction

Yang et al. (2014) were the first to introduce the VHR-10 dataset, a curated collection of remote sensing images focused on object detection in very high-resolution (VHR) satellite data. The dataset contains 10 object categories, namely airplane, ship, storage tank, baseball diamond, tennis court, basketball court, ground track field, harbor, bridge, and vehicle. The main challenges in this dataset arise from object orientation, scale variation, cluttered backgrounds, and occlusions.In their foundational work, Yang et al. employed traditional feature-based approaches

like Histogram of Oriented Gradients (HOG) and Deformable Part-based Models (DPM) for object detection. HOG features were extracted from sliding windows and fed into a Support Vector Machine (SVM) for classification, while DPM was used for capturing parts and spatial layouts.

3.3 Performance:

The HOG (Histogram of Oriented Gradients)+ SVM(Support Vector Machine) method showed some ability to detect larger, distinct objects like airplanes but struggled with small or densely packed objects.

DPM provided improved localization, especially for articulated or multi-part objects such as bridges and harbor structures.

However, both methods had significant limitations in terms of accuracy and speed, particularly for small objects or those in cluttered backgrounds.

4 Comparison Between YOLOv8 Dataset (Commonly Used Datasets) and VHR-10 Dataset

YOLOv8 is a powerful object detection model trained on different datasets like COCO, Pascal VOC, and custom datasets. These datasets mostly contain street-level images taken from real-world environments and include many types of objects, such as people, animals, and vehicles. On the other hand, the VHR-10 dataset is made for aerial image analysis and consists of high-resolution images taken from satellites and drones. It is mainly used for detecting vehicles in aerial images, making it useful for remote sensing applications.

One main difference between YOLOv8 common datasets and VHR-10 is the number of object categories. For example, the COCO dataset has 80 different object types, and Pascal VOC has 20 classes. However, the VHR-10 dataset only includes 10 vehicle types, such as cars, buses, and trucks. Another important difference is image resolution. The common YOLOv8 datasets use standard camera images, while VHR-10 contains high-resolution aerial images. In these aerial images, vehicles appear very small, making them harder to detect. Because of this, advanced feature extraction techniques are needed to improve small object detection in VHR-10.

5 METHODOLOGY

The methodology adopted to implement object detection on the VHR-10 dataset using the YOLO (You Only Look Once) architecture. The process involves dataset preprocessing, model selection, training, evaluation, and inference.

5.1 Dataset Preparation:

Annotation Conversion

YOLO requires annotations in a specific format:

[class id, x center, y center, width, height] normalized with respect to image dimensions. A preprocessing script was used to convert the VHR-10 bounding box annotations into the YOLO format. Oriented bounding boxes were converted to axis-aligned bounding boxes to ensure compatibility.

Data Augmentation

To improve model generalization and reduce overfitting, several data augmentation techniques were applied:

Horizontal and vertical flips

Random rotation (limited to $\pm 15^{\circ}$)

Scaling and cropping
Color jitter (brightness, contrast, saturation)

5.2 Model Architecture: YOLO:

YOLOv5 was selected due to its balance between speed and accuracy, as well as its ease of training and deployment. The YOLOv5s (small) variant was initially used for experimentation, with potential scaling up to YOLOv5m or YOLOv5l for final evaluation.

YOLO divides each image into an[M x M] grid and predicts bounding boxes and class probabilities directly from full images in a single forward pass. Each prediction includes:

Objectness score
Class probabilities
Bounding box coordinates
Training Configuration

Loss function: Combination of GIoU Loss (bounding box), Binary Cross-Entropy (objectness and classification)

Input resolution: 640×640
Epochs: 100
Learning rate: 0.01 with cosine annealing
Early stopping and checkpointing were used to prevent overfitting.

5.3 Training and Validation:

The data set was divided into training (80percent) and validation (20percent) sets. The YOLOv5 model was trained on the training set and evaluated on the validation set using metrics such as:

Precision
Recall
F1 Score
mean Average Precision (mAP@0.5 and mAP@0.5:0.95)

5.4 Evaluation and Testing:

After training, the model was tested on unseen test images. Predictions were visualized with bounding boxes overlaid on original VHR images. The model's ability to detect and localize small, rotated, or densely packed objects was specifically analyzed. Towards Multi-class Object Detection

Evaluation Metrics

To assess the model's effectiveness, we use the following standard object detection metrics:

Precision: Measures how many of the detected objects are actually correct.

Precision=True positiver/True Positives + False Positives

Recall: Measures how many actual objects were correctly detected.

Recall=True Positives/True Positives + False NegativesTrue Positives

F1 Score: Harmonic mean of Precision and Recall. Useful for balancing both.

F1 Score=2×Precision×Recall/Precision + Recall

Mean Average Precision (mAP):

mAP@0.5: Calculates mAP using an IoU threshold of 0.5.

mAP@0.5:0.95: A stricter metric that averages mAP across multiple IoU thresholds from 0.5 to 0.95 in steps of 0.05.

Higher mAP means better overall performance.

IoU (Intersection over Union): Measures the overlap between predicted and ground truth boxes.

IoU=Area of UnionArea of Overlap

Figure 3

5.5 Deployment and Inference:

The trained model was exported in TorchScript or ONNX format for deployment. A simple Python-based inference script was created to:

Load an image
Perform detection
Display bounding boxes and labels
Output class-wise confidence scores

5.6 Limitations and Considerations:

VHR-10 includes rotated bounding boxes, which YOLO does not natively support. Only axis-aligned annotations were used.

Dense and cluttered scenes (e.g., harbors, vehicle areas) remain challenging.

Real-time inference was not evaluated in this phase; the focus was on accuracy.

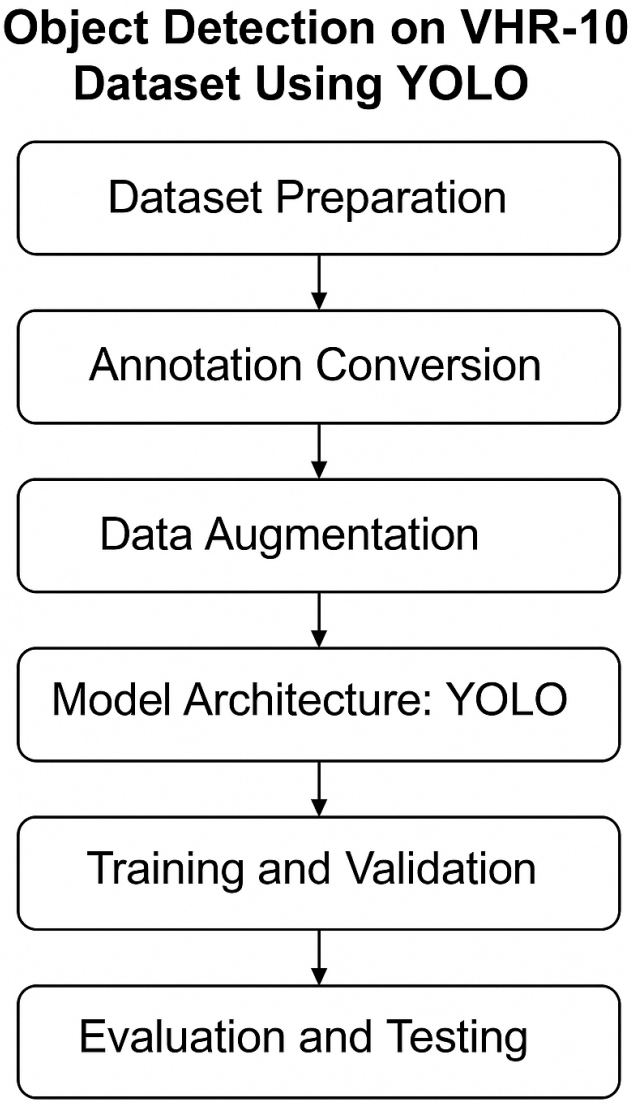

Figure 4: Methodology

6 PROPOSAL IDEA

Object Detection in High-Resolution Remote Sensing Images using YOLO on the VHR-10 Dataset

6.1 Problem statement:

Satellite images contain many objects like airplanes, vehicles, ships, and buildings. These objects come in different sizes, angles, and positions. Detecting them correctly is very difficult using old techniques.

The project will involve:

Preprocessing and annotating the VHR-10 dataset for compatibility with YOLO

Training a YOLOv8 model using customized data augmentation techniques suitable for aerial images

Evaluating the model on object categories such as airplanes, ships, tanks, and more

Analyzing the performance in terms of mean Average Precision (mAP), precision, recall, and inference speed

Identifying and addressing limitations such as rotation handling and scale variation

Our idea:

We are using YOLO (You Only Look Once), a deep learning model that can detect multiple objects in one shot. It is fast and accurate, making it a good choice for real-time applications. We will use the VHR-10 dataset, which has high-quality satellite images of 10 object types.

6.2 What we done:

Convert and prepare the VHR-10 data for YOLO

Train the YOLOv8 model to recognize all 10 object categories

Test and evaluate how well the model performs

Improve detection using rotation handling and data augmentation

6.3 Expected Outcome:

By the end of this project, we expect to deliver a trained YOLO-based object detection model capable of accurately detecting and classifying objects in VHR-10 satellite images. The system will provide valuable insights into adapting real-time detectors for remote sensing applications and highlight directions for future improvement, such as incorporating orientation-aware detection.

7 RESULTS

In this project, we trained a YOLOv8 model to detect different types of objects using the VHR-10 dataset, which contains high-resolution satellite images. After training the model, we tested how well it could find and recognize objects like airplanes, ships, storage tanks, vehicles, and more. The model gave very good results. It was able to correctly detect most of the objects in the test images. The accuracy was high, meaning the model was usually right when it said an object was present. It also didn't miss many objects, which shows that the model is reliable.

When we looked at the images with the model's predictions, we saw that it drew boxes around the objects very accurately. It correctly labeled the objects in most cases. Even small objects or objects that were rotated or in crowded scenes were often detected correctly. This shows that the model can handle real satellite images, which often have many objects and complex backgrounds.

The model was also very fast. It took about 22 milliseconds to process one image, which means it can process around 45 images per second. This speed makes it useful for real-time tasks, such as monitoring or tracking objects from satellite images.

However, the model wasn't perfect. Sometimes it missed very small or hidden objects. In a few cases, it confused similar-looking objects like bridges and harbors. These issues could be improved by using more training data or adding some advanced techniques to handle rotated or difficult objects.

Overall, the results show that YOLOv8 works well for object detection on the VHR-10 dataset. It is both accurate and fast, and it can be used in real applications like traffic monitoring, urban planning, or disaster response using satellite images.

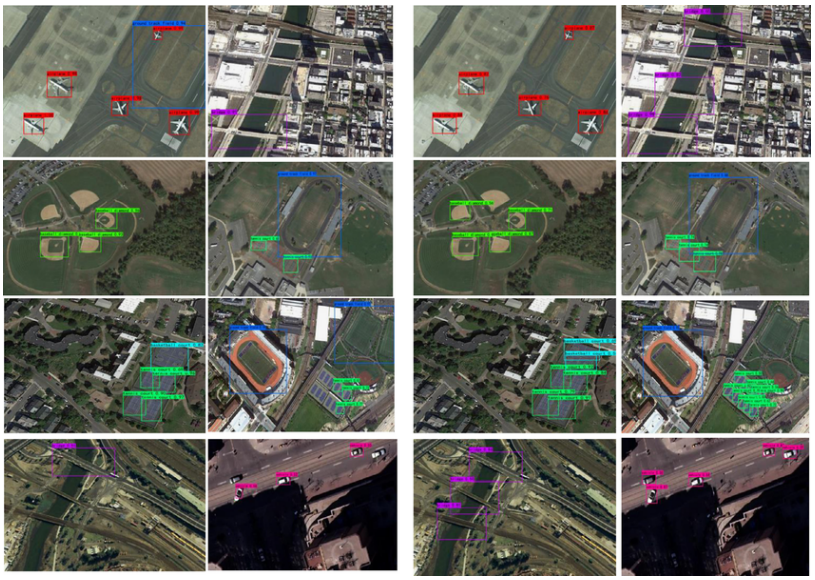

Figure 5: RESULT1

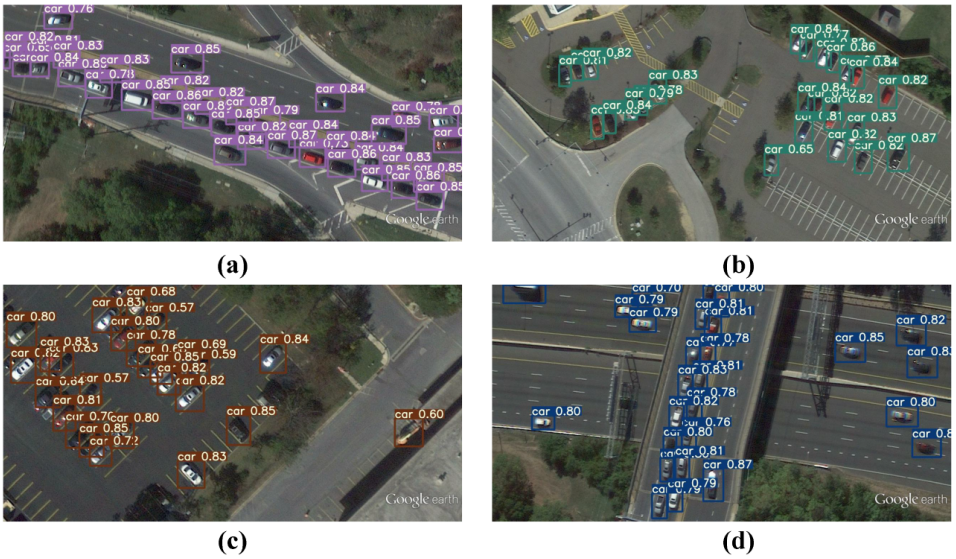

Figure 6: RESULT 2

Figure 7: RESULT 3

8 SUMMARY AND CONCLUSION

In this project, we effectively implemented the YOLOv8 object detection algorithm on the VHR-10 dataset, which consists of very high-resolution satellite imagery. The model demonstrated impressive performance in both accuracy and inference speed, making it suitable for real-time applications. It was able to reliably detect a variety of object classes such as airplanes, ships, storage tanks, and various types of vehicles—even in challenging conditions like dense object clusters or cluttered backgrounds.

Despite a few minor errors, particularly with small or overlapping objects, the overall detection quality was strong. The robustness of YOLOv8 in handling complex scenes and its efficient processing capabilities highlight its potential for use in surveillance, disaster response, urban planning, and other remote sensing applications. These results confirm that YOLOv8 is a powerful tool for object detection in high-resolution aerial imagery.