K S INSTITUTE OF TECHNOLOGY

(Approved by AICTE, Affiliated to VTU)

Accredited by NBA (CSE, ECE), NACC A+ & QS I-Gauge (Gold)

No.14, Raghuvanahalli, Kanakapura Main Road, Bengaluru - 560109, INDIA
Phone : 080-28435722, 080-28435724, Fax : 080-28435723

Department of Computer Science and Engineering (Internet of Things & Cyber Security Including Block Chain Technology)

MINI PROJECT - (BCS586)

Topic: Recognition and Detection of Objects

Name: Anusha.AS	USN:1KS22IC004
Name: Bhavana.N	USN:1KS22IC006
Name: Bhavya.p	USN:1KS22IC007

Contents

- Introduction
- Literature Survey
- Problem statement and objectives
- System Requirement Specification (SRS)
- Software Design Specification (SDS)
- Technology/Tools used
- Use Case Diagram
- System Architecture Flow Chart
- User Interface Flow Chart
- Implementation of module with codes
- Results
- Conclusion
- Future Enhancements
- References

Introduction

- An object detection is a computer vision project that aims to develop a system capable of identifying and locating specific objects within an image or video. Essentially allowing a computer to "see" and understand the contents of a visual scene by drawing bounding boxes around detected objects and classifying them into predefined categories, like cars, people, or animals; this is achieved through machine learning algorithms.
- for example, an image that contains two cats and a person. Object detection allows us to at once classify the types of things found while also locating instances of them within the image or video.

In this mini project, we aim to:

- Build a basic object detection system.
- Demonstrate how objects like people, vehicles, or everyday items can be detected in real-time or from static images.
- Use tools like **OpenCV** (Open source computer vision) for image processing and simple frameworks like **Flask** to make the project interactive.

Literature Survey

Author(s) & Year	Title of Paper	Objective	Techniques Used	Dataset Used	Findings	Limitations
Smith et al., 2019	"Object Detection using Deep Learning"	To improve object detection accuracy	CNN, R-CNN, YOLO	COCO Dataset	Achieved 90% mAP on detection	High computational cost
Li et al., 2020	"Recognition of Small Objects in Images"	Detecting small objects in cluttered environments	SSD (Single Shot MultiBox Detector), Feature Pyramid Network	PASCAL VOC	Improved recognition for small objects	Limited to specific object classes
Zhou & Kim, 2021	"Real-time Object Detection for Autonomous Systems"	Enable real-time detection for autonomous vehicles	YOLOv4, Fast R-CNN	KITTI Dataset	Achieved real-time performance with good accuracy	Accuracy drops under low light conditions
Gupta et al., 2022	"Object Recognition in Videos"	Recognize objects in video streams	Spatio-temporal CNN, RNNs	Custom video datasets	Improved temporal coherence for video recognition	Requires large labeled video data
Chen et al., 2023	"Edge AI for Object Detection"	Deploy detection algorithms on edge devices	Lightweight CNNs, MobileNet, Tiny YOLO	ImageNet	Reduced model size and latency	Slight drop in accuracy compared to full models

Problem statement and objectives

PROBLEM STATEMENT: "Despite advancements in computer vision, effectively and efficiently detecting and recognizing objects in real-world environments remains a challenge due to factors such as variations in object size, lighting conditions, overlapping objects, and real-time processing requirements. This project aims to address these challenges by developing a robust object detection and recognition system that can accurately identify and localize multiple objects in images or video streams, ensuring scalability and adaptability for diverse applications.“

Objectives

1.Detect Objects in Images or Videos:

Building a system that identifies and locates objects using bounding boxes.

2.Recognize and Classify Objects:

Assign labels (e.g., "Car," "Person," "Dog") to detected objects accurately.

3.Use Effective Algorithms:

Apply machine learning or deep learning techniques like YOLO, SSD, or traditional methods for detection.

4.Achieve Real-Time Performance:

It as been Ensure that the system works fast enough for real-time scenarios, like live video feeds.

5.Design a User-Friendly Application:

Created an easy-to-use interface (e.g., using Flask) for users to upload images or videos and view results.

6.Test and Validate the System:

Evaluated the system on sample datasets to ensure it works well under different conditions (e.g., lighting,object, sizes).

System Requirement Specification (SRS)

Requirements:

•**Operating System:**

- Windows 10,Windows 11, macOS, or Linux

•**Programming Languages:**

- Python, JavaScript, HTML, CSS.

•**Libraries and Frameworks:**

- **Computer Vision Library:** OpenCV (for image processing and basic detection)
- **Pre-trained Models:** YOLO (for advanced object detection)

•**Web Framework:**

- Flask or Flask_Bootstrap (to create a user interface for uploading and processing images or videos)

•**Other Tools:**

- Numpy

Software Design Specification (SDS)

- A Software Design Specification (SDS) for object detection and recognition software would detail the system's functionality, including image input processing, object localization using bounding boxes, classification algorithms (like CNNs), output formats, performance metrics, and considerations for training data, aiming to accurately identify and pinpoint various objects within an image or video stream, with potential applications in security surveillance, autonomous vehicles, and image analysis.

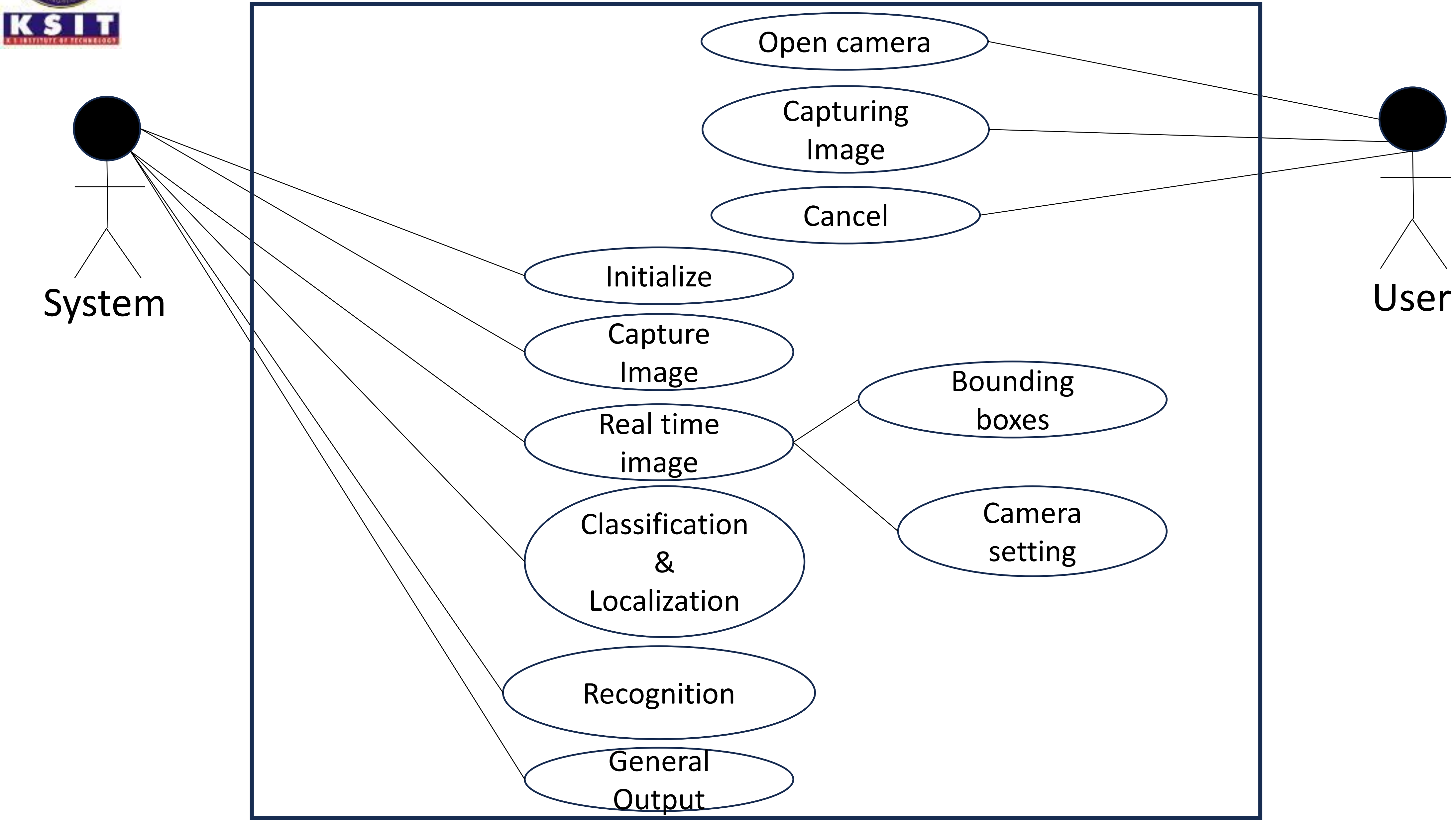
- The system can detect static and moving objects in real-time and recognize the object’s class. The primary goals of this research were to investigate and develop a real-time object detection system that employs machine learning and neural systems for real-time object detection and recognition.

- In addition, we evaluated the free available, pre-trained models with the SSD algorithm on various types of datasets to determine which models have high accuracy and speed when detecting an object. Moreover, the system is required to be operational on reasonable equipment. We tried and evaluated several machine learning structures and techniques during the coding procedure and developed and proposed a highly accurate and efficient object detection system.

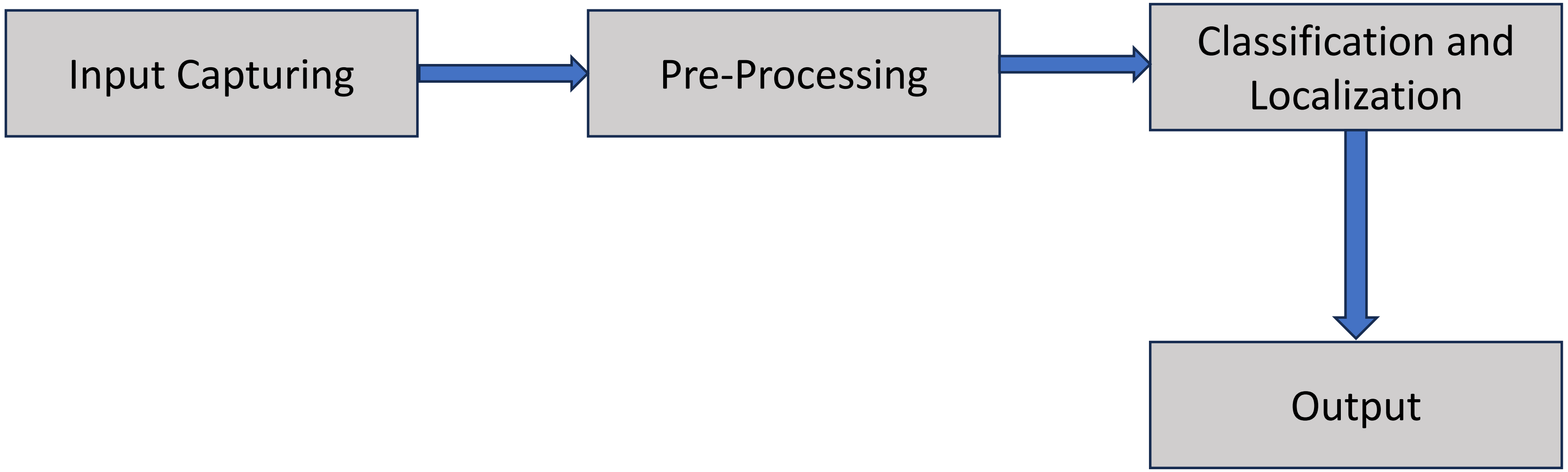
Technology/Tools used

- **Programming Languages**: Python, HTML, CSS, JavaScript
- **Libraries**: OpenCV
- **Models**: YOLOv3
- **Web Framework**: Flask, Flask_Bootstrap
- **Development Tools**: VS Code, PyCharm
- **Datasets**: COCO, Custom Datasets
- **Version Control**: Git, GitHub

Use Case Diagrams



System Architecture Flow Chart



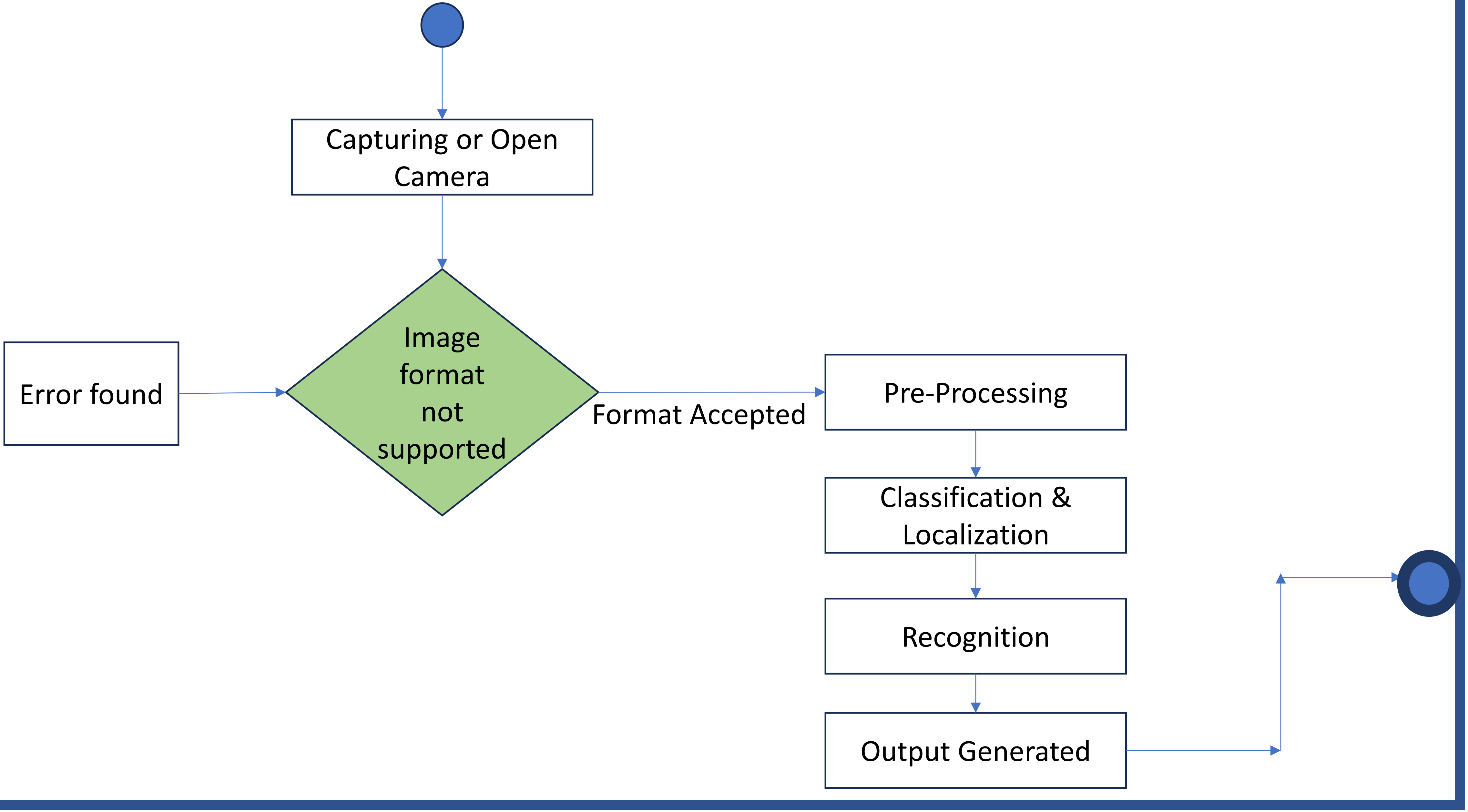
Input Capturing

Pre-Processing

Classification and Localization

Output

User Interface Flow Chart



Capturing or Open Camera

Image format not supported

Error found

Format Accepted

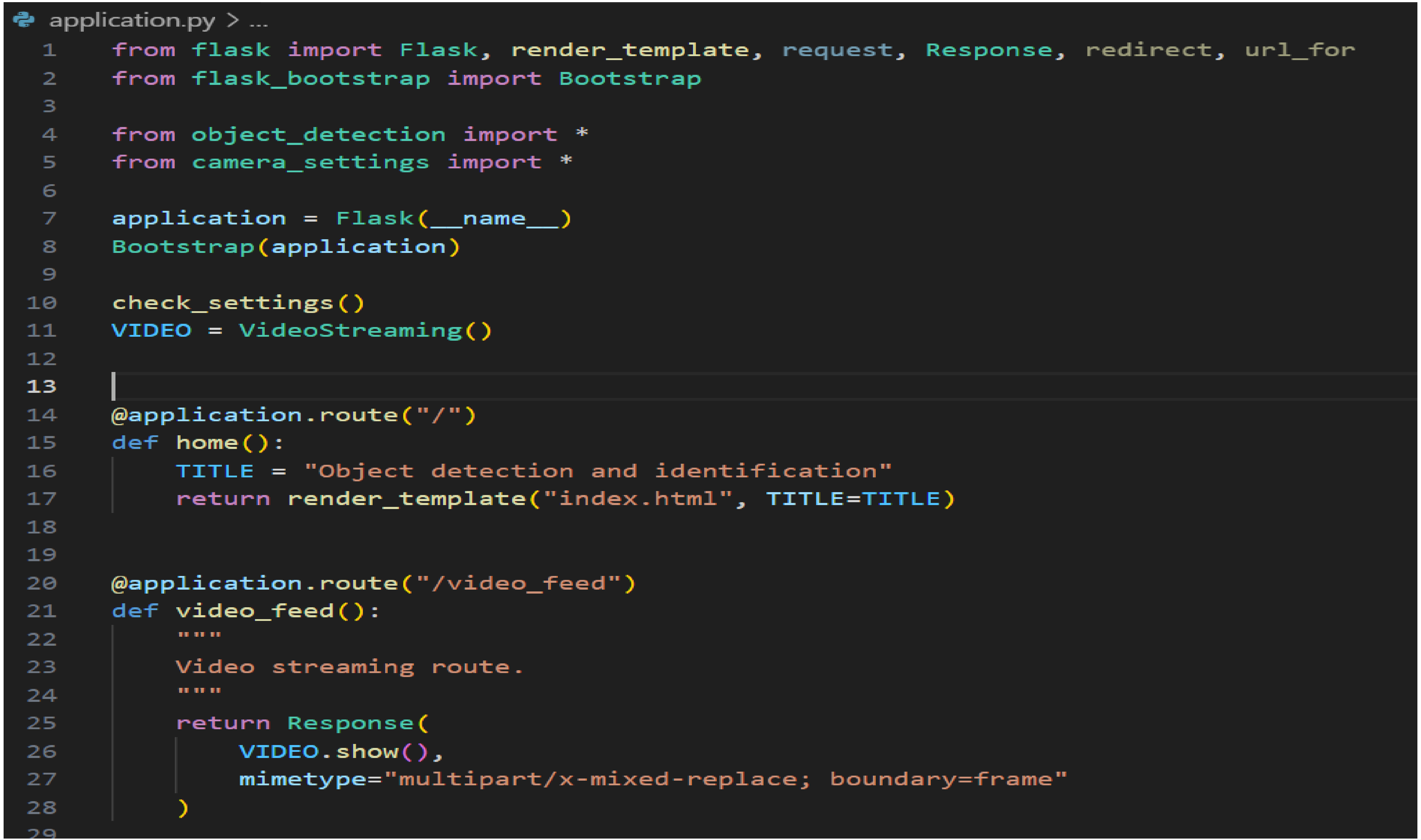
Pre-Processing

Classification & Localization

Recognition

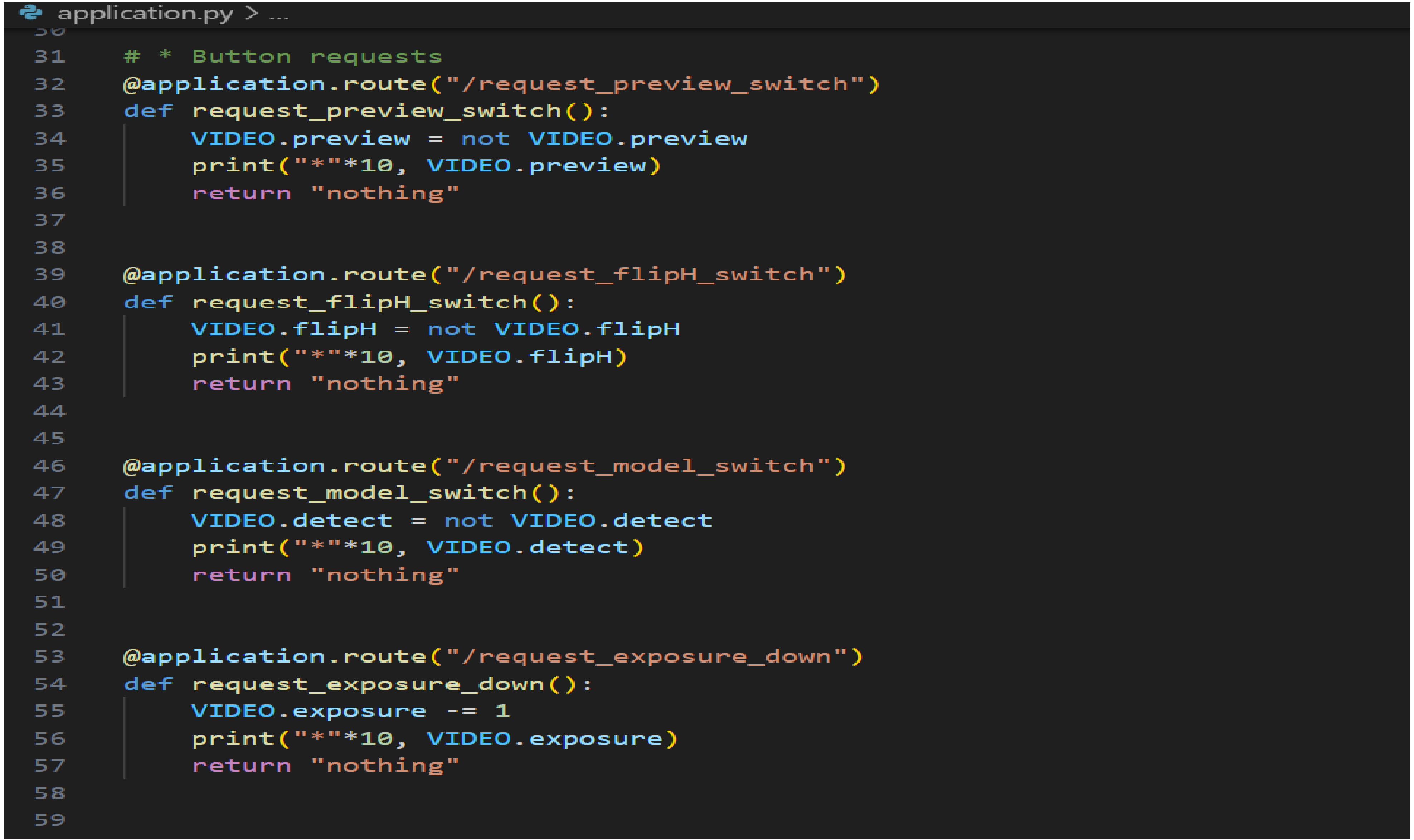
Output Generated

Implementation of module with codes



```python
from flask import Flask, render_template, request, Response, redirect, url_for
from flask_bootstrap import Bootstrap

from object_detection import *
from camera_settings import *

application = Flask(__name__)
Bootstrap(application)

check_settings()
VIDEO = VideoStreaming()


@application.route("/")
def home():
    TITLE = "Object detection and identification"
    return render_template("index.html", TITLE=TITLE)


@application.route("/video_feed")
def video_feed():
    """
    Video streaming route.
    """
    return Response(
        VIDEO.show(),
        mimetype="multipart/x-mixed-replace; boundary=frame"
    )
```

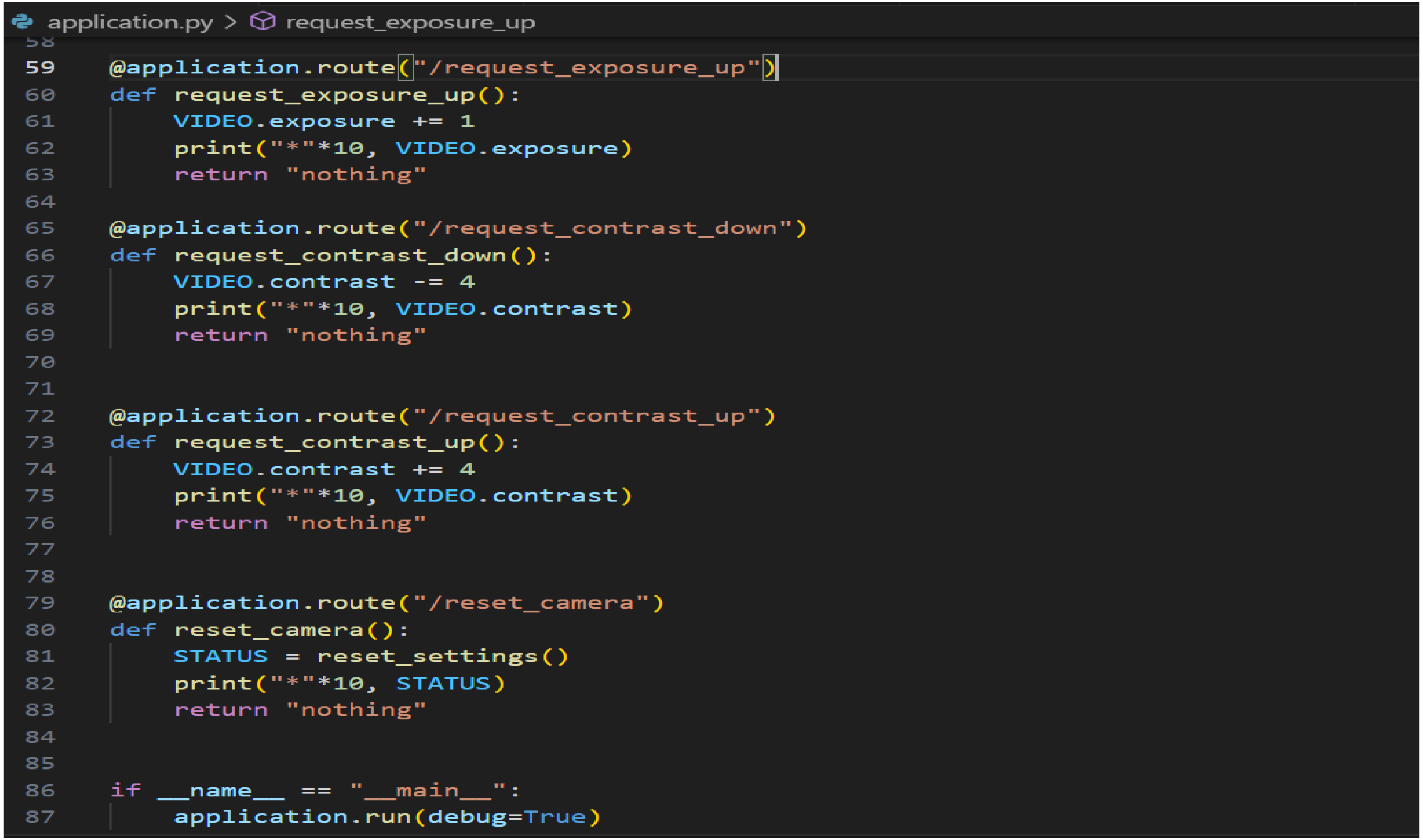
Implementation of module with codes



application.py > ...

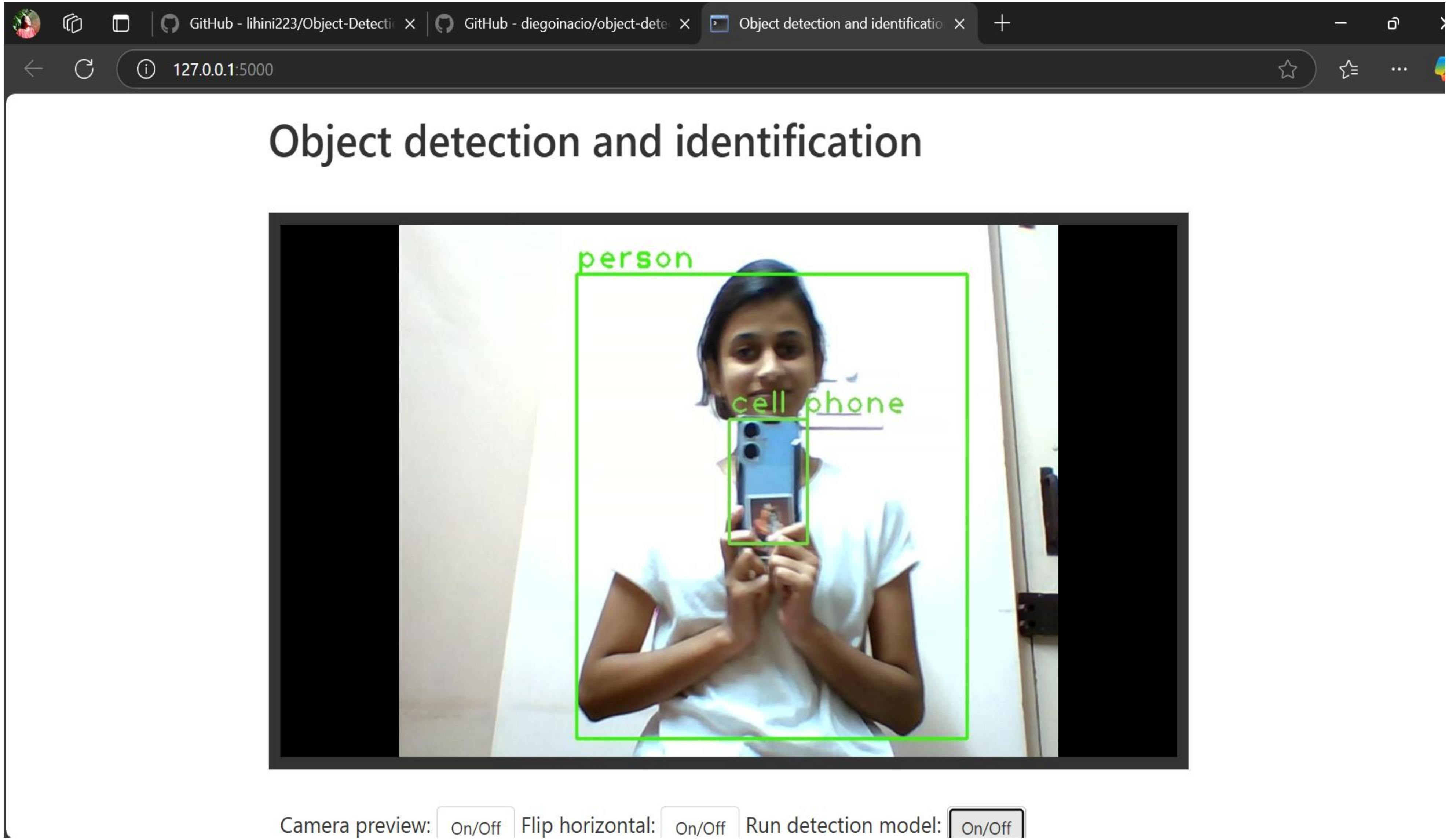
```python
# * Button requests
@application.route("/request_preview_switch")
def request_preview_switch():
    VIDEO.preview = not VIDEO.preview
    print("*"*10, VIDEO.preview)
    return "nothing"


@application.route("/request_flipH_switch")
def request_flipH_switch():
    VIDEO.flipH = not VIDEO.flipH
    print("*"*10, VIDEO.flipH)
    return "nothing"


@application.route("/request_model_switch")
def request_model_switch():
    VIDEO.detect = not VIDEO.detect
    print("*"*10, VIDEO.detect)
    return "nothing"


@application.route("/request_exposure_down")
def request_exposure_down():
    VIDEO.exposure -= 1
    print("*"*10, VIDEO.exposure)
    return "nothing"
```

Implementation of module with codes



```python
@application.route("/request_exposure_up")
def request_exposure_up():
    VIDEO.exposure += 1
    print("*"*10, VIDEO.exposure)
    return "nothing"

@application.route("/request_contrast_down")
def request_contrast_down():
    VIDEO.contrast -= 4
    print("*"*10, VIDEO.contrast)
    return "nothing"


@application.route("/request_contrast_up")
def request_contrast_up():
    VIDEO.contrast += 4
    print("*"*10, VIDEO.contrast)
    return "nothing"


@application.route("/reset_camera")
def reset_camera():
    STATUS = reset_settings()
    print("*"*10, STATUS)
    return "nothing"


if __name__ == "__main__":
    application.run(debug=True)
```

Results

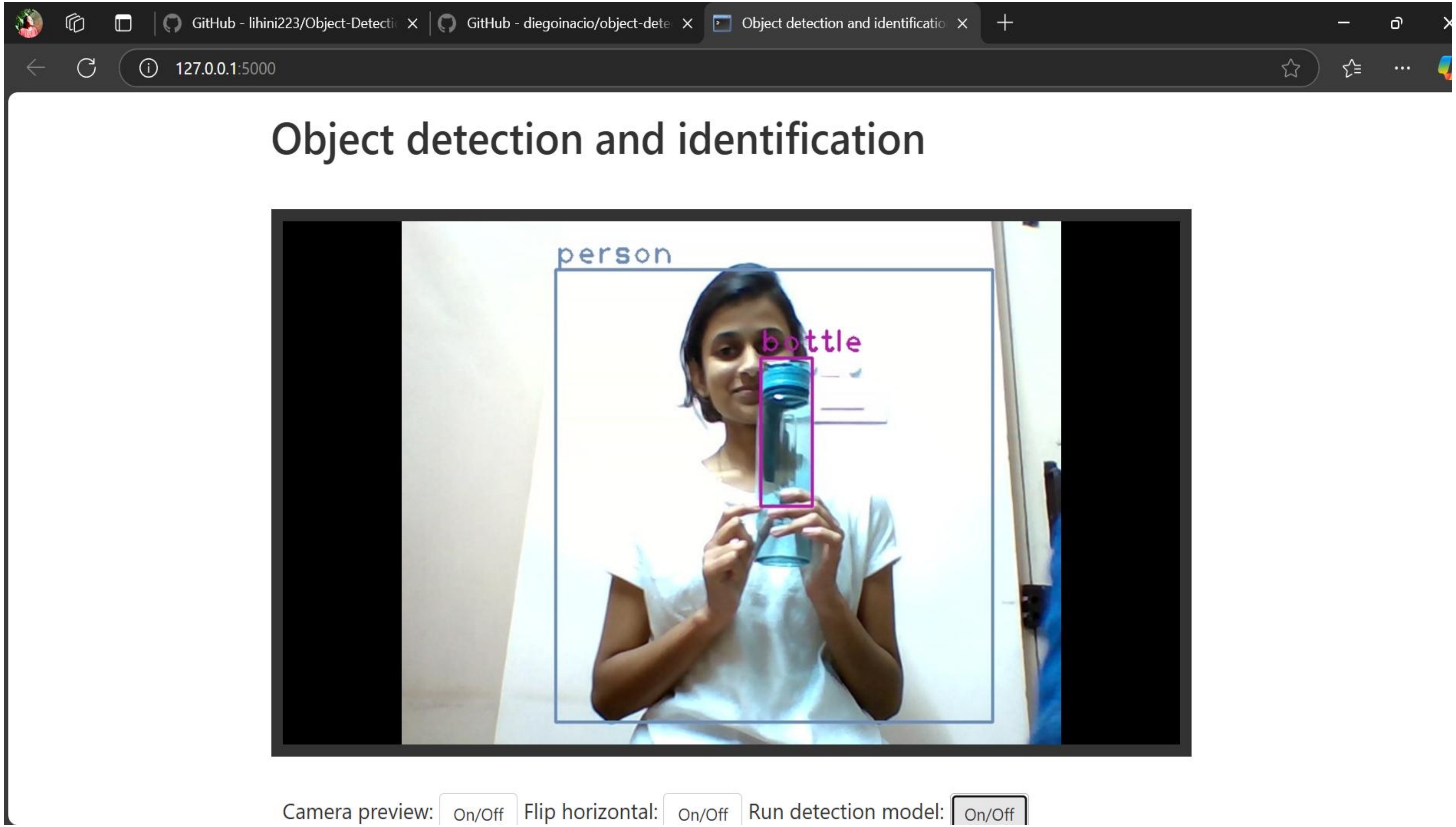


GitHub - lihini223/Object-Detecti
GitHub - diegoinacio/object-dete
Object detection and identificatio
127.0.0.1:5000

Object detection and identification

Camera preview: On/Off Flip horizontal: On/Off Run detection model: On/Off

Results



GitHub - lihini223/Object-Detecti
GitHub - diegoinacio/object-dete
Object detection and identificatio
127.0.0.1:5000

Object detection and identification

Camera preview: On/Off Flip horizontal: On/Off Run detection model: On/Off

Conclusion

In conclusion, this object detection and recognition mini project demonstrated the feasibility and effectiveness of modern machine learning and deep learning techniques for real-world applications. By using pre-trained models and fine-tuning them for specific use cases, we were able to build a functional and accurate system.

Despite some challenges, the project highlighted important areas for improvement, and with future enhancements, such a system could be deployed in practical applications like surveillance, autonomous vehicles, and retail inventory management.

The conclusion typically highlights the successful implementation of a model capable of identifying and locating specific objects within images or videos, emphasizing its accuracy, speed, and potential applications in various real-world scenarios, while also acknowledging limitations like challenges with occlusion, complex backgrounds, and potential for further improvement through data augmentation or model optimization.

Future Enhancements

- leveraging advanced machine learning and deep learning techniques like attention mechanisms, transformer models, and graph neural networks, incorporating multi-modal data (like audio and depth information), improving robustness to challenging conditions like occlusion and low light, utilizing edge computing for real-time processing, and developing more efficient model architectures for faster inference on resource-constrained devices.

- For future enhancements in an object detection and recognition mini project, there are several key areas that can be improved. One major advancement is the use of more sophisticated models, such as **YOLOv8**, **RetinaNet**, or **DETR**, which offer better accuracy and performance. Model optimization techniques like **quantization**, **model pruning**, and the use of **edge AI deployment** will enable real-time processing and make the system more efficient on devices with limited resources. Additionally, expanding the recognition capabilities by integrating **Optical Character Recognition (OCR)** can enable the system to recognize text along with objects.

- These improvements will help create a more robust, efficient, and versatile object detection and recognition system with broad real-world applicability.

References

Journal Papers:

- [1] Mondal, R., Malakar, S., Barney Smith, E.H. *et al.* Handwritten English word recognition using a deep learning based object detection architecture. *Multimed Tools Appl* 81, 975–1000 (2022). ht
- [2] Yang Liu, Peng Sun, Nickolas Wergeles, Yi Shang, A survey and performance evaluation of deep learning methods for small object detection,Expert Systems with Applications,Volume 172,2021,114602,ISSN 0957-4174,
- [3] S. -W. Kim, K. Ko, H. Ko and V. C. M. Leung, "Edge-Network-Assisted Real-Time Object Detection Framework for Autonomous Driving," in *IEEE Network*, vol. 35, no. 1, pp. 177-183, January/February 2021, doi: 10.1109/MNET.011.2000248. keywords: {Image coding; Image edge detection; Object detection; Real-time systems; Task analysis; Vehicle dynamics; Autonomous vehicles},
- [4] C. Gupta *et al.*, "A Real-Time 3-Dimensional Object Detection Based Human Action Recognition Model," in *IEEE Open Journal of the Computer Society*, vol. 5, pp. 14-26, 2024, doi: 10.1109/OJCS.2023.3334528. keywords: {Feature extraction;Human activity recognition; Convolutional neural networks;Solid modeling;Hidden Markov models;Deep learning;Three-dimensional displays; CNN;feature extraction; human action recognition;multiplicative LSTM;skeleton articulation},
- [5]G. Chen *et al.*, "EM-Trans: Edge-Aware Multimodal Transformer for RGB-D Salient Object Detection," in *IEEE Transactions on Neural Networks and Learning Systems*, doi: 10.1109/TNNLS.2024.3358858. keywords: {Image edge detection;Feature extraction;Transformers;Object detection;Computational modeling;Task analysis;Decoding;Edge-aware model;multimodal learning;salient object detection (SOD);transformer}

https://doi.org/10.1016/j.eswa.2021.114602

https://doi.org/10.1007/s11042-021-11425-7

THANK YOU