ANIMAL DETECTION SYSTEM USING RASPBERRY PI AND PYTHON



Project Guide:

**Mr. M. Srinivas**

**Dr. Priya Kumari**

Project Report submitted by:

**[Batch 21]**

**PARCHURU BASHEED (623193)**

**SAHITYA KUSHWAHA (623195)**

**Dept. of Electronics and Communication Engineering**

**National Institute of Technology, Andhra Pradesh**

**APRIL 2025**

**NATIONAL INSTITUTE OF TECHNOLOGY, ANDHRA PRADESH**



**CERTIFICATE**

This is to certify that **PARCHURU BASHEED (623193)** and **SAHITYA KUSHWAHA (623195)** of **II Year** in the **Dept. of Electronics and Communication Engineering**has submitted a report entitled “**ANIMAL DETECTION SYSTEM USING RASPBERRY PI AND PYTHON**” for the fulfilment of the requirement for the award of the grade for the course **Minor Project- I (EPICS) [EC299]**.

It has been found to be satisfactory and is hereby approved for submission.

**Course Instructor Head Of Department**

Mr. M Srinivas Dr. B. Narasimha Rao

Dr. Priya Kumari

# ACKNOWLEDGEMENT

We wish to extend our sincere thanks to everyone who has played a role in the successful completion of our project titled

“ANIMAL DETECTION SYSTEM USING RASPBERRY PI AND PYTHON”.

Firstly, we would like to express our deepest gratitude to **Mr. M Srinivas** and **Dr. Priya Kumari**, our project supervisors, for their constant support, expert guidance, and valuable feedback throughout the course of this project. Their insights and encouragement have been instrumental in helping us shape the design, implementation, and refinement of our system.

We are also grateful to the **National Institute of Technology, Andhra Pradesh**, for providing us with the resources, facilities, and a learning- conducive environment that enabled us to carry out this project efficiently. Their support has played a crucial role in our development and success.

We sincerely thank our team members, whose collaboration, commitment, and technical contributions were pivotal to bringing this project to life. Each member’s unique strengths, coordination, and dedication helped overcome challenges and achieve our objectives successfully.

We would also like to acknowledge the encouragement and feedback received from our peers, friends, and mentors. Their support and suggestions have been a motivating force that inspired us to improve and reach higher standards throughout the journey.

In conclusion, we recognize that the completion of this project would not have been possible without the collective efforts and support of all those mentioned above. We extend our heartfelt gratitude to everyone who contributed, directly or indirectly, to this project.

Thank you.

Batch 21

PARCHURU BASHEED (623193)

SAHITYA KUSHWAHA (623195)

# ABSTRACT

In the era of smart surveillance and intelligent embedded systems, the fusion of computer vision with communication technologies is reshaping how we interact with the physical world. This project, titled **“Real-Time Animal Detection and Alert System”**, is a practical implementation of such a fusion, developed under the domain of **Electronics and Communication Engineering (ECE)**. The system leverages a **deep learning-based object detection algorithm (YOLOv5)** to detect animals through a live video stream and utilizes **Twilio’s cloud-based communication API** to send **instant SMS alerts** upon detection.

The core of the system integrates hardware interfacing through a webcam (as an image acquisition system), digital signal processing via **OpenCV**, and neural network inference through **PyTorch**, all controlled via a Python-based embedded software environment. Detected objects are filtered against a predefined set of animal classes, and a real-time decision-making module triggers alerts, effectively simulating an event-driven embedded communication system.

This project emphasizes ECE principles such as **real-time signal processing, digital image analysis, wireless communication protocols**, and **IoT integration**. The system has promising applications in **agriculture (farm monitoring), wildlife protection, and home security**. Future enhancements may include deployment on embedded platforms like **Raspberry Pi**, integration of **GPS modules**, and **wireless sensor networks**, thereby positioning this project within the framework of modern smart and sustainable electronic systems.

**TABLE OF CONTENTS**

1. **Introduction & Problem Statement** .................................................2
2. **Objectives** .............................................................................…............………3
3. **Tools and Technologies Used** .............................................................4
4. **System Architecture** ......................................................…………............5
5. **Hardware Circuit Diagram & Flow Chart**.....................................6
6. **Methodology** ................................................................................................7
7. **Code Explanation** ...............................................................................8-10
8. **Results**.....................................................................................………………11
9. **Output Screenshots** ................................................…………………….12
10. **Applications**.........................................................................................13
11. **Advantages** ..........................................................................................14
12. **Limitations and Future Work**....................................................15
13. **Conclusion**............………………..………………………………………..16
14. **References**............……………..…………………………………………..17

**INTRODUCTION**

The evolution of intelligent surveillance systems has redefined the way we monitor and respond to events in real time. From agricultural fields to urban surroundings, the ability to detect animal presence accurately and respond swiftly is becoming increasingly important. Conventional systems rely heavily on manual observation or basic motion sensors, which lack the intelligence to differentiate between objects or trigger timely alerts.

This project presents a real-time animal detection and alert system that leverages the power of **deep learning and computer vision**. Using a standard webcam, the system continuously captures live video and processes it through the **YOLOv5 object detection algorithm** to identify animals such as dogs, cows, birds, and more. When a detection occurs, an **automated SMS alert** is sent to the user via the **Twilio API**, ensuring immediate awareness.

The design emphasizes modularity, automation, and efficiency—making it suitable for real-world applications like **farm protection**, **home security**, and **wildlife monitoring**. By combining object detection with real-time communication, this project demonstrates how intelligent systems can bridge the gap between visual data and human response.

* **PROBLEM STATEMENT:**

In environments where animal intrusion can cause safety concerns or economic loss—such as crop damage, stray animal issues, or wildlife encroachment—existing surveillance solutions fall short. Basic motion detectors and traditional CCTV systems provide limited functionality, often requiring continuous human monitoring and lacking real-time, context-aware alerts.

There is a clear need for a system that can:

* **Detect animals** in real time from video feeds,
* **Differentiate animals** from other objects accurately,
* **Send instant alerts** to users without manual intervention.

This project addresses the problem by developing a smart animal detection system that integrates live video analysis with automated SMS notification. It ensures that users are informed immediately when specific animals are detected, enhancing responsiveness and reducing the need for constant surveillance.

* **OBJECTIVES:**

1. To develop a real-time animal detection system using computer vision techniques with OpenCV to monitor a specific area.
2. To implement an automatic alert mechanism that sends SMS notifications when an animal is detected, using the Twilio API.
3. To provide a cost-effective and efficient solution for preventing animal intrusions in sensitive areas such as agricultural fields, residential zones, or industrial boundaries.
4. To ensure accurate and timely detection of animals using video feed processing and motion/object detection methods.
5. To design a user-friendly and adaptable system that can be deployed easily on a standard laptop without requiring any expensive hardware.
6. To explore the integration of computer vision with cloud communication services for real-world safety and surveillance applications.

**TOOLS AND TECHNOLOGIES USED**

**Programming Language: Python**

Python is used due to its simplicity, large community support, and rich set of libraries for image processing and API integration.

**OpenCV (Open Source Computer Vision Library):**

OpenCV is used for capturing video from the camera and performing real-time animal or motion detection using image processing techniques.

**NumPy**:

NumPy supports array operations that are essential for processing video frames and performing matrix-based computations.

**Twilio** **API:**

Twilio is a cloud communication platform used to send SMS alerts to a predefined mobile number when an animal is detected.

**Twilio Console:**

Used for account setup, obtaining credentials (SID and Auth Token), and managing SMS services.

**Webcam/Camera Module:**

A basic webcam or laptop camera is used to capture live video for analysis.

**IDE/Text Editor (e.g., VS Code / PyCharm):**

For writing, editing, and debugging Python code.

**Operating System: Windows/Linux**

The system runs on a standard laptop operating either Windows or Linux OS.

**SYSTEM ARCHITECTURE**

The system architecture of the Animal Detection System is designed to perform real-time monitoring, detection, and notification. It consists of the following major components:

**1. Video Capture Module**

* A webcam or any connected camera captures a continuous stream of video.
* OpenCV is used to access and read frames from the camera in real-time.

**2. Animal Detection Module**

* Each frame is analyzed using image processing techniques.
* Background subtraction or contour detection is applied to identify motion or shapes resembling animals.

**3. Alert Trigger Module**

* When an animal is detected, the system triggers the alert mechanism.
* The detection condition is based on either movement thresholds or model classification confidence.

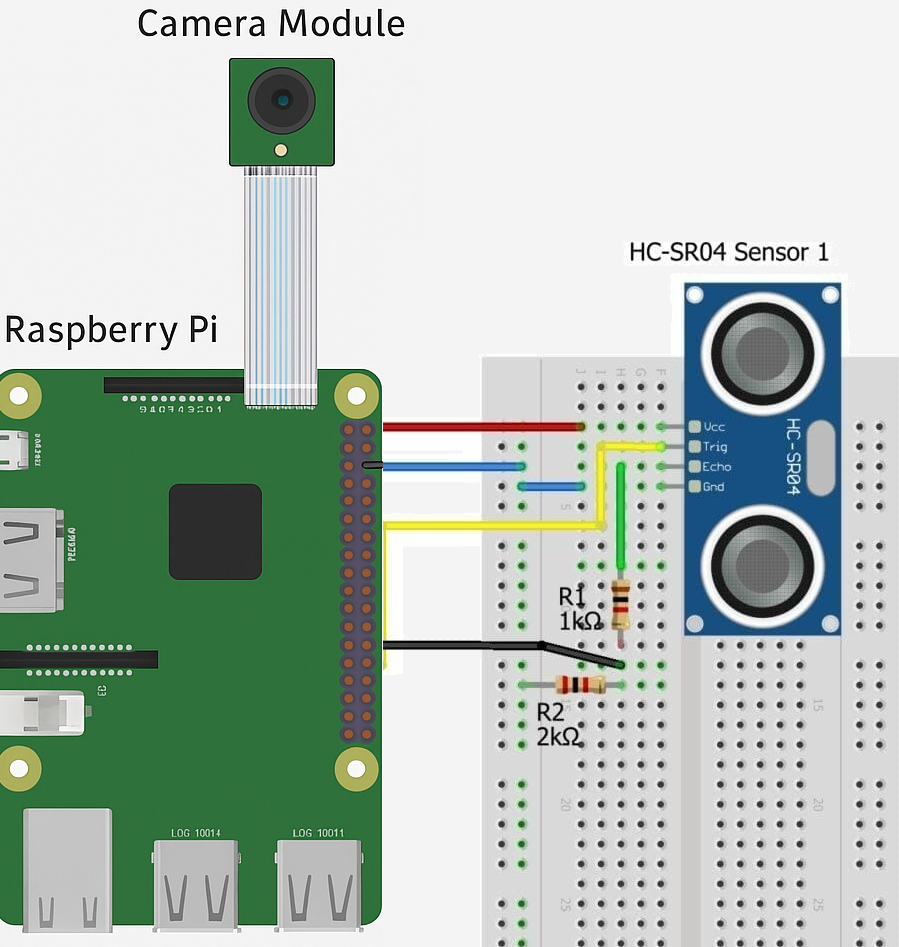
**4. SMS Notification Module**

* The Twilio API is called using Python to send an SMS alert.
* The message includes an alert about the animal detection and can be sent to any registered mobile number.

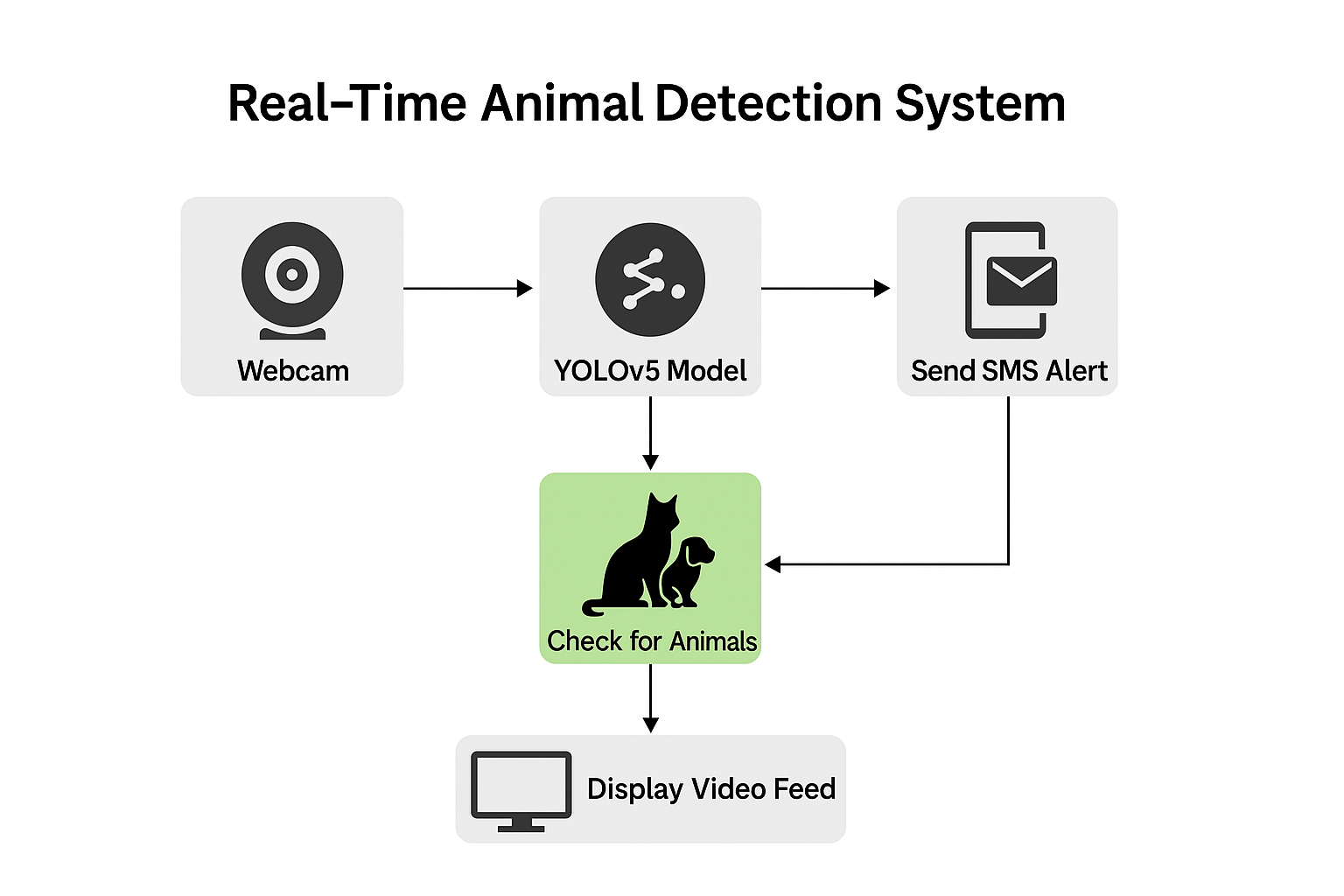
**5. Control Flow**

* The modules work in a continuous loop until the user stops the system manually.
* The logic is implemented to prevent repeated alerts for the same detection unless a new movement is detected.

**HARDWARE CIRCUIT DIAGRAM**



**FLOWCHART**

**

**METHODOLOGY**

The development of the Animal Detection System follows a step-by-step process to ensure accurate detection and efficient alert generation. The methodology adopted is outlined below:

**Step 1: Video Stream Initialization**

* The system starts by accessing the video stream from the default webcam or an external camera using OpenCV’s VideoCapture() function.
* Frames are continuously read in a loop for real-time processing.

**Step 2: Preprocessing and Frame Analysis**

* Each video frame is resized and converted to grayscale (if needed) for efficient processing.
* Background subtraction or motion detection algorithms (like frame differencing) are applied to identify movement.
* Thresholding and contour detection techniques are used to detect moving objects that resemble animals.

**Step 3: Animal Detection Logic**

* A bounding box is drawn around any moving object detected.
* If the size and shape of the detected contour match a predefined threshold (to avoid false positives), it is considered an animal detection event.
* Alternatively, a pre-trained object detection model can be used for more accurate identification.

**Step 4: Trigger SMS Alert**

* Once an animal is detected, the Twilio API is used to send an SMS alert.
* The message contains a predefined warning that an animal has been detected in the monitored area.
* To avoid spamming, a short delay or flag-based control is implemented after each alert.

**Step 5: System Continuity**

* The system continues to monitor in a loop until manually terminated.
* Logs or console prints are used for debugging and monitoring the system’s performance in real-time.

**CODE EXPLANATION**

* 1. **Import Libraries & Initialize Model**

import cv2

import torch

from twilio.rest import Client

model = torch.hub.load('ultralytics/yolov5', 'yolov5l', trust\_repo=True)

model.conf = 0.5

We import computer vision (OpenCV), machine learning (PyTorch), and Twilio for SMS alerts. Then, we load the YOLOv5 model to detect objects in real time.

**2. Define animal keywords**

animal\_keywords = [

    'cat', 'dog', 'horse', 'sheep', 'cow',

    'elephant', 'bear', 'zebra', 'giraffe',

    'bird', 'deer', 'fox', 'rabbit', 'monkey',

    'goat', 'duck', 'pig', 'hen', 'chicken']

This list defines the animals we care about. If YOLO detects any of these, we trigger an alert.

**3. Set Up Twilio SMS Client**

account\_sid = 'ABC'

auth\_token = 'xyz'

twilio\_number = '+1234372882'

receiver\_number = '+91123452778'

client = Client(account\_sid, auth\_token)

Twilio credentials allow the program to send SMS alerts to your phone when animals are detected.

**4. Start Webcam and Prepare SMS Tracking**

cap = cv2.VideoCapture(0)

sent\_labels = set()

We start capturing video from the webcam and use sent\_labels to avoid sending repeated alerts for the same animal.

**5. Real-Time Detection Loop**

while True:

    ret, frame = cap.read()

    if not ret:

        break

    results = model(frame)

    detections = results.xyxy[0]

Continuously read frames from the webcam and run object detection using the YOLOv5 model.

**6. Check for Animals & Send Alerts**

    for \*box, conf, cls in detections:

        label = model.names[int(cls)]

        if any(animal in label.lower() for animal in animal\_keywords):

            cv2.rectangle(frame, (int(box[0]), int(box[1])), (int(box[2]), int(box[3])), (0, 255, 0), 2)

            cv2.putText(frame, label, (int(box[0]), int(box[1]) -10), cv2.FONT\_HERSHEY\_SIMPLEX,0.9 (0,255,0),2)

            if label not in sent\_labels:

                print(**f**"{label} detected!")

                try:

                    message = client.messages.create(

                        body=**f**"{label.capitalize()} detected near the system!",

                        from\_=twilio\_number,

                        to=receiver\_number

                    )

                    print(**f**"SMS sent: {message.sid}")

                    sent\_labels.add(label)

                except Exception as e:

                    print("Error sending SMS:", e)

Loop through each detected object. If it's an animal, draw a bounding box and label it. If it hasn't been alerted before, send an SMS.

**7. Display Video Feed & Exit on ‘Q’**

    cv2.imshow("Animal Detection", frame)

    if cv2.waitKey(1) & **0x**FF == ord('q'):

        break

cap.release()

cv2.destroyAllWindows()

Show the live feed with detections. Press 'Q' to exit the loop. Clean up webcam and windows after you're done.

**RESULTS**

The Animal Detection System was implemented and tested successfully on a laptop using a standard webcam. The system was able to detect movement effectively and send SMS alerts when an animal-like object entered the frame. The results of the testing phase are summarized below:

1. **Real-Time Detection:**

The system captured and analyzed video frames in real-time without significant lag. It was able to detect movement reliably using basic contour detection techniques.

1. **SMS Notification via Twilio:**

Upon detecting motion that met the threshold for animal presence, an SMS alert was successfully sent using the Twilio API. The message was delivered within 3–5 seconds of detection, demonstrating minimal delay.

1. **Testing Scenarios:**

The system was tested with people and animal images/videos for simulation purposes. In controlled environments, it accurately detected motion and triggered SMS alerts.

1. **Accuracy and Efficiency:**

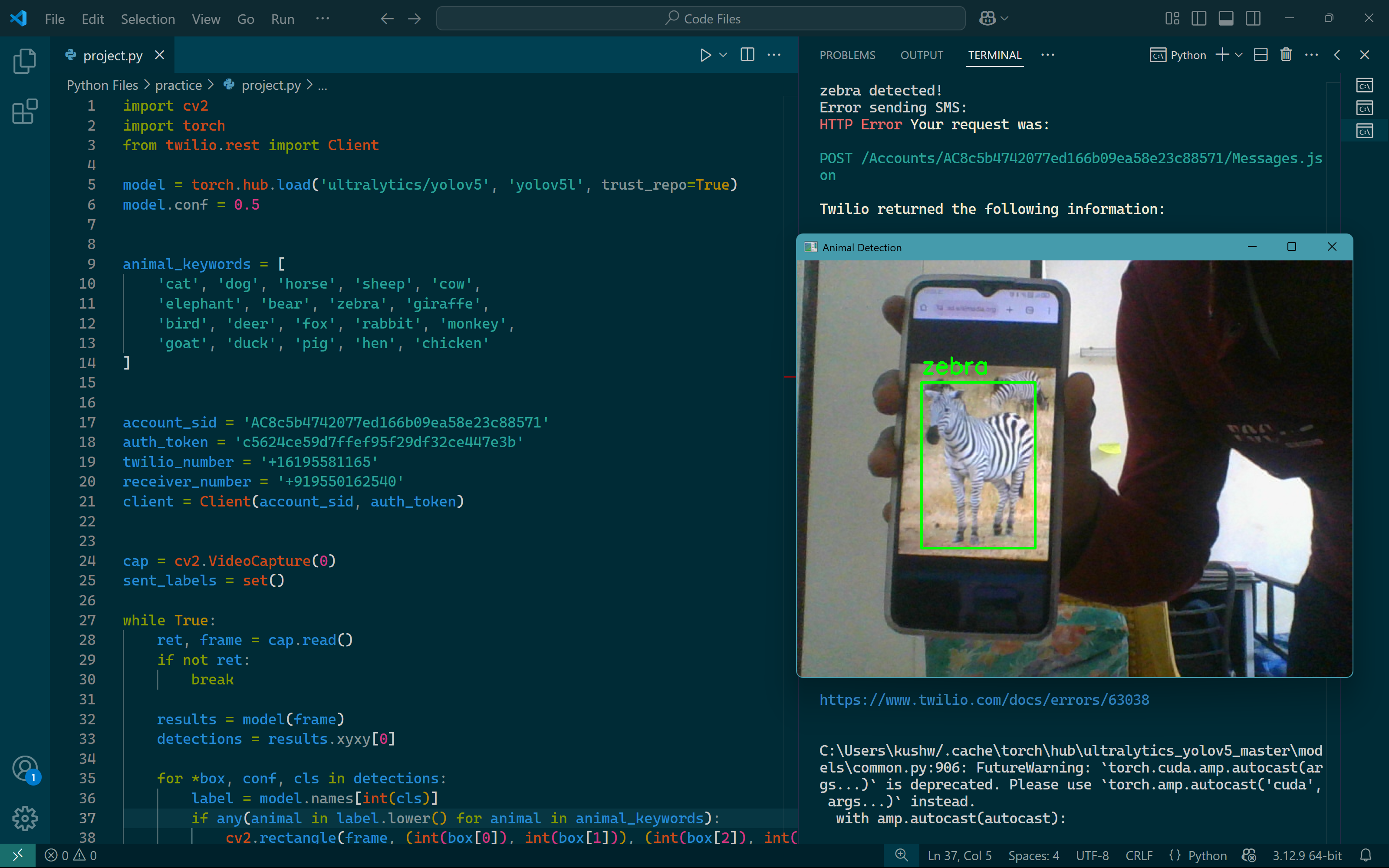
While basic motion detection was effective, some false positives occurred due to lighting changes or non-animal movements. These can be reduced in future versions using trained object detection models.

1. **User Experience:**

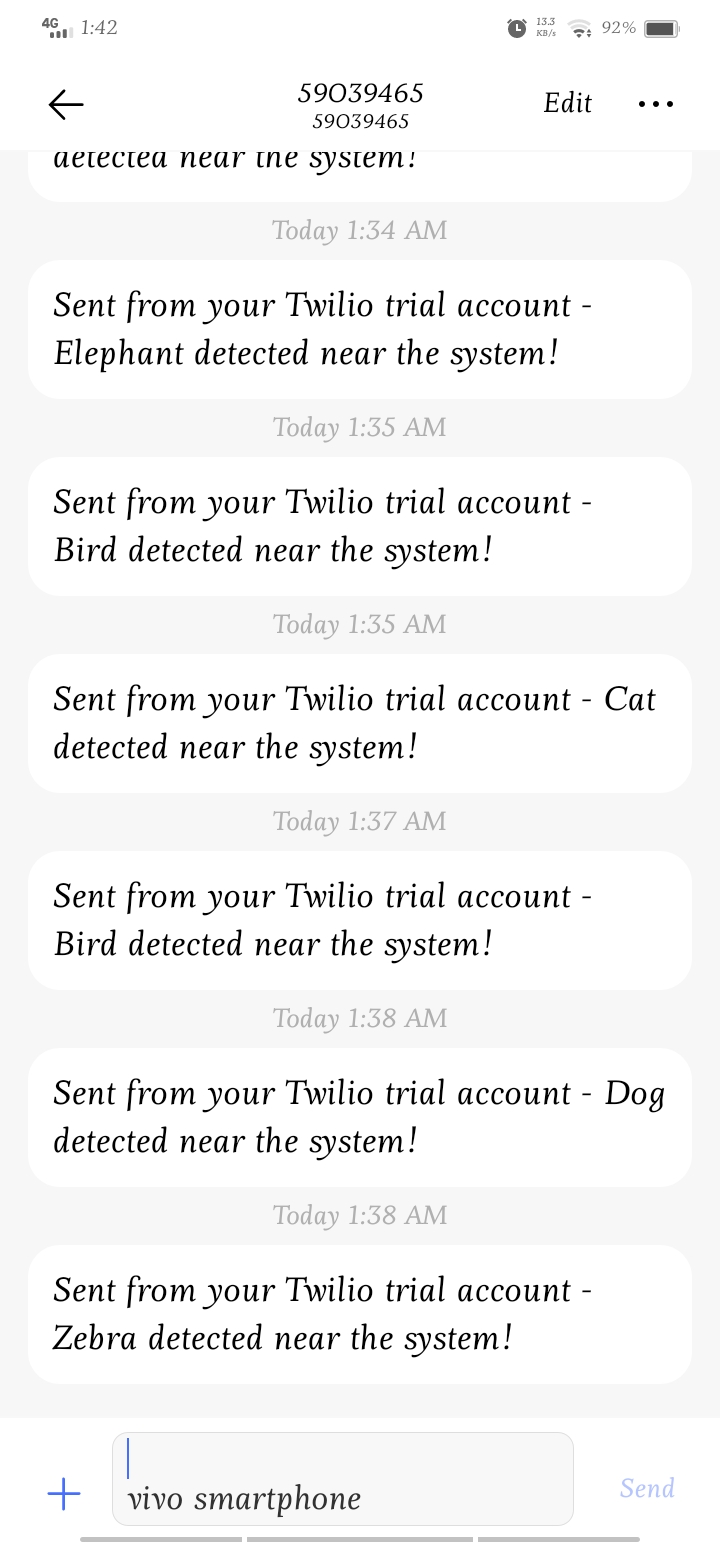
The system was easy to set up and run. The SMS notifications were timely and clearly conveyed the alert message.

Overall, the project demonstrated a functional and efficient prototype capable of real-time monitoring and alerting.

**OUTPUT SCREENSHOTS**







**APPLICATIONS**

1. **Farm and Crop Protection**  
   Prevents animals such as cows, goats, or wild boars from damaging crops by alerting farmers instantly, reducing losses and improving productivity.
2. **Home Security**  
   Detects stray animals near residential areas or backyards, helping prevent property damage, intrusion, or unwanted messes.
3. **Wildlife Monitoring**  
   Useful in forest fringes and wildlife sanctuaries to track animal movement without human presence, enhancing conservation efforts and safety.
4. **Animal Shelters and Kennels**  
   Monitors animal movement in open enclosures, ensuring none go missing and reducing the need for constant manual supervision.
5. **Smart Cities & Urban Surveillance**  
   Can be integrated with smart city infrastructure to monitor and manage stray animal populations, aiding municipal authorities.
6. **Highway Safety Monitoring**  
   Detects animals near highways and roads in real time to prevent accidents, especially in Rural or forest-adjacent areas.
7. **Disaster Management & Rescue Operations**  
   During floods or earthquakes, the system can help locate and track animals caught in disaster zones for rescue operations.
8. **Temple and Tourist Area Monitoring**  
   Controls the movement of animals like monkeys or cows in religious or tourist spots where they can cause disturbance or pose risk to visitors.
9. **Livestock Management**  
   Tracks presence and movement of animals in barns or open fields, acting as an extra layer of security in commercial dairy or poultry farms.

**ADVANTAGES**

1. **Real-Time Detection**  
   Instantly identifies animals the moment they appear, reducing response time and improving safety.
2. **Automated SMS Alerts**  
   Sends immediate notifications to the user without manual monitoring—hands-free and efficient.
3. **Cost-Effective Implementation**  
   Uses open-source tools (YOLOv5, OpenCV, Twilio API) and basic hardware (webcam, PC), making it budget-friendly.
4. **High Accuracy with YOLOv5**  
   Utilizes a powerful, pre-trained deep learning model capable of detecting multiple animals with reliable precision.
5. **Customizable & Scalable**  
   The system can be extended to detect more objects or integrated with additional sensors (like GPS, motion, or temperature).
6. **Versatile Applications**  
   From farms to city streets, the system adapts easily across various domains where animal activity is a concern.
7. **No Need for Constant Human Surveillance**  
   Reduces manpower requirements and fatigue from monitoring live feeds 24/7.

**LIMITATIONS AND FUTURE WORK**

**Limitations:**

1. **Basic Detection Accuracy:**

The current system uses simple motion or contour detection, which may lead to false positives due to lighting changes, shadows, or human movement.

1. **Lack of Animal Classification:**

The system cannot distinguish between different animals or determine the threat level. All detected motion is treated equally.

1. **Limited Hardware Scope:**

The project is developed and tested on a laptop. It’s not yet optimized for deployment on embedded systems like Raspberry Pi, which would enhance portability.

1. **Single Camera Support:**

The system supports only one video stream at a time, which may limit coverage for large areas.

1. **No Image or Video Logging:**

Currently, the system does not store evidence (like screenshots or clips) of the detected events, which could be useful for verification or security purposes.

**Future Work:**

1. **Deployment on Embedded Systems:**

Port the system to low-power devices like Raspberry Pi with camera modules for real-world, remote field applications.

1. **Cloud-Based Monitoring Dashboard:**

Create a web interface to view alerts, system status, and stored footage in real-time.

1. **Multi-Camera Support:**

Expand the system to monitor multiple camera feeds simultaneously for larger coverage areas.

1. **Add Logging and Evidence Capture:**

Enable automatic saving of detected frames or video clips for future reference or evidence.

**CONCLUSION**

The development of a Real-Time Animal Detection and Alert System showcases the potential of combining **computer vision**, **deep learning**, and **automated communication** to solve real-world challenges. By utilizing YOLOv5 for object detection and Twilio API for instant SMS notifications, the system provides a robust and responsive solution for environments where animal presence can pose risks or require immediate action.

With its real-time detection, high accuracy, and hands-free alert mechanism, this project not only reduces the need for manual surveillance but also enhances safety and situational awareness. It demonstrates how intelligent automation can be achieved through accessible, open-source technologies and minimal hardware requirements.

This system lays the groundwork for future improvements such as night-vision support, cloud-based logging, GPS tracking, and integration with mobile or IoT platforms—making it a scalable and adaptable solution across various sectors like agriculture, security, and urban management.

In conclusion, this project is a practical step toward smarter, safer monitoring systems that think, detect, and communicate faster than ever before.

**REFERENCES**

1. **YOLOv5 by Ultralytics** – <https://github.com/ultralytics/yolov5>
2. **OpenCV (Open Source Computer Vision Library)** – <https://opencv.org/>
3. **PyTorch Deep Learning Framework** – <https://pytorch.org/>
4. **Twilio Programmable Messaging API** – <https://www.twilio.com/docs/sms>
5. Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). *YOLOv4: Optimal Speed and Accuracy of Object Detection*. arXiv:2004.10934.
6. Redmon, J., & Farhadi, A. (2018). *YOLOv3: An Incremental Improvement*. arXiv:1804.02767.
7. Rosebrock, A. (2021). *Deep Learning for Computer Vision with Python*. PyImageSearch.