International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 04 | April - 2023 Impact Factor: 8.176 ISSN: 2582-3930

Anomaly and Object Detection for Security Surveillance

Rohaan Advani[1], Yash Dhake[2], Dhruva Kapadia[3] , Dr. Vahida Z. Attar[4]

[1]Student, Computer Engineering, COEP Technological University
[2]Student, Computer Engineering, COEP Technological University
[3]Student, Computer Engineering, COEP Technological University
[4] Associate Professor and Dean: School of Computational Sciences, COEP Technological University

---

Abstract - Human operators' inability to concentrate on live footage adequately and consistently is a limitation of existing video surveillance systems. After twenty minutes of attention, operators monitoring a single monitor often lose 95% of their focus and judgment. Nowadays, with most surveillance systems having a few dozen cameras, it is obvious that the surveillance duty is beyond the capacity of humans. To increase the effectiveness of video surveillance, this project aims to develop a smart system that will automatically highlight unusual activity and alert the appropriate authorities. The issue is framed as a classification problem, and the objective is to distinguish between typical and unusual conduct in public contexts. Additionally, this project makes use of computer vision and deep learning technology to find numerous things for monitoring. The methods used in this project can broadly be divided into the region- and model-based methods. This has applications in the security and military domains.

Key Words: Computer Vision, Security Surveillance, Deep Learning

1. INTRODUCTION

Anything that deviates from what is standard, typical, or anticipated is referred to as an anomaly. Anomaly detection locates questionable activity that deviates from your known patterns of usual conduct. Anomalies may include invasions of privacy, criminal activity, or unusual patterns in human behavior. Recognizing a crime as having been committed, identifying a suspect, and collecting adequate evidence to bring the suspect before a court are the three stages of crime detection. Finding patterns in data that deviate from expected norms based on prior observations is the challenge of anomaly detection. The capacity to identify or spot unusual behavior can offer extremely valuable insights across sectors.

Detecting instances of semantic objects of a certain class (such as people, buildings, or cars) in digital photos and videos is the task of Object Detection, a branch of computer science linked to computer vision and image processing. Face and pedestrian detection are two well-studied object detection areas. Several computer vision fields, such as image retrieval and video surveillance, use object detection. In computer vision tasks like image annotation, vehicle counting, activity recognition, face detection, face identification, and video object co-segmentation, it is frequently employed. It can also be used to monitor moving objects, such as a cricket bat or a ball during a game of football, or it can be used to track people in videos. The test photos are frequently drawn from a different data distribution, which considerably increases the difficulty of the object detection task. Numerous unsupervised domain adaptation techniques have been presented to deal with the issues brought on by the domain difference between training and test data.

It is possible to communicate with and manage a computer system without making direct physical contact by using gesture control, which is the capacity to recognize and interpret movements of the human body. Imagine a scenario where someone creating a presentation could add a quote or move a picture with a flick of the wrist as opposed to a mouse click. Soon, we will be able to interact with virtual reality as effortlessly as we can in the real world, using our hands to perform small, complex actions like picking up a tool, pressing a button, or squeezing a soft object front of us. The technology of this nature is still being developed. Yet, the computer scientists and engineers working on these projects assert that they are close to making hand and gesture detection systems for widespread use, much like many people already use speech recognition to dictate texts or computer vision to identify faces in photographs.

Pose estimation is one such means of gesture control. It is the process of employing a machine learning model to infer a person's pose from an image or a video by inferring the locations of important body joints (key points). Pose estimation models can be used to track an object or person in real-world environments, including multiple people. They can have an advantage when compared to object detection models, which can locate objects in an image but only offer coarse-grained localization with a bounding box enclosing the object.

Now we will introduce region-based anomaly and object detection methods. A geofence is a digital boundary for a physical geographic region. A geofence may be built dynamically (such as a circle around a certain area) or it may match a specified set of limits (such as school zones or neighborhood boundaries). Geofencing is the use of a geofence, and one example of this is the entry or exit of a geofence by a location-aware device of a location-based service (LBS) user. The geofencing strategy is founded on the observation that users travel from one location to another before remaining there for some time. This approach combines the user's awareness of their current location with their awareness of potentially interesting sites nearby. These interesting sites within a fence boundary can be referred to as points of interest or POI and the paths taken from one POI to another are referred to as activity paths. This activity could trigger an alert to the device's user as well as messaging to the geofence operator. This info, which could contain the location of the device, could be sent to a mobile telephone or an email account.

A Quadtree is a tree data structure that is used to represent a two-dimensional space recursively. For purpose of this paper, this space is the video frame array itself. It is often used to efficiently partition a space so that search and manipulation operations can be performed more quickly. In a Quadtree, each

node represents a rectangular region in space. The root node represents the entire space, and each child node divides its parent region into four equal parts or quadrants. This process continues recursively until each leaf node represents a sufficiently small region of space. Quadtree nodes can be used to represent different kinds of data, such as points, lines, or polygons (also referred to as vectors). For example, in a point quadtree, each leaf node represents a single point in space. To perform a search for points within a specific region, the search algorithm traverses the tree, testing each node to see if it intersects the search region. If a node does not intersect the search region, the algorithm can skip its subtree and move on to the next node. Quadtrees have many applications in computer graphics, image processing, and spatial databases.

An Octree is a tree data structure commonly used in computer graphics, computer vision, and other fields to efficiently represent three-dimensional objects or scenes. For purpose of this paper, this space is the video frame array along with the respective pixel depth values. The octree divides a three-dimensional space into smaller subspaces, with each node in the tree representing one of these subspaces. Each node in the tree has eight child nodes, which correspond to the eight smaller subspaces obtained by dividing the parent subspace in half along each of its three axes. Octrees are useful for routing algorithms, spatial indexing, collision detection, ray tracing, and other operations that involve identifying which objects or parts of objects are in a particular area of 3D space. By recursively subdividing the space into smaller and smaller subspaces, an octree allows for efficient queries that can quickly narrow down the search to a specific subset of the objects in the scene.

To deploy an octree in a computer vision project, methods to gain pixel-depth information must be understood. Depth estimation is the process of inferring the distance of objects in an image or a scene. It is a crucial task in computer vision, with applications in robotics, autonomous driving, and augmented reality. Depth estimation can be used in object detection to improve the accuracy and reliability of the detection. By estimating the depth of objects in an image, we can determine their relative positions and sizes, which can help in distinguishing between objects and background clutter. For instance, in a scene with multiple objects, the depth information can help identify which objects are closer to the camera and which are farther away, enabling more accurate object localization. There are two methods for depth estimation. Since it is challenging to determine the depth of points using a single point of view, the first requires many cameras (often 2 or 3). The second employs model-based techniques that use two-dimensional images as input and provide pixel depth values or graphic representations like depth maps.

YOLO (You Only Look Once) is a popular deep learning-based object detection algorithm because it is fast and can detect objects in real-time video streams. YOLOv5 comes in four main versions: small (s), medium (m), large (l), and extra-large (x), each offering progressively higher accuracy rates. Each variant also takes a different amount of time to train. YOLO accomplishes this by treating object detection as a regression problem, predicting bounding boxes and class probabilities directly from raw pixels in a single pass through the neural network. This makes it more efficient and faster than other object detection algorithms that require multiple passes through the network. YOLO is also known for its high accuracy and ability to detect small objects.

OpenCV Python provides a range of functions to perform security surveillance tasks. Motion detection refers to the process of identifying changes in the position of objects in a video sequence over time. Background subtraction, on the other hand, is the process of separating the foreground (moving objects) from the background (stationary objects). OpenCV provides several algorithms for background subtraction such as MOG, MOG2, and GMG. These algorithms work by comparing each frame in a video to a reference background frame and then subtracting the differences to create a mask image that highlights the foreground objects.

With the help of all the above concepts, we have built gesture-controlled software to perform video anomaly and object detection for security surveillance. This paper is aimed to discuss the proposed solution and experimental setup required to build such software.

2. PROBLEM STATEMENT AND ARCHITECTURE

Create an Anomaly and Object Detection for Security Surveillance. Deep learning models must be trained to detect illegal items, disasters, breaches of privacy, crimes, or unusual patterns of human/vehicle intent/behavior. This project is a vision of a generalized video security surveillance system. Its tools may be configured as per the location of the video and application domain. The project will include the following contents:

Anomaly Detection: 1. Abuse 2. Arrest 3. Arson 4. Assault 5. Burglary 6. Explosion 7. Fighting 8. Falling 9. Road Accidents 10. Robbery 11. Shooting 12. Shoplifting 13. Stealing 14. Vandalism 15. Motion Detection 16. Human Body Tracking 12 17. Punching 18. Kicking 19. Overcrowding 20. Overspeeding 21. Group Analytics (Meet, Split) 22. Smoke.

Object Detection: Objects to be detected can broadly be classified into three categories, dynamic (human body, vehicles, and objects in motion), static(furniture, devices, tools, walls, and other objects at rest), and semi-static. 1. Illegal Items – Weapons / Tools 2. Disasters – Fire, Flood 3. Vehicle Detection and Number Plate Recognition (Surveillance Purposes).

Additional Tools: 1. Quad division frame analysis 2. Dynamic Virtual Fencing (Using Convex Hull) 3. Notification and Alarm system 4. Location Mapping 5. Satellite Surveillance Imagery 13 6. Drone Routing using Depth Estimation 7. Edge Detection and Background Subtraction 8. Visual Question and Answering 9. Youtube Video API using CamGear 10. Android Video API using IP Webcam app. 11. Optical Character Recognition (OCR).

User Interface: Gesture Controller using Mediapipe Hand Tracking Module.

Testing: Saved / Online Videos, Webcam, IP Webcam Application.

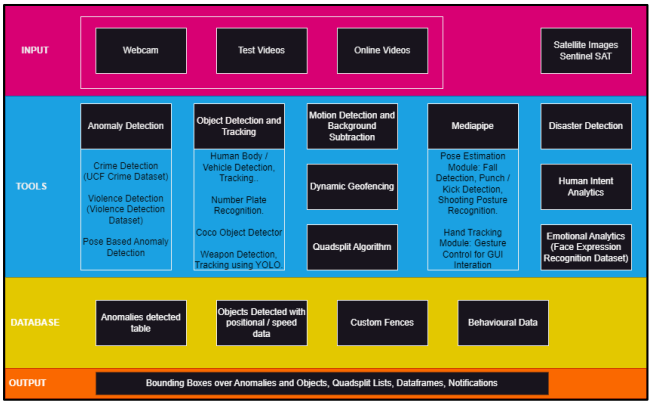

Fig 2.1: Software Architecture

3. PROPOSED METHOD AND EXPERIMENTAL SETUP

3.1. Quadsplit (A Quadtree-based Image Division Algorithm):

This algorithm is a frame analytics tool. It can be used to split an image/video frame into 4 equal parts (top right/top left/bottom right/bottom left). Each part of the image is added to an array representation of the quadtree data structure. The user gets to decide up to which level the quadsplit may occur. This algorithm aims to identify which part of the image has a particular anomaly/object. Also, testing models on smaller frames is far less time-consuming than on bigger frame matrices. Smaller segments of photos that contain anomalies and objects can be recorded in the database with positional information relative to the larger frame.

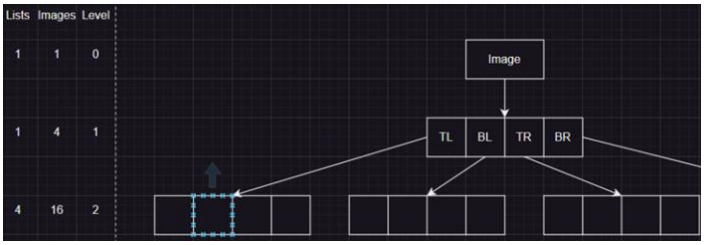

Fig 3.1: Quadsplit Algorithm Representation

3.2. Virtual Fencing:

It is a useful security surveillance tool. It gives the user the means to create a virtual boundary in each frame. This boundary can be used for various purposes such as identifying breaches in an area or keeping livestock within the fence. Objects and anomalies can be detected inside the fence separately. Moreover, it can be used to keep track of human behavior within a fixed area. In our project, we have gone a step further and created a dynamic geofence that alters in shape/area as per the user's requirements. Users can control the fence using gesture control. The project includes two geofencing modules, shape-based geofencing, and list-based geofencing. Shape-based geofencing enables users to build rectangular, triangular, or circular boundaries and detect objects within. List-based geofencing enables the user to add two-dimensional pixel coordinates (x, y) to a list data structure and build geofences in custom shapes and designs. One problem faced during this procedure is the convex hull problem. The convex hull problem is a common problem in computational geometry and can be used for geo-fencing applications. The problem involves finding the smallest convex polygon that contains a set of points in a plane. This convex polygon is known as the convex hull of the points. In geo-fencing, the convex hull can be used to define a boundary around a set of geographic points. For example, if you have a set of GPS coordinates representing a region of interest, you can find the convex hull of those points to create a polygonal boundary around that region. This boundary can then be used as a geo-fence to restrict the movement of objects or people within the region.

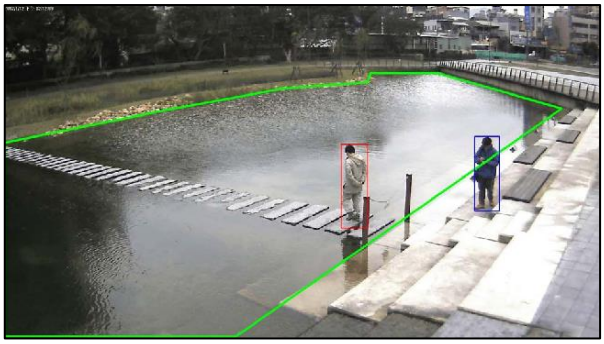
Fig 3.2: Virtual Fencing Representation

3.3. Motion detection and background subtraction:

These features are vital parts of a security surveillance system. The motion detection algorithm aims to find the absolute difference between two consecutive frames to mark changes in the positions of persons/objects. The background subtraction feature goes a step further and only displays the moving parts of the frames. These can be used for burglary alarm systems among other applications.

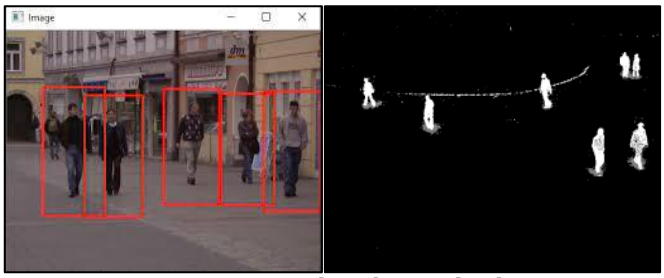

Fig 3.3: Motion Detection and Background Subtraction

In this project, we have also performed edge detection as an additional feature in motion detection. The most used technique for edge detection is the Canny edge detection algorithm. This algorithm works by first applying Gaussian smoothing to the input image to remove noise, then calculating the image gradient to find areas of rapid intensity change, and finally, applying non-maximum suppression and hysteresis thresholding to obtain the final edge map.

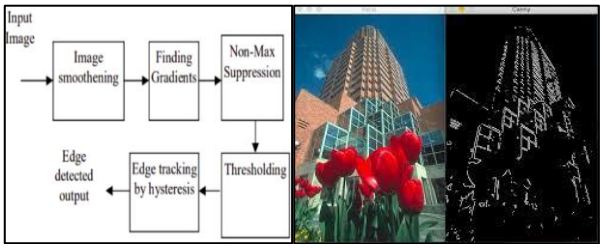

Fig 3.4: Edge Detection Algorithm and Edgemap

3.4. Human body detection and tracking:

This program uses available Haar/Hog cascades to identify human bodies in each frame and mark bounding boxes over them. The count of these boxes can be maintained separately. This facilitates overcrowding detection; the max count can be set by the user depending on the area and location of the given window. Overcrowding can be a sign of mass movements/protests. By gaining the positional data of these bounding boxes, these human bodies can be tracked in two-dimensional space. This facilitates applications such as stillness detection, walking detection, running detection, and checking if security personnel/ workers are at designated positions. The cascade gives us larger bounding boxes in case multiple humans are in a group. By calculating the difference of bounding box areas in consecutive frames, we can identify group activities such as meets and splits. Human body detection can also be done using pose estimation which will be explained in further sections.

Fig 3.5: Pedestrian Detection

3.5. Vehicle detection and tracking:

This program also uses cascades/models to detect and mark bounding boxes on vehicles. Can be used for counting/tracking applications in the same way as humans. Tracking can be used for overspeed detection/road accident detection purposes. Lastly, number plate recognition can be used to keep a record of entering/exiting vehicles. This code performs object recognition and Optical Character Recognition to detect and store number plates. Image processing using OpenCV is done and then a Haar Cascade Classifier is used to detect number plates in a frame. The number plate is then cropped out and the resultant image is then sent for Optical Character Recognition (OCR).

Fig 3.6: Vehicle and Number Plate Recognition

3.6. Weapon/Illegal item detection:

For the weapon detection module, we have decided to use the YOLOV5 object detection model. Weapon detection can be done using two methods, object detection-based methods and visual question-and-answer methods. Object detection-based methods take the video frame as input, search for weapons or illegal items and mark bounding boxes over them. On the other hand, visual questions and answers enable users to ask questions as to the presence of illegal items in the video frame.

Fig 3.7: Weapon Detection

3.7. Hand Tracking:

A hand tracking module is a mediapipe tool that can be used to track hands/fingers and multiple landmark points in a hand. This feature has been used in our project as gesture control. This enables the user to interact with the application without physical contact with the machine. The project uses this feature to interact with the user interface. Can be used for features such as virtual zoom/drag and drop also. The hand tracking module includes functionalities such as marking and drawing the landmarks, building the landmark list, calculating distances between any two given landmarks, and checking which fingers on the palm is open or closed.

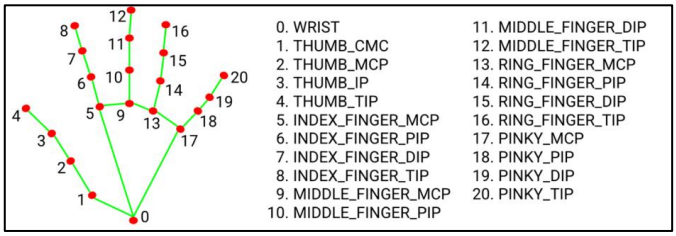

Fig 3.8: Hand tracking and hand landmarks

3.8. Pose Estimation:

Like hand tracking, a pose estimator is a tool that can identify human bodies, their postures, and various landmarks on the body. This feature can be used for various purposes such as identifying human posture. This is used in applications such as fall detection, and punch/kick detection. The pose estimation module includes functionalities such as marking and drawing the landmarks, building the landmark list, calculating distances between any two given landmarks, and calculating the angle between any three landmarks. These angles can be used to identify standing and lying down postures. If the time difference between these 2 postures is less than a few seconds, a fall can be detected. Similarly, angles of the arms and legs have been used to identify punches and kicks. Display bars have been designed to check if a punch or kick has been completed. These bars alter as per the direction of motion of the arms and legs. Additionally, percentage completion values and counts of punches and kicks have been displayed.

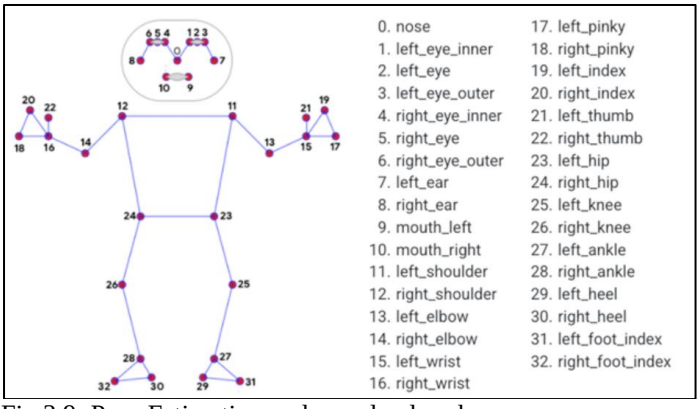

Fig 3.9: Pose Estimation and pose landmarks

3.9. Fire Detection:

Fire detection is a must for a surveillance system. Disaster management and on-time reporting can save lives. To perform this task, the project converts the frames into HSV colorspace. The HSV or Hue, Saturation, and Value of a given object is the colorspace associated with the object in OpenCV where Hue represents the color, Saturation represents the greyness and Value represents the brightness and it is used to solve the problems related to computer vision because of its better performance when compared to RGB or Red, Blue and Green color space and the Hue range in HSV is [0,179], the Saturation range in HSV is [0,255] and the Value range in HSV is [0,255] and to perform object detection, finding the range of HSV is necessary. Adding the HSV range for fire, we can detect it in images/video frames.
Additionally, as part of this project, we have added a smoke detection model since smoke is a strong indicator of a possible fire situation. The YOLOv5m model is used for smoke detection in computer vision due to its capability to accurately detect objects in real time. It is a medium-sized version of the YOLOv5 model, which is optimized for a good balance between accuracy and computational efficiency. Firstly, the YOLOv5 model is loaded from a pre-trained checkpoint named 'keremberke/yolov5m-smoke'. Next, various parameters of the YOLOv5 model such as the confidence threshold for non-maximum suppression (NMS), IoU threshold for NMS, the class-agnostic setting for NMS, multiple labels per box allowance, and the maximum number of detections per image are set. The YOLOv5 model is then used to perform inference on the image, and the results are stored in the 'results' variable. The predicted bounding boxes, scores, and categories are parsed from the results. The 'results.show()' function is used to display the detected bounding boxes on the original image.

Fig 3.10: Fire and Smoke Detection

3.10. COCO names object detector:

One of the most popular object detection models trained on the COCO dataset is the COCO name object detector. This model uses a deep convolutional neural network (CNN) to extract features from images and then applies a region proposal network (RPN) to generate candidate object locations. This object detector can be used to detect 91 classes of objects for surveillance purposes. This model has been used to test the region-based methods of our project such as Quadsplit and Geofencing. For time being, all objects in the COCO dataset are being detected and marked using bounding boxes in the video frame either in the entirety of the frame or in specific regions. This can be configured to detect specific classes of objects based on their application domain. This includes a feature of depth estimation. This model has been devised from the below research paper.
'GLOBAL-LOCAL PATH NETWORKS FOR MONOCULAR DEPTH ESTIMATION WITH VERTICAL CUTDEPTH' - *Doyeon Kim, Woonghyun Ka, Pyunghwan Ahn, Donggyu Joo, Sewhan Chun, and Junmo Kim.* The paper proposes a novel approach for improving monocular depth estimation from a single image using convolutional neural networks. The approach includes a hierarchical transformer encoder, lightweight decoder, and selective feature fusion module.
Following are the experimental steps for the same:
1. Convert the NumPy image to a PIL image.
2. Extract features from the PIL image.
3. Use the model to predict depth values of input features.
4. Format output depth values to build the depth map.

3.11. Database module:

Pandas is a popular data manipulation library for Python that provides powerful data structures such as series and data frames for working with structured data. One advantage of using Pandas tables as a database is that they offer a lot of functionality for data manipulation and analysis. In this project, we have made use of Pandas tables to store outputs of our various applications and respective anomalies detected. These data frames are interactive in nature and can be used to both save and load data. The major application of data loading used in our project is drawing custom geofences on test/online/android videos. This is done using the respective frame IDs. Frame IDs are being used to distinguish between webcam, test videos, Youtube videos, and Android videos. We have the following data frames with respective fields:
1. Anomaly: Frame ID, Timestamp, Anomaly type
2. Fire: Frame ID, Timestamp, Fire-type (Fire, smoke)
3. Fence: Frame ID, Timestamp, Video Name, Coordinates (Coordinates of the fence)
4. Tracker: Frame ID, Timestamp, Number of objects to track, Distance, Speed, and Direction lists
5. Quadsplit: Index, Frame ID, Timestamp, Leaf Nodes
6. Pedestrian: Frame ID, Timestamp, Anomaly
7. Vehicle: Frame ID, Timestamp, Number Plate
8. Weapons: Frame ID, Timestamp, Weapon.

3.12. Youtube Video API:

VidGear is a powerful video processing library that provides a simple and flexible way to capture, process, and stream video frames from various sources, including webcams, network streams, and video files. It is designed to work with OpenCV, an open-source computer vision library, and provides a high-level interface for video processing tasks. Integrating VidGear with YouTube is a straightforward process that involves utilizing the YouTube API to fetch video data and using VidGear to process the video frames. First, you'll need to obtain API credentials from Google Developers Console and install the VidGear and Google API Client libraries using pip. Next, you can use the YouTube Data API to fetch video data, including the video ID and its URL. Once you have the video URL, you can use VidGear to open the video stream, process the video frames as per your requirements, and display the video output.

3.13. Android API:

Connecting an Android phone camera to Python using OpenCV and IP Webcam can be a useful way to access your phone's camera from your computer and use it for various computer vision applications. Here are the steps to connect your Android phone camera to Python using OpenCV and IP Webcam:
1. Install the IP Webcam app on your Android phone from the Google Play Store.
2. Open the IP Webcam app and scroll down to the "Start Server" section. Tap on "Start Server" to start the server.
3. Once the server is started, the app will display a URL that you can use to access the camera feed on your phone.
4. Open your Python editor and import the necessary libraries. You will need to import the cv2 (OpenCV) library, as well as the urllib.request library to access the camera feed.
5. Run the code and you should be able to see the camera feed from your Android phone on your computer.

3.14. Visual QnA model:

ViLT is a VLP model that can process both image and text inputs without using convolutional neural networks (CNNs) or region supervision, which are commonly used in previous VLP models. The architecture of ViLT is based on the Transformer model, which has shown great success in natural language processing tasks. ViLT uses a minimal visual embedding pipeline and follows a single-stream approach, which means that it processes image and text inputs together in the same sequence. The program makes use of the ViLT model to get a description of the contents of a frame. Context-specific questions are asked for every frame based on the anomalies being checked for, and alerts are raised if the model returns positive.
Following are the steps to deploy a visual QnA model:
1. Initialize the model and question set.
2. Convert the NumPy array to a PIL image.
3. Ask questions to detect specific anomalies in PIL images.
4. Extract the output answer from the result list.
5. Check the answer to display if an anomaly detected.

3. CONCLUSIONS

We use anomaly detection and object detection for security surveillance in our project. It is a significant and beneficial use of machine learning in the security industry. The goal of the project is to recognize and track numerous objects in real-time video surveillance streams while also looking for any anomalies that might point to security issues. Advanced computer vision techniques will be used in this project. By combining both anomaly detection and object detection techniques for various potential security threats, the project has shown promising results in detecting abnormal events and tracking specific objects in different environments. This can help to enhance security measures and provide quick responses to potential threats, while also reducing false alarms and minimizing human error. Overall, this project demonstrates the potential of using machine learning and computer vision to improve security surveillance and aid in real-time decision-making. With further development and integration into existing security systems, the techniques used in this project could have a significant impact on improving public safety and security.

REFERENCES

[1] Roberta Vrskova, Robert Hudec, Peter Sykora, Patrik Kamencay, Miroslav Benco. "Violent Behavioral Activity Classification Using Artificial Neural Network", 2020 New Trends in Signal Processing (NTSP), 2020.

[2] Jinnan Hu, Guo Li, Haolan Mo, Yibo Lv, Tingting Qian, Ming Chen, Shenglian Lu. "Crop Node Detection and Internode Length Estimation Using an Improved YOLOv5 Model", Agriculture, 2023.

[3] Jun-Horng Chen, Teng-Hui Tseng, Chin-Lun Lai, Sheng-Ta Hsieh. "An Intelligent Virtual Fence Security System for the Detection of People Invading", 2012 9th International Conference on Ubiquitous Intelligence and Computing and 9th International Conference on Autonomic and Trusted Computing, 2012.

[4] "Human-Computer Interaction. Multimodal and Natural Interaction", Springer Science and Business Media LLC, 2020.

[5] Anand R, Reddi Vivek Vardhan, Katam Jayacharan Kalyan, Boyapalli Sree Kanth Reddy. "Designing a Wearable Jacket for the Visually Impaired People", 2022 6th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS), 2022.

[6] Cucchiara, R., C. Grana, A. Prati, and R. Vezzani. "Computer vision system for in-house video surveillance", IEE Proceedings - Vision Image and Signal Processing, 2005.

[7] "Computer Vision", Springer Science and Business Media LLC, 2017.

[8] Fenglin Liu, Xian Wu, Shen Ge, Xuancheng Ren, Wei Fan, Xu Sun, Yuexian Zou. "DiMBERT: Learning Vision-Language Grounded Representations with Disentangled Multimodal-Attention", ACM Transactions on Knowledge Discovery from Data, 2021.
[9] "Intelligent Computing and Information and Communication", Springer Science and Business Media LLC, 2018.

[10] Zaher Al Aghbari. "cTraj: efficient indexing and searching of sequences containing multiple moving objects", Journal of Intelligent Information Systems, 2011.

[11] Erxun Zhao, Yang Zhang, Jingmin Gao. "Monocular Depth Estimation of Noncooperative Spacecraft Based on Deep Learning", Journal of Aerospace Information Systems, 2023.

[12] "Computer Vision – ACCV 2012", Springer Science and Business Media LLC, 2013.

[13] Rahmaniar, Wang, Chen. "Real-Time Detection and Recognition of Multiple Moving Objects for Aerial Surveillance", Electronics, 2019.

[14] Young-Jin Cha, Wooram Choi, Gahyun Suh, Sadegh Mahmoudkhani, Oral Büyüköztürk. "Autonomous Structural Visual Inspection Using Region-Based Deep Learning for Detecting Multiple Damage Types", Computer-Aided Civil and Infrastructure Engineering, 2017.