

omar 227750 1.docx

 British University in Egypt

Document Details

Submission ID

trn:oid:::11892:279748857

Submission Date

Apr 29, 2025, 12:19 AM GMT+3

Download Date

Apr 29, 2025, 12:20 AM GMT+3

File Name

omar 227750 1.docx

File Size

7.2 KB

6 Pages





1,007 Words

6,196 Characters




11% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

- 
9 Not Cited or Quoted 11%
 Matches with neither in-text citation nor quotation marks
- 
0 Missing Quotations 0%
 Matches that are still very similar to source material
- 
0 Missing Citation 0%
 Matches that have quotation marks, but no in-text citation
- 
0 Cited and Quoted 0%
 Matches with in-text citation present, but no quotation marks

Top Sources

- 7%  Internet sources
- 3%  Publications
- 7%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- 9** Not Cited or Quoted 11%
Matches with neither in-text citation nor quotation marks
- 0** Missing Quotations 0%
Matches that are still very similar to source material
- 0** Missing Citation 0%
Matches that have quotation marks, but no in-text citation
- 0** Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

- 7% Internet sources
- 3% Publications
- 7% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Submitted works	Middlesex University on 2025-04-11	3%
2	Publication	R. N. V. Jagan Mohan, B. H. V. S. Rama Krishnam Raju, V. Chandra Sekhar, T. V. K. P...	1%
3	Internet	www.openj-gate.com	1%
4	Submitted works	Khalifa University of Science Technology and Research on 2025-04-26	1%
5	Internet	www.mdpi.com	1%
6	Submitted works	University of Sunderland on 2024-01-11	1%
7	Internet	studmed.uio.no	<1%
8	Submitted works	Cranfield University on 2024-04-18	<1%

Name: Omar Nagy

ID:227750

Anomaly Detection CCTV

1. Introduction

Anomaly detection in CCTV footage has become an essential tool for modern security systems. Traditionally, human operators would manually review hours of video, a time-consuming and error-prone task. However, with advancements in computer vision and machine learning, automated systems can now detect suspicious activities in real time, making surveillance more efficient and responsive. This report explores a computer vision-based anomaly detection system using YOLOv8 for detecting loitering and unusual activities in CCTV footage.

The aim of this project is to design a system that can analyze real-time video streams or recorded footage, identify individuals, track their movements, and detect anomalies, specifically loitering, using YOLOv8-based object detection. This system can be deployed in various surveillance applications, alerting security personnel to potential threats, enabling proactive responses to prevent harm.

2. Literature Review

Existing Techniques

Traditional anomaly detection methods in CCTV footage include background subtraction, optical flow, and frame differencing. These methods analyze the differences between frames to detect motion, but they are susceptible to high false

positive rates in dynamic environments. More advanced methods leverage machine learning, particularly deep learning, to improve accuracy. CNNs (Convolutional Neural Networks) and object detection models, such as YOLO (You Only Look Once), have become popular due to their ability to perform real-time object detection with high accuracy.

YOLOv8

YOLO is a deep learning-based object detection model designed to be fast and efficient for real-time processing. YOLOv8, the latest version of the YOLO series, improves upon previous versions with enhanced accuracy and speed. It is widely used in surveillance systems for detecting various objects in video frames, including people, vehicles, and other potential threats.

Loitering Detection

Loitering, or the act of lingering in a specific area without any apparent purpose, is often considered a sign of suspicious activity. Detecting loitering behavior in surveillance footage requires monitoring a person's position over time and determining whether they stay within a specific area for an extended period. This type of detection plays a critical role in identifying potential threats in environments such as shopping malls, airports, and public spaces.

3. Methodology

System Design

The system is designed to process CCTV footage (or webcam video), detect people using YOLOv8, and track their movements across frames. When a person stays in the same area for an extended period, they are flagged for loitering. The system also provides real-time feedback in the form of bounding boxes, tracking trails, and an information panel that shows the number of active tracks and loitering alerts.

Tools and Libraries Used

The system leverages the following tools:

OpenCV: A library for computer vision tasks such as video processing, image manipulation, and object tracking.

NumPy: A library for numerical computations, used for mathematical operations in the code.

YOLOv8: An object detection model from the ultralytics package, used for detecting people in video frames.

4. Implementation

Code Explanation

The CCTV Anomaly Detection class is at the core of the system. Here's a breakdown of the key functions:

Initialization (`__init__`):

Sets up the video source, output directory, and frame interval.

Initializes YOLOv8 for people detection and sets tracking parameters.

People Detection (`_detect_people`):

Converts each frame to RGB and uses YOLOv8 to detect people.

Extracts the bounding boxes and centers of detected people.

Tracking (`_track_people`):

Matches new detections with existing tracked people based on Euclidean distance.

Updates the tracks with new detections.

Loitering Detection (`_detect_loitering`):

Detects loitering by checking if a person stays in the same area for a specified time threshold (`loitering_time_threshold`).

Alerts are triggered when the person stays within a small area for longer than the threshold.

Drawing Tracking Trails (`_draw_tracking_trail`):

Draws lines between consecutive detection points to visualize the movement of tracked individuals.

Information Panel (`_add_info_panel`):

Displays an overlay with real-time information about the active tracks, loitering alerts, and total people detected.

Video Processing (`process_video`):

Captures the video feed, processes each frame, detects people, tracks their movements, and detects loitering.

Saves the processed video with overlays to the output directory and displays the frames in real time

Code Example:

Here's a simplified snippet of the code that detects people and tracks them across

frames:

```
def _detect_people(self, frame):
```

```
    rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
```

```
    results = self.yolo_model.predict(source=rgb_frame, imgsz=416, conf=0.5)
```

```
    people = []
```

```
    for result in results:
```

```
        for box in result.boxed:
```

```
            cls = int(box.cls[0])
```

```
            conf = float(box.conf[0])
```

```
            if cls == 0 and conf > 0.5:
```

```
                x1, y1, x2, y2 = map(int, box.xyxy[0])
```

```
                people.append({'bbox': (x1, y1, x2-x1, y2-y1), 'center': (x1 + (x2-x1)//2, y1 + (y2-y1)//2)})
```

```
    return people
```

5. Results and Discussion

Evaluation

The system was evaluated based on its ability to detect people, track their movements, and identify loitering. The YOLOv8 model provided accurate people detection, with bounding boxes drawn around detected individuals. The tracking mechanism efficiently followed people across frames, even in crowded scenes. Loitering detection worked well, triggering alerts for individuals who remained in the same area for more than the

specified threshold.

Performance Optimization

To optimize performance, the frame processing interval was adjusted based on the type of video source (webcam or video file). The YOLOv8 model was configured with a frame size of 416x416, balancing accuracy and speed. Further optimizations could include using a smaller model variant like YOLOv8n for faster processing in real-time applications.

Possible Improvements

Advanced Tracking: Integrating advanced tracking algorithms such as DeepSORT or SORT could improve person re-identification and tracking accuracy.

False Positives: The system occasionally flagged non-loitering individuals as loiterers due to short pauses. Implementing additional behavior analysis (e.g., direction changes) could reduce false positives.

Scalability: The system could be expanded to handle multiple cameras and larger surveillance areas, with a centralized monitoring dashboard for security personnel.

6. Conclusion

In conclusion, the CCTV Anomaly Detection system based on YOLOv8 has successfully demonstrated the potential of using deep learning for real-time surveillance and anomaly detection. By combining object detection with tracking and loitering detection, the system provides valuable tools for security personnel to identify suspicious behavior. While the system works effectively in controlled environments, further optimization and expansion are needed to handle more complex real-world scenarios, such as crowded spaces or varying environmental conditions.