Project 2 Report: Real-Time Object Detection and Tracking System

Prepared by: Ahmed Refaay
Date: November 24, 2024

---

1. Introduction

Project Title: Object Detection with YOLO for Home Security
Objective: To develop and deploy a real-time object detection and tracking system utilizing YOLOv11 and a home security RTSP camera feed. The program is designed to identify objects, save their images, and generate a summary report.

---

2. Methodology

1. **System Setup:**
    - Configured a Hikvision DVR system for RTSP feed streaming.
    - Integrated the YOLOv11 model with the video feed for object detection.
2. **Object Detection and Tracking:**
    - Used the YOLOv11 `track` function with `persist=True` for maintaining consistent object IDs across frames.
    - Implemented bounding box tracking and saved the largest cropped images for each object ID.
3. **Data Processing:**
    - Organized detected objects into a dictionary, categorizing them by ID and class.
    - Saved images with filenames containing ID, class, and timestamp for easy identification.
4. **Performance Metrics:**
    - The system processes the RTSP feed with ~1 GB CPU memory and ~500 MB GPU memory usage.
    - High accuracy in object detection with smooth real-time performance.
5. **Output:**
    - Cropped images of detected objects saved locally.
    - A summary report listing detected object IDs, classes, and timestamps.

---

3. Achievements

1. Implemented a robust real-time detection pipeline integrated with the home security feed.
2. Resolved frame synchronization and video saving issues, ensuring smooth performance.
3. Achieved high detection accuracy while maintaining efficient resource usage.
4. Saved object images systematically, enabling further analysis or reporting.

---

4. Limitations

1. **Camera System Issues:**
    - Poor lighting and camera obstructions (trees) reduce detection quality.
    - Cameras are not optimally positioned, affecting object visibility.
    - System shuts down during power outages without backup.
2. **Technical Challenges:**
    - Duplicate IDs for the same object across frames.
    - Irrelevant classes detected (e.g., airplanes, trains) due to pre-trained weights.
    - Image processing enhancements (e.g., sharpening, deblurring) not yet implemented.
3. **Functional Enhancements Needed:**
    - Scenario-based security planning for potential home intrusion detection.
    - Optimization of FPS to balance security level and resource usage.

---

5. Next Steps and Action Items

1. **Technical Enhancements:**
    - Implement IoU-based bounding box comparison or feature matching to resolve duplicate IDs.
    - Apply image processing techniques like sharpening, histogram normalization, and colorization.
2. **System Customization:**
    - Fine-tune YOLOv11 on a custom dataset with relevant classes.
    - Define specific security use cases and configure the system accordingly.
3. **Performance Optimization:**
    - Limit FPS to decrease resource usage without compromising detection reliability.
    - Test multi-threaded or multi-processing setups for better performance scalability.
4. **Deployment:**
    - Plan deployment on edge devices like NVIDIA Jetson for cost-effective operation.

---

6. Required Resources and Studies

1. **Resources:**
    - Backup power supply for uninterrupted operation.
    - Night vision-capable cameras for better low-light performance.
2. **Tools:**
    - Image processing libraries (OpenCV, scikit-image).

    - Dataset preparation tools (Roboflow, Label Studio).
3. **Learning Areas:**
    - Advanced object tracking algorithms.
    - YOLO fine-tuning techniques.
    - Edge AI deployment with TensorRT or OpenVINO.

---

7. Attachments

- Final code implementation: Project2.py
- Example output images: Available in the local `outputs/images` directory.