Final Project: Object Detection and Video Recording with OpenCV

Objective: Develop a program that utilizes OpenCV to detect any kind of object within a video stream and simultaneously record the video with the identified objects highlighted.

Requirements:

* Programming proficiency in Python
* Familiarity with the OpenCV library
* Access to a webcam or video file

Project Phases:

Phase 1: Setting Up the Environment

1. Install Python and OpenCV on your computer.
2. Ensure you have a webcam or access to a video file.
3. Create a project directory to organize your code and data.

Phase 2: Object Detection Algorithm Selection

1. Research and select an appropriate object detection algorithm, as learned during the course.
2. Familiarize yourself with the algorithm's documentation and input/output formats.

Phase 3: Video Processing and Object Highlighting

1. Import the necessary libraries, including OpenCV and the chosen object detection algorithm.
2. Initialize the video capture object using the webcam or video file path.
3. Define a function to process each video frame: a. Capture the current frame from the video stream. b. Employ the object detection algorithm to identify objects within the frame. c. Draw bounding boxes around the detected objects using OpenCV's drawing functions. d. Display the object class and confidence score for each detected object.

Phase 4: Video Recording and Output

1. Initialize the video writer object, specifying the output file name, codec, frame rate, and frame dimensions.
2. Write each processed frame to the video writer object.
3. Display the processed frame in a window for real-time feedback.

Phase 5: User Interaction and Termination

1. Implement a mechanism to gracefully terminate the program upon user input, such as pressing the 'q' key.
2. Release the video capture and video writer objects once the program is finished.
3. Close all OpenCV windows to prevent resource leaks.

Evaluation Criteria:

1. Completeness: The program should successfully detect objects, highlight them, and record the video.
2. Accuracy: The object detection algorithm should accurately identify and classify objects.
3. Performance: The program should process frames efficiently and maintain a smooth playback rate.
4. User Experience: The program should be easy to use and provide clear feedback to the user.
5. Code Quality: The code should be well-documented, structured, and adhere to best practices.

Additional Challenges:

1. Implement object tracking to follow the detected objects across frames.
2. Experiment with different object detection algorithms to compare their performance and accuracy.
3. Explore techniques to improve the visual appearance of the highlighted objects and labels.
4. Develop a graphical user interface (GUI) to enhance user interaction and provide additional features.