Multi-Person Pose Tracking with YOLO

This project includes a Python script for object detection using YOLOv8 and real-time human pose tracking with Mediapipe. The script leverages OpenCV for video processing and TensorFlow for efficient GPU memory management. It overlays pose landmarks and connections dynamically on video frames, allowing for both object detection and detailed human pose estimation.

Features:

- Real-time detection of human body poses using Mediapipe.
- Object detection in video frames using YOLOv8.
- Pose landmarks and connections are drawn in dynamic colors.
- Supports video file input or camera feed for live pose tracking.

Requirements:

Before running the scripts, ensure that the following Python packages are installed:

• **Python 3.x**	: Recommended Python 3.8 or higher
• **OpenCV**	: For image and video processing.
• **Mediapipe**	: For human pose detection.
• **NumPy**	: For array and matrix operations.
• **TensorFlow**	: To enable GPU memory growth.
• **Ultralytics YOLOv8**	: For object detection.

You can install these dependencies by running:

```
pip install opencv-python mediapipe numpy tensorflow ultralytics
```

Make sure that you have a **GPU** available for faster processing, especially for YOLOv8.

Usage:

Code: Pose Tracking with YOLO Object Detection

This script performs object detection with YOLOv8, crops detected objects, and then applies pose tracking to the cropped regions.

1. **Initialize YOLOv8 and Mediapipe**:
   - YOLOv8 detects objects in the video, while Mediapipe detects poses within the cropped regions.
2. **Process the video frames**:
   - The frames are resized, and objects are detected using YOLOv8.
   - For each detected object, a region of interest (ROI) is extracted, and pose estimation is applied.

3. **Overlay pose landmarks**:
    - Pose landmarks are drawn for each detected object, with different colors for each connection.
    - Both the main frame (with bounding boxes and pose landmarks) and a blank frame showing only the pose landmarks are displayed.

Code Breakdown:

Code:

1. **Initialize YOLOv8 and Mediapipe**:
    - The YOLO model is loaded, and dynamic memory growth is enabled for TensorFlow to handle GPU memory effectively.
2. **Object Detection**:
    - YOLOv8 is used to detect objects in the video frames.
    - For each detected object, the coordinates of the bounding box are extracted.
3. **Pose Estimation on Cropped Objects**:
    - After detecting an object, the region within the bounding box is cropped and pose estimation is applied using Mediapipe.
    - Pose landmarks and connections are drawn on the cropped image.
4. **Overlay Pose Landmarks**:
    - The cropped image with pose landmarks is overlaid onto the main frame.
5. **Display Video**:
    - The main video frame is displayed with both detected objects and pose landmarks.
    - Another window displays only the pose landmarks on a black background.
6. **Exit Condition**:
    - Press 'x' to exit the video feed.

Contributing

If you'd like to contribute to this project, feel free to fork the repository, make improvements or submit issues. Contributions such as additional features, bug fixes, or optimizations are always welcome!