Documentation for Pedestrian Tracking and Analysis Code

1 Overview

The provided code processes videos from a specified folder, extracting frame-by-frame information about pedestrians. It detects pedestrians, tracks them across frames, and determines whether they are walking or standing, looking or not looking, their speed, and the direction of motion. The results are saved into output video files and annotations are stored in text files.

2 Key Components and Variables

2.1 Input and Output Folders

- `input_folder`: Directory containing the input video files.
- `output_folder`: Directory where the processed video files will be saved.
- `output_annotation_dir`: Directory where annotation files will be saved.

2.2 Counters and Thresholds

- `videos_count`: Counts the total number of videos processed.
- `frame_count`: Counts the total number of frames processed.
- `total_pedestrians`: Total number of pedestrians detected in all frames.
- `transition_counter`: Buffer counter to confirm status changes.
- `transition_threshold`: Threshold for buffer time to confirm status change.
- `movement_threshold`: Threshold for movement detection.
- `ped_num_per_frame`: List to store the number of pedestrians detected per frame.

2.3 Functions

- `calculate_distance(point1, point2)`: Calculates the Euclidean distance between two points.
- `get_direction(point1, point2)`: Determines the direction of movement based on the change in position.
- `estimatespeed(Location1, Location2)`: Estimates the speed of movement based on the distance covered between frames.
- `write_to_txt(file_name, data)`: Writes the given data to a text file, either as a dictionary or as plain text.

2.4 Model Loading

The code imports necessary libraries and loads the YOLOv8 model for pose estimation.

2.5 Video Processing

The code loops over each video file in the specified input folder. For each video:

- Initializes a video capture object and retrieves the video properties such as frame width, height, and FPS.
- A video writer object is created to save the output video.

2.6 Frame-by-Frame Processing

For each frame in the video:

- The YOLO model is used to detect and track pedestrians, extracting bounding boxes and keypoints.
- The code calculates the walking/standing status, looking/not looking status, speed, and direction of each detected pedestrian.
- Maintains histories of object positions to track movement across frames.

2.7 Annotations and Display

- Bounding boxes and annotations (walking/standing, looking/not looking, speed, direction) are drawn on the frames.
- The annotated frame is displayed in a window, and the original frame is written to the output video file.

2.8 Saving Results

After processing all frames, the video capture and writer objects are released. Annotations are saved to text files.

3 Main Arrays and Their Usage

3.1 `boxes`

Array of bounding boxes for detected pedestrians in each frame.

- Used to draw rectangles around detected objects and for movement analysis.

3.2 `keypoints_data`

Array of keypoints data for all detected persons.

- Contains keypoints for body parts, used to determine looking/not looking status and movement analysis.

3.3 `object_histories`

Dictionary storing historical positions of each tracked object.

- Helps track the movement of objects across frames and analyze their movement.

3.4 `prev_walking_status`

Dictionary storing the previous walking/standing status of each object.

- Used to apply hysteresis in status changes and smooth out the walking/standing determination.

3.5 `Directions`

Dictionary storing the direction of movement for each object.

- Stores the current direction of each object, used for annotations.

3.6 `Speeds`

Dictionary storing the speed of each object.

- Contains the estimated speed of each object, used for annotations.

3.7 `ped_num_per_frame`

List storing the number of pedestrians detected per frame.

- Helps keep track of pedestrian count statistics across frames.

3.8 `transition_counter`

Counter used for confirming status changes.

- Buffers the transition between statuses to avoid rapid fluctuations.

3.9 `transition_threshold`

Threshold value for the transition counter.

- Determines how many frames are considered before confirming a status change.

3.10 `movement_threshold`

Distance threshold for determining movement.

- Used to differentiate between walking and standing statuses based on the distance moved.

3.11 `ids`

Array of unique IDs for each detected object.

- Used to track objects consistently across frames, ensuring the correct association of historical data.

4 Summary

This documentation provides an overview and detailed explanation of the code's functionality, key components, and main variables. It should help in understanding how the code processes video files, tracks pedestrians, and saves the results.