Documentation of the automated traffic monitoring system

This prototype shows computer vision applications using OpenCV for analyzing and detecting moving objects in traffic videos. Starting with library installation and imports, the focus shifts to video processing and object detection. For these prototypes the city has sent 2 videos, this prototype has to run on these videos.

Requirements for the prototype:

The requirement consists of developing an application for detecting and tracking moving cars from a camera recording. Apart from that, is also require to develop a computer vision application for counting the number of cars that go from the city's downtown to the city Centre for a specific time interval.

Explanation on the approach taken for the requirements

- **Requirement 1**

  Involves frame-by-frame video analysis to identify and count moving objects. Techniques like background subtraction and various image processing methods (Gaussian blur, thresholding, morphological operations) are applied to enhance detection accuracy.

- **Requirement 2**

  Extends the analysis to multiple videos, emphasizing the presentation of results. It counts cars, calculating the total number and rate per minute, displayed in a tabular format for videos analyzed. This systematic approach showcases the practicality of applying specific criteria to filter movements and accurately detect objects within traffic video data.

	Total number of cars	Cars per minute
Traffic_1.mp4	6	2.02338
Traffic_2.mp4	4	2.27101

Frame Extraction and Noise Reduction

- **Initial Step**: Utilizing ***cv2.VideoCapture()*** from OpenCV, videos are processed frame by frame.
- **Noise Mitigation:** To enhance the quality of each frame for analysis, a Gaussian blur ***(cv2.GaussianBlur())*** is applied. This step effectively smooths out high-frequency components, such as noise and sharp transitions, preparing the frames for more sophisticated processing.

Object Detection via Background Subtraction

- **Background Subtraction:** The ***cv2.createBackgroundSubtractorKNN(***) function establishes a method for distinguishing moving objects from their static background. The KNN (K-nearest neighbors) algorithm was chosen after evaluating various options for its effectiveness in noise reduction and maintaining object contours over time.
- **Foreground Mask Refinement:** The subtractor's output is further processed with ***cv2.threshold()*** to minimize noise, followed by ***cv2.morphologyEx()*** to close gaps within detected foreground objects.

Contour Processing and Object Identification

- **Contour Identification:** Using ***cv2.findContours(),*** the shapes of moving objects are identified. Objects deemed insignificant or outside the area of interest (the lower portion of the image) are filtered out, based on a predefined minimum area threshold (***minimum_area_contour***).
- **Shape Handling:** For the detected relevant objects, ***cv2.convexHull()*** is applied to accommodate their convex nature, leading to the creation of bounding rectangles around these objects using ***cv2.boundingRect().*** These rectangles are then visually represented on the original frames with ***cv2.rectangle()***

Enhanced Tracking with the Tracker Class and Kalman Filter

- **Tracker Class Implementation:** In requirement 2, an advanced tracking system is introduced via a ***"Tracker"*** class, which systematically records and updates the positions of objects across frames, based on detected contours.
- **Kalman Filter Integration:** For more accurate tracking, a Kalman filter ***(cv2.KalmanFilter())*** is utilized. This sophisticated algorithm acts as a predictive tool, analyzing the movement history of objects to forecast their future positions. It proves particularly useful for ensuring continuous tracking of objects, even when they momentarily vanish from the frame, by providing estimated positions in lieu of direct observations. This significantly enhances the tracking accuracy and reliability.

In conclusion both prototypes have the same background extraction to keep track of moving object. Then in the prototype 2 is implemented other types of tools that let us keep count of the cars that are going to the center of the city. These prototypes showcase a comprehensive approach to video analysis in traffic monitoring, from initial video frame processing to advanced

object tracking. Through careful application of image processing techniques and the strategic use of algorithms, the files demonstrate the power of computer vision in real-world applications.