Autonomous Car

Lane

&

Object intersection **detection**

Orian Dabod

308337062

2023

**OpenCV**

(Open Source **Computer Vision**) is an open-source library of programming functions mainly aimed at **real-time** computer vision. The library includes a vast number of algorithms and tools that can be used to perform various computer vision tasks, such as image and video processing, object detection, tracking, recognition, and more.

Originally developed by Intel, OpenCV is now maintained by a large community of developers and researchers, and it is available under a Apache 2 license.

OpenCV supports multiple programming languages, including C++, Python, Java, and MATLAB, and it can run on a variety of operating systems, including Windows, macOS, Linux, and Android. This makes it a flexible and versatile option for developing computer vision applications even on autonomous vehicles.

OpenCV is widely used in various industries, such as robotics, automotive, healthcare, and more, for applications such as autonomous vehicles, medical imaging, security and surveillance, and more.

**OpenCV Advantages:**

* OpenCV is an open-source library, which means that it is free to use, modify, and distribute. This makes it an accessible and affordable option for developers and researchers.
* OpenCV is widely used in the computer vision industry, and as a result, it has a large community of developers and users. This means that there is a wealth of resources available for learning, troubleshooting, and expanding the library.
* OpenCV supports a wide range of programming languages. This makes it a versatile choice for developers who may have different language preferences or requirements.
* OpenCV has a vast collection of image and video processing algorithms and tools. This includes tools for object detection, tracking, recognition, and more. These tools can save developers a significant amount of time and effort in implementing computer vision applications.
* OpenCV is platform-independent, which means that it can run on a variety of operating systems. This makes it a flexible option for developing applications across different platforms.

**OpenCV Disadvantages:**

* OpenCV can be complex and difficult to learn, especially for beginners. The library has a vast number of functions and modules, and understanding how to use them effectively can require significant effort and expertise.
* Some of the algorithms and tools provided by OpenCV may not be suitable for all use cases or may not produce accurate results in some scenarios. This means that developers may need to implement custom algorithms or modify existing ones to meet their specific requirements.
* OpenCV may not have as many features or capabilities as other computer vision libraries or frameworks, such as TensorFlow or PyTorch. This may limit its usefulness in some applications.
* OpenCV does not have native support for deep learning models, which are the current state-of-the-art solutions for many taks.

**Object Detection**

A computer vision task that involves identifying and localizing objects in an image or video. The goal is to identify the presence and location of specific objects within an image.

Object localization involves identifying the location of objects in an image, often by drawing a bounding box around the object.

Object detection is a crucial task in many applications, including self-driving cars, security and surveillance, robotics, and healthcare. For example, in self-driving cars, object detection is used to identify other vehicles, pedestrians, and obstacles on the road. In security and surveillance, object detection can be used to identify suspicious activity or to track the movement of people and vehicles.

There are many different techniques and algorithms used for object detection, including deep learning-based methods such as Faster R-CNN, YOLO. These methods have achieved state-of-the-art performance in object detection tasks by leveraging convolutional neural networks (CNNs) to learn and recognize object features.

**Line detection**

Line detection is a common computer vision task that involves detecting and extracting lines from an image. The lines can be straight or curved and may represent edges, contours, or other features of interest in the image.

Some common techniques include the Hough transform, Canny edge detection, and gradient-based methods.

Line detection is an essential computer vision task for autonomous cars because it enables the car to navigate and stay in its lane. Autonomous cars use a variety of sensors, such as cameras, LiDAR, and radar, to perceive their surroundings and make decisions based on that information. Cameras are particularly important because they can provide high-resolution images that allow the car's computer to detect and track lane markings.

By using line detection algorithms, the car's computer can identify the lane markings on the road and determine the position and orientation of the car within the lane. This information can then be used to control the car's steering, acceleration, and braking to keep the car in its lane and maintain a safe distance from other vehicles.

Line detection is also important for lane departure warning systems, which can alert the driver if the car is drifting out of its lane. In addition, line detection can be used to detect obstacles, such as other vehicles or pedestrians, that are crossing the lane, which can help the car avoid collisions and ensure the safety of passengers and other road users.

**Code Explanation:**

The code is a python script for performing lane detection and object detection in videos using OpenCV.

We start a loop over a video file (or directory containing frame images of video) located at `VIDEOS\_LOCATION`, AKA we iterate over the frames of the video.

Each frame is first resized to a width of RESIZE\_TO pixels, this improves performance for all video sizes.

Next, we apply the Canny edge detection algorithm (popular technique for this task developed by John Canny in 1986)

Next, we apply region of interest (ROI) mask in order to find lines only on the actual road

The Hough transform (feature extraction technique to detect simple shapes such as lines and circles in an image. proposed by Paul Hough in 1962) is then applied to the frame to detect lines in the image.

Next, we apply an averaging method to the detected lines from the current frame (which pass slop threshold). If no lines detected, we use previous frame line position.

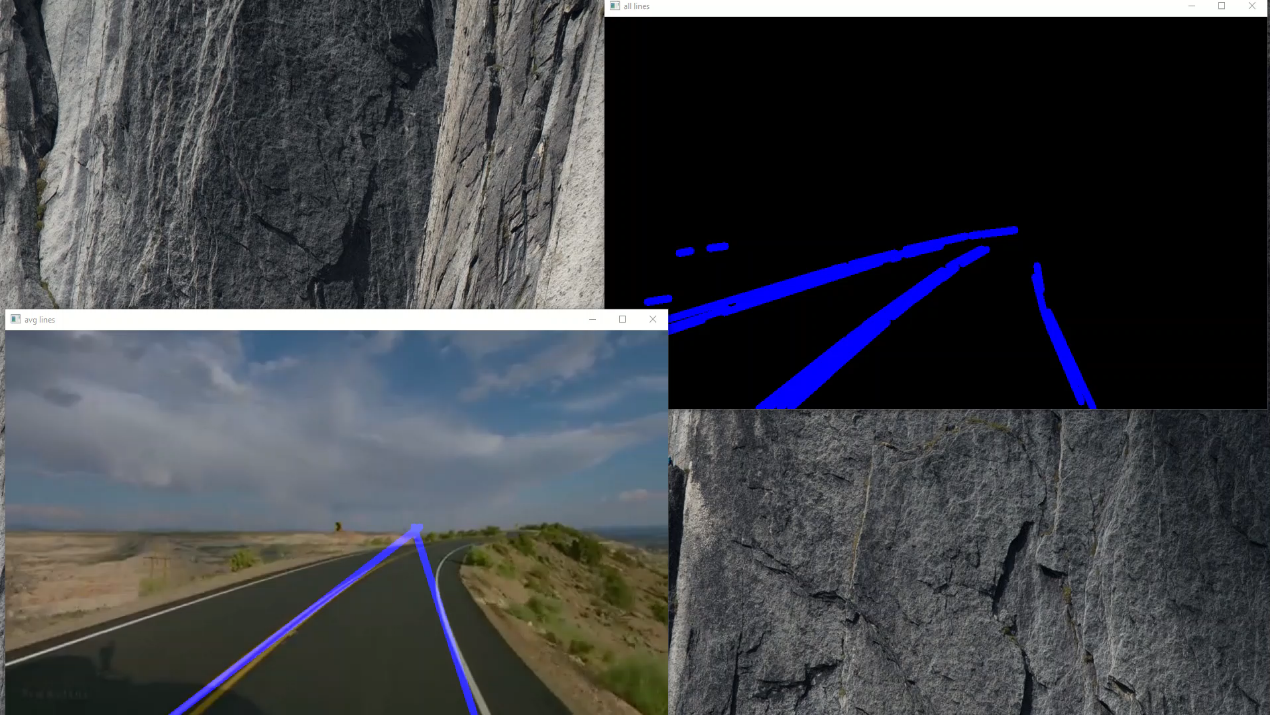
If this is not the first frame with lanes detected, then the script applies object detection to the area within the ROI mask by comparing the Canny image of the current frame to the Canny image of the previous frame. The resulting contours (i.e., detected objects) are divided by color if they intersect with our lane then added to the frame.

**Usage**

1. Run `pip install -r requirements.txt`.
2. Change `VIDEOS\_LOCATION` parameter in main.py to the desired video or image directory (should be under <project-root>/data directory).
3. Select which view you want to see by simply commenting out the wanted views:
   1. resize / canny / canny cropped / all lines / avg lines / final with object detection.
4. Run the program using the command `python main.py`.
5. Windows will pop-out with the views you asked for.

**Hyper-parameters:**

1. VIDEOS\_LOCATION -> the desired video or image directory
2. RESIZE\_TO -> resizing input image (noticing `region\_of\_interest` function which may need to be change also)
3. Rectangle variable under `image\_manipulation.py` will change ROI in frame for lane detection task (for different camera locations)
4. Slope threshold in function `average\_slope\_intercept` under `lines.py`, will remove any lines with lower slop (change for semi horizontal roads / more straight ones (highways) )
5. Under function `extract\_cnts` on `lanes\_and\_objects.py` there are many params, which should be changed to detect more/less objects (depends on the road conditions)

**Wassah (וואסח)**



