CAR LICENSE PLATE DETECTION

EMNA METHNANI

MATRICOLA : 533499

License plate detection is a computer vision technology that is used to automatically detect and read license plates from images or video footage. It is typically used in applications such as traffic management, parking management, law enforcement, and toll collection.

License plate detection technology has been developed over the years with the advancement of computer vision and machine learning techniques. Early license plate recognition systems were developed in the 1970s and 1980s, and were based on simple image processing techniques. However, with the advent of more powerful hardware and sophisticated algorithms, modern license plate detection systems are able to achieve much higher levels of accuracy and efficiency.

Today, license plate detection technology is widely used in many different industries and applications. It has become an essential tool for law enforcement agencies, parking management companies, toll operators, and many other organizations that need to quickly and accurately identify vehicles and track their movements.

In this Project we devoloped a python code to detect car license plate, this code uses the OpenCV library to detect a license plate in an image. Here's a brief overview of what each part of the code does:

1) Load the image using the cv2.imread() function and store it in the variable img.
2) Convert the image to grayscale using the cv2.cvtColor() function and store it in the variable gray.
3) Apply a Gaussian blur to the grayscale image using the cv2.GaussianBlur() function and store it in the variable blur. This helps reduce noise in the image.
4) Apply adaptive thresholding to the blurred image using the cv2.adaptiveThreshold() function and store the thresholded image in the variable thresh. This binarizes the image, making it easier to find contours.
5) Find contours in the thresholded image using the cv2.findContours() function and store them in the variables contours and hierarchy.
6) Sort the contours by area using the sorted() function and keep only the top 10 largest contours. Store them in the variable contours.
7) Loop through each contour in contours. For each contour:

a. Calculate its perimeter using the cv2.arcLength() function.

b. Approximate the contour using the cv2.approxPolyDP() function. This reduces the number of points in the contour while still retaining its shape.

c. Check if the approximated contour has 4 vertices. If it does, it's likely to be the license plate contour, so store it in the variable plate_contour and break out of the loop.

8) If plate_contour is not None, draw it on the original image using the cv2.drawContours() function.

9) Display the resulting image with the license plate contour using the cv2.imshow() function. Wait for a key press using the cv2.waitKey() function, and then close the window using the cv2.destroyAllWindows() function.

The algorithms used in this code are:

1. Gaussian blur: This is a widely used image processing technique that helps to smooth out the image and reduce noise. It is used in this code to preprocess the image before thresholding.
2. Adaptive thresholding: This is an image processing technique used to create a binary image from a grayscale image. It is used in this code to create a binary image of the license plate region after preprocessing.
3. Contour detection: This is an algorithm that is used to detect the boundaries of objects in an image. It is used in this code to find the boundary of the license plate region in the binary image.
4. Sorting and filtering of contours: The algorithm filters out small contours and sorts the remaining contours in decreasing order of their areas. It then selects the largest contour that has four corners and assumes it to be the license plate.
5. Drawing contours: Once the license plate contour is detected, the algorithm uses the cv2.drawContours() function to draw the boundary of the license plate on the original image.

Overall, the algorithm in this code is a combination of several image processing techniques and contour detection.