License Plate Extracting Project

By: Tyler Abbott & Lewis Crabtree

Abstract

License plates all around the world play an important role as a means of identifying and authenticating the operator controlling a motorized vehicle. As an example, when a critical incident/moment occurs between a registered vehicle and the surrounding environment, the license plate plays a vital role in ensuring that the subsequent action or consequence is met with the registered vehicle's primary operator. Typically in Canada and most areas within the United States, a registered vehicle has 6 alpha-numeric possibilities. Given this information, there are 2,176,782,336 possible combinations of license plates per province or state. For this reason, it is key that during the process of image processing and analysis, that the extraction of a license plate is accurate and readable.

<u>Sample Method of Extracting License Plate Information:</u>

1. Image acquisition
2. Image processing:

a. Conversion to grayscale

b. Apply bilateral filter to blur the background of the image

c. Apply contrasting routines to enhance the contrast of the image

d. Apply Canny edge detection algorithm

3. Find all contours in the image
4. Filter out the contours that are too small
5. Filter out contours that are not rectangular
6. Present the top-most contour

Technical discussion

Final Product:

```
#input the image and convert it to RGB
image = cv2.imread('input.jpg')
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Convert the image to GrayScale.
gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)

# Blur image to reduce detail in background
blur = cv2.bilateralFilter(gray, 7, 95, 95)

# Use cv2.canny to get the edges in the image
edges = cv2.Canny(blur, 40, 200)

# Find contours
contours, new = cv2.findContours(edges, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)

# Use 20 largest contours
contours = sorted(contours, key=cv2.contourArea, reverse=True)[:20]

#find contour that is a rectangle
plate = gray
for c in contours:
    perim = cv2.arcLength(c, True)
    edges = cv2.approxPolyDP(c, 0.02 * perim, True)
    if len(edges) == 4:
        x,y,width,height = cv2.boundingRect(c)
        plate = gray[y:y+height, x:x+width]
        break

cv2.imwrite('plate.jpg', plate)
```

Techniques used

The first step in our program is to load input images into opencv. For bulk processing of images we used ‘glob’ to load all .jpg images in an input folder. Next, the image is converted from BGR to grayscale to simplify edge detection by using only a single channel. To reduce background detail in the image and reduce unnecessary background noise, a bilateral filter was applied to the image. The bilateral filter is similar to Gaussian

Blur, in the way that neighbouring pixel values are weighted and applied to the center pixel, with an additional measure that only sums pixel values that are within a certain value range. This causes less blurring to be applied across edges, which in our case keeps edges intact so that they can be more easily detected. To detect the edges in the image we used Canny. cv2.Canny has two parameters to specify cutoff values for strong, weak, and suppressed pixels. Strong pixels are considered part of an edge, while weak pixels are only considered part of an edge if it is neighbouring a strong pixel. The output of Canny is an image with only the edges left. This binary image with just the edges is input to a method that converts the edges into contour objects. A contour is a set of lines that join points in an image with the same intensity values. To find the contours in the image we used the opencv method findContours. findContours has parameters to specify what types of contours to return, as well as what mode to use to retrieve the contours. In our case we used retr_list to return a list with all contours in the image. We also used chain_approx_simple to simplify redundant points in a line to only the endpoints of the line. We then sorted the contours from largest to smallest and used only the 20 largest contours. For each contour, c, the perimeter is calculated with cv2.arcLength. A polygon is drawn around the contour with cv2.approxPolyDP. If the number of edges this polygon has is 4, then there is a good chance that it is either a rectangle, or a rectangle captured in the image at an angle. We considered the largest quadrilateral detected from these contours to be the license plate. Further work could be done to apply classification or deep learning to our program to improve results.

Discussion of results

Notable Findings:

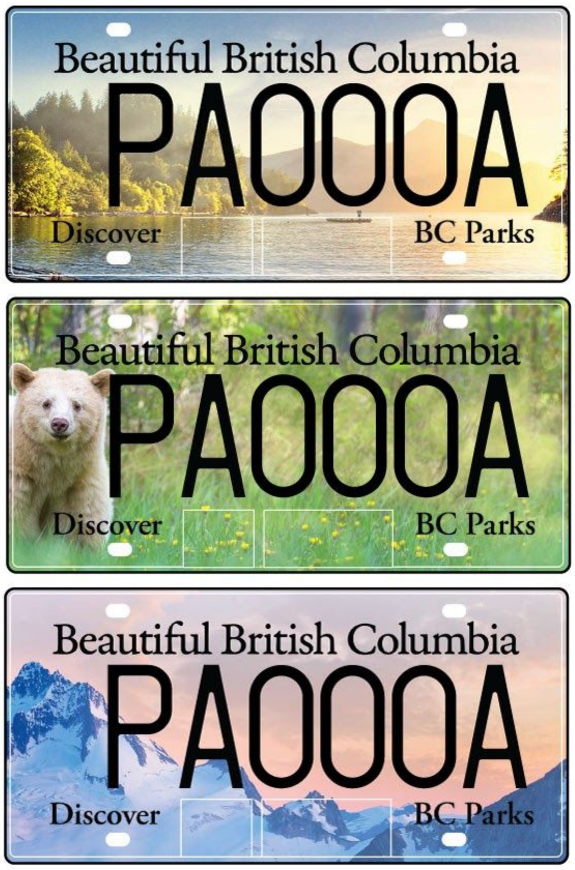
License plates differ per region: As mentioned previously, North America and European countries have wildly different shapes of license plates. North American plates have rounded edges, and have around a 2:1 ratio in terms of width and height of the given plate. European plates have sharper edges, making the shape more rectangular, and is wider than Canada's domestic plates at 20.5 inches by 4.3 inches (~4.8:1 ratio). Due to the nature of the European plate's specifications, it is debatably easier to detect, as rounded corners may be detected as a blob rather than a sharp rectangle.

Other procedures attempted:

Conditional contour height / width ratio check: In early iterations of this project, an idea came to fruition to check whether any of the listed contours within the image met the condition that width and length ratio is similar to a standard registration license plate. A license plate in Canada is typically 15 centimetres by 30 centimetres, giving this license plate a 2:1 ratio. As an idea, it would eliminate big rectangle contours that are not the license plate, and would 'skip' past these contours, it did not work as expected. There was too much deviation between different license plates, as North American license plates are far different in comparison to European license plates, thereby restricting this project to one or the other.

Applying dilation after edge detection: By applying dilation after the edge detection process, it will make the white edges thicker, depending on the kernel size given for the dilation function. Applying a small dilation would assist the edges to connect, if the edges did not connect to complete a contour prior to dilation. Making it slightly bigger would yield different results on each image provided. This yielded tremendous results, but was not included in the final product.

In addition to this phenomenon, North American license plates can have decorative enhancements or plate protectors that may distort the true shape of a traditionally white license plate.


Beautiful British Columbia
PA000A
Discover
BC Parks
Beautiful British Columbia
PA000A
Discover
BC Parks
Beautiful British Columbia
PA000A
Discover
BC Parks

(New Decorative ICBC License Plates [2])

Low contrast images and thresholding: Thresholding is a useful tool for enhancing the distinct changes in contrast or intensity. In theory, this would assist with edge detection, as the stark difference between the edges of the license plate and the bumper of the vehicle will allow for the ability to show the object starts and ends. However, problems arise as colors and intensities are manipulated. Glare, color of vehicle, color/decoration on the registration plate are just some of many problems. For example: If the color of the vehicle, and the color of the plate are matching, the intensity values are going to be similar and will blend together when thresholding. Glare and decoration may create artifacts on the plate, which may deviate the overall shape of the plate further from a rectangle and closer to a contour blob.

Results

Original Image

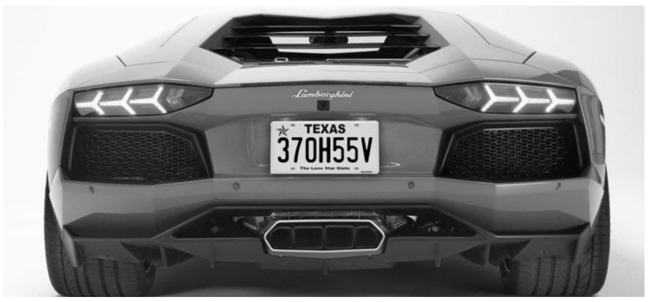
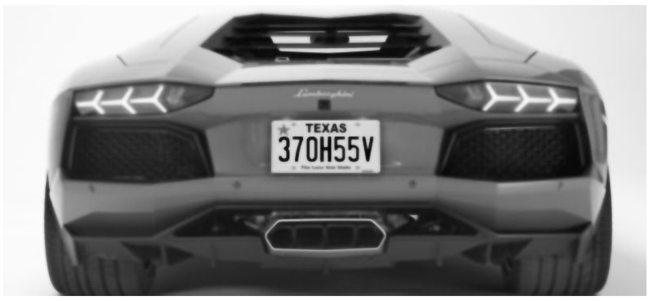
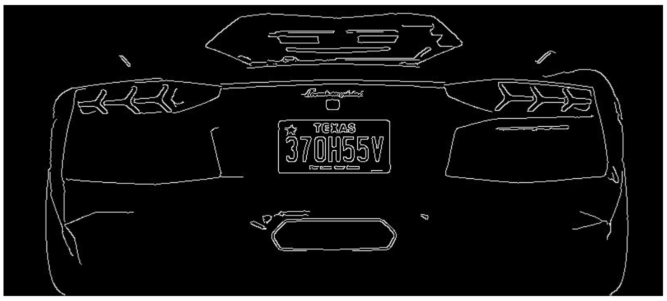


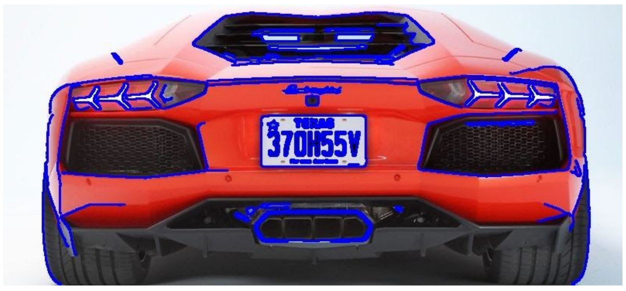
Image converted to gray scale



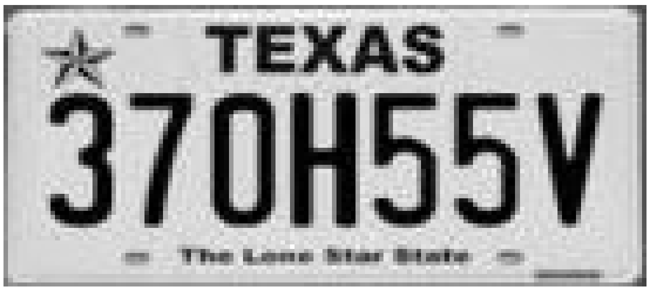
Bilateral Filter applied to image



Edges extracted from image



Contours drawn on the original image



License plate extracted from the image

Citations:

Sarbjit Kaur. (2014). *An Efficient Approach for Number Plate Extraction from Vehicles Image under Image Processing* [Scholarly project]. Retrieved June 02, 2020, from https://www.researchgate.net/publication/262843164_An_Efficient_Approach_for_Number_Plate_Extraction_from_Vehicles_Image_under_Image_Processing

CBC, T. (n.d.). New ICBC Decorative License Plates [Digital image]. Retrieved June 03, 2020, from https://www.cbc.ca/news/canada/british-columbia/license-plates-bc-parks-1.3941983

OpenCV Documentation. (n.d.). Retrieved June 01, 2020, from https://docs.opencv.org

Contributions / Roles:

We worked in tandem on this project. As we were developing our program we communicated frequently what was working and not working. As we approached completion we combined our techniques and tweaked our individual versions separately, and then discussed and changed our final version in accordance to what was working the best. We did this routine several times, with each of us making multiple improvements to our program.