Finding the Convex Hull with Application to Distance Measurements for a Formula Student Car

Benedikt Schlereth

June 16, 2020

Contents

1 Introduction

Formula-Student is an international design and construction competition where teams from all around the world are constructing and then racing single seat formula race-cars. With the new event of Formula-Student-Driver-less Teams are encouraged to develop their own self-Driving cars. My motivation is to kick-offthe development of our own Driver-less Vehicle by implementing the

rst step and providing a way to detect the boundaries of a race track. The tracklayout

2 Steps to extract the features

To find the cones inside a picture one could create a dataset of 100+ cones and then train a neural network. But working with a neural network brings the additional difficulties to choose a network type that is efficient to run on a mobile platform while being reliable. It is also not necessary for this type of application to detect different objects in one frame. With only cones in three different colors to detect it is quite simple to describe the features of them. To visualize the process i choose a scene with green cones an different lighting and ground. The picture is taken from a height of $1,2m$ and the distance between

Figure 1: example image

the cones is $2m$ with the nearest $2m$ and the farthest $10m$ away from the camera.

2.1 color filtering

The most obvious feature is the color. So the first step is to implement a colorfilter.

```
1 hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
2
3 # value for colorfilter
4 lower_green = np.array([30, 100, 0])
5 upper_green = np.array([90, 255, 255])
6
7 mask = cv2.inRange(hsv, lower_green, upper_green)
8 res = cv2.bitwise_and(hsv, hsv, mask=mask)
9
10 blur = cv2.GaussianBlur(res, (15, 15), 0)
11 return blur, mask
```

The image provided for the colorfilter needs to be converted from the RGB values to the HSV colorspace. The HSV describes the color not with an eight-Bit value for red, green and blue but with a circle where the first value is for the color with 0 beeing red and 255 blue. The second value is needed for the saturation and the third and last value describes how bright the color will be represented.

With knowing how the HSV-colorspace works it is possible to find the right values which will describe the color of our cones. It will be necessary to try the colorfilter for images in bright sunlight as well as cloudy conditions the make sure to be able to detect cones in every possible environment.

Next up is comparing every Pixel with our given value. With the OpenCV function "inRange" one image will be read and compared to our boundaries. It returns a binary image which means everywhere the color was in Range a one is remaining and everywhere else a zero.

2.2 contours

After the Color-filter we are left with a binary image. Now it is quit easy to detect the edges of where it changes from black to white.

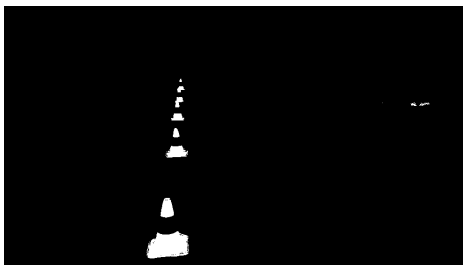

Figure 2: binary-image

But as seen in the top right hand corner of the image some noise will be left inside the picture. Therefore another filter is needed. A good difference between important and noisy areas is the size of the contour. This will delete all small contours which are most probably not important but also cones that are far away. The first test showed that excluding such cones does not matter.

2.3 hierarchy

Sometimes small dots or areas are left inside on parts of the cones. Luckily the contour-function gives us back the hierarchy of the contours. By filtering out the smaller regions inside the bigger ones no leaves only parts of the cones with no noise inside the areas.

2.4 convex-hull

The next step will be to find the convex hulls. What those are and how they are found will be described in the section 3. The convex-hull leaves a very good approximation of the cones edges as seen in Figure 3

Figure 3: convex-hull

2.5 aspect ratio

A nice feature of the cones is the round top part. So no matter at what angle or distance the convex hull is detected the ratio between the height and width stays the same.

2.6 bounding box

With every top part of the cones found now the bottom part needs to be added. Therefore the distance to every other contour is calculated and the nearest one below the top part belongs to the same cone. The maximum points of both contours give the outlines of a bounding box which is drawn with red-color in Figure 4

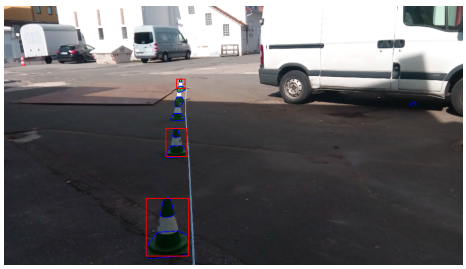

Figure 4: bounding-box