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

June 3, 2020

1 Introduction

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. For this application it is not necessary to detect different objects so it is also possible to describe the features of a cone and search for them inside a picture. To visualize the process i choose a scene with green cones an different lighting and ground. The picture is taken from a height

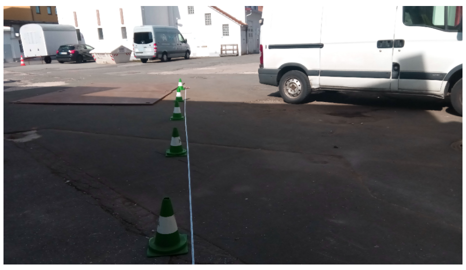

Figure 1: example image

of $1,2m$ and the distance between 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.

```
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

# value for colorfilter
lower_green = np.array([30, 100, 0])
upper_green = np.array([90, 255, 255])

```

```
mask = cv2.inRange(hsv, lower_green, upper_green)
res = cv2.bitwise_and(hsv, hsv, mask=mask)

blur = cv2.GaussianBlur(res, (15, 15), 0)
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.

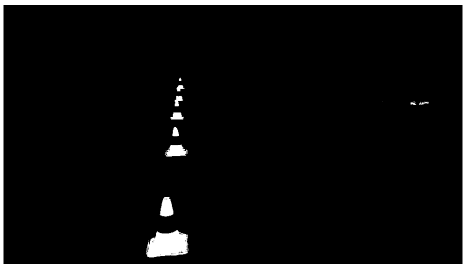

Figure 2: binary-image

But as seen in the top right hand corner of the image

2.3 convex-hull

2.4 aspect ratio

2.5 bounding box

3 convex-Hull

4 wrap-up