Visual Recognition

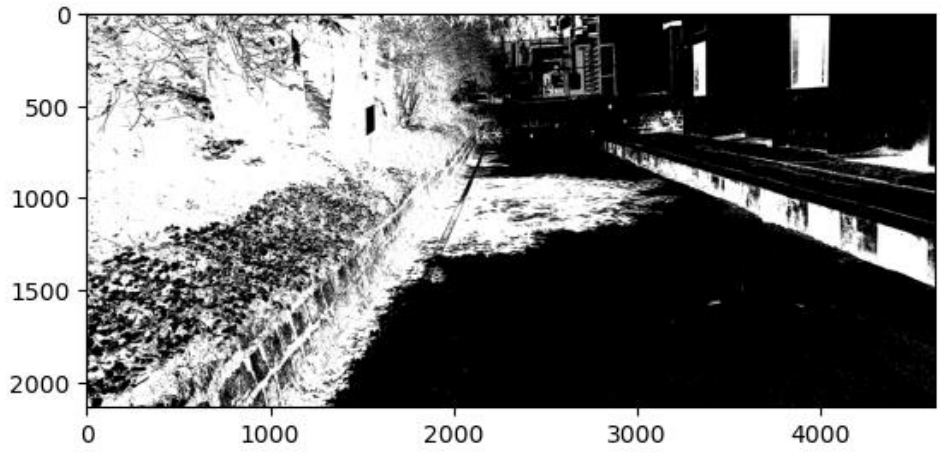
Assignment1

Q1b. Shadow Detection & Inpainting:

Used Image:

Pre Processing

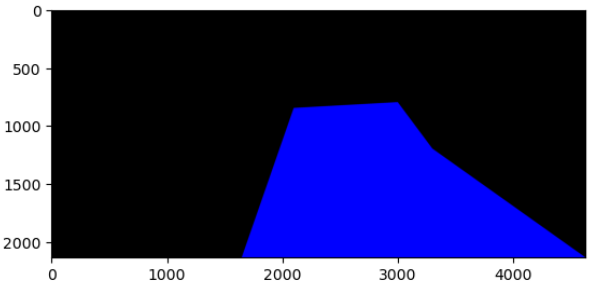
- First, I converted this image into RGB color scheme as openCV by default uses BGR color scheme.
- Then I converted this image into gray scale image and then into binary image named 'im' using thresh = 70 and maxval=255.
- Then created im2 (i.e. 255-im) and saved it as 'cb.jpg'.



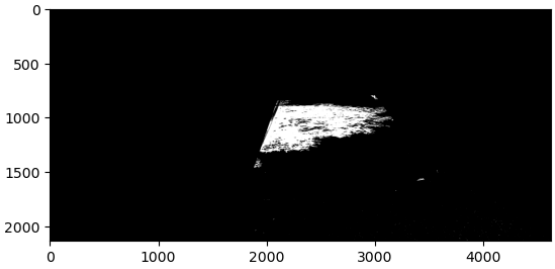
cb.jpg

- Here I first create an array of zeros which has the shape of the source image. Then I drew a polygon whose 6 corners' co-ordinates were (2100, 850),(1650, height of the source

image),(3700,height of the source image), (width of the source image, height of the source image),(3300,1200),(3000,800). This was the mask. Then I did bitwise_and between 'cb.jpg' and mask and then converted the resultant image into gray scale & basically got the below image.



Mask

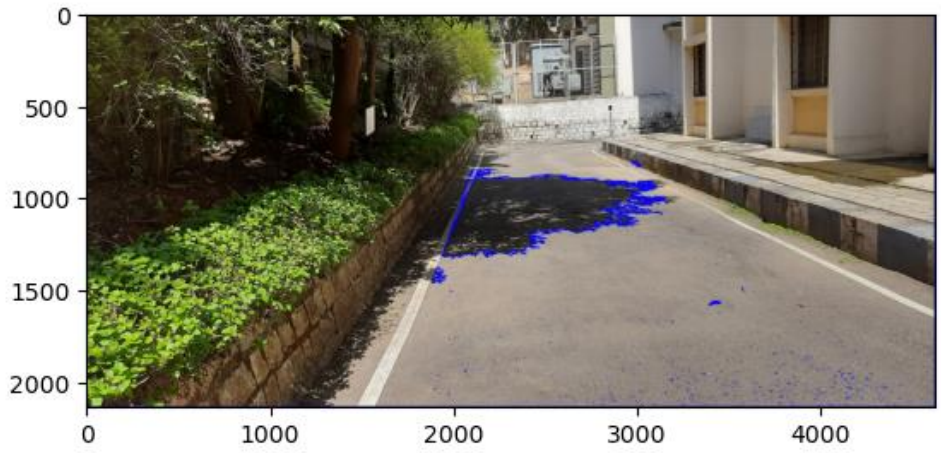


Resultant Image

- Resultant Image basically gives us the shadow region (white area 255 valued pixels) in the source image.

Shadow Detection:

Then I used **cv2.findContours()** on the resultant image and got list of 1172 contours in the image (Each individual contour is a Numpy array of (x,y) coordinates of boundary points of the object.) and stored them into a variable named contours. Then using **cv2.drawContours()** and contours variable as argument in it I detected the shadow region in the source image.



Shadow Removal:

After shadow detection for shadow removal, I basically used four methods. Two of them I did **manually** and I two of them I did using **cv2.inpaint()** function.

- In first manual method I cropped the road area without shadows of the image and took the average pixel value of that area and replaced those values in the shadow area's pixels using cv2.fillPoly() function.

- In second manual method I replaced the shadow region's pixel with a road pixel.

Using Inpainting,

- **In this method at first, I used cv2.inpaint() with cv2.INPAINT_TELEA algorithm and inpaintRadius as 50 using resultant image as the mask with the original source image and got the below image as output.**

- **In this method at first, I used cv2.inpaint() with cv2.INPAINT_NS algorithm and inpaintRadius as 50 using resultant image as the mask with the original source image and got the below image as output.**

Results:

Original

Method1

Method2

Method 3

Method 4