Assignment 1 (Color Blindness Visual Aid Device)

Name: Khaled AbdulRazek Moawad

Course: Image processing

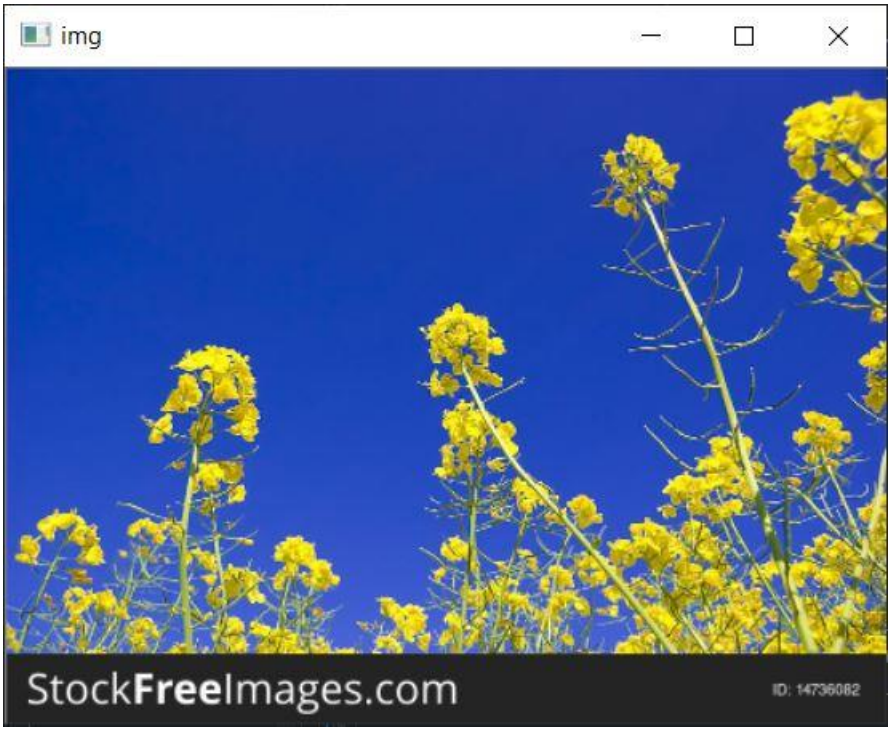
Prof. Walid Al-Atabany

1. Read the image in RGB color space:

Here is the part of code that read and resize the image and name the image by “img”

```
7
8   import cv2
9   import numpy as np
10
11
12  img = cv2.imread('10.jpg')
13
14  #original size
15  h = img.shape[0]
16  w = img.shape[1]
17
18  #new size
19  new_h = int(h/2)
20  new_w = int(w/2)
21
22  #resize image
23  resized_image = cv2.resize(img,(new_w,new_h))
24  cv2.imshow('img',resized_image)
25
```

And this is the output when we run



2. Transfer image to Lab color space
3. Separate b channel which contains blue(negative values) and yellow colors (positive values)

Here is the code that do this

```
26
27  # convert the image from rgb to lab
28  lab =  cv2.cvtColor(resized_image ,cv2.COLOR_BGR2LAB)
29  #normalize it to make it from(0:255) to (-127:127)
30  norm_image = cv2.normalize(lab, None, alpha = -127, beta = 127, norm_type = cv2.NORM_MINMAX, dtype = cv2.CV_64F)
31
32  # separate the lab image to it's three components L A B
33  L,A,B=cv2.split(norm_image)
34
35  rows = lab.shape[0]  #540
36  cols = lab.shape[1]  #960
37  threshold = 0  # This is our threshold here (above is +ve (yellow)) (below is -ve (blue))
38
```

4.Separate both colors by a new edge (white color)

Here is the code of our algorithm with comments that illustrate it

Assignment1.py

```
38
39  # Now we ar gonna loop over the image and check if there is any blue-yellow edge
40  # if there is any edge we will whiten it so we make sure there is no direct contact between
41  # blue and yellow colors
42
43  for i in range(1,rows-1):
44      for j in range(1,cols-1):
45
46          if B[i][j-1] >= threshold and B[i][j+1]<threshold: #Horizontal edge detection
47              lab[i][j] = np.array([255,127,127])   # make it white pixel
48              # Here the white is [255,127,127] not [127,0,0] because we work here in lab
49              # which is the originalimage not the normalized one
50
51          elif B[i][j-1] < threshold and B[i][j+1]>=threshold: #Horizontal edge detection
52              lab[i][j] = np.array([255,127,127])
53
54          elif B[i-1][j] >= threshold and B[i+1][j]<threshold: #Vertical edge detection
55              lab[i][j] = np.array([255,127,127])
56
57          elif B[i-1][j] >= threshold and B[i+1][j]<threshold: #Vertical edge detection
58              lab[i][j] = np.array([255,127,127])
59
60          elif B[i-1][j-1] >= threshold and B[i+1][j+1]<threshold: #Main Diagonal
61              lab[i][j] = np.array([255,127,127])
62
63          elif B[i-1][j-1] < threshold and B[i+1][j+1]>=threshold: #Main Diagonal
64              lab[i][j] = np.array([255,127,127])
65
66          elif B[i+1][j-1] >= threshold and B[i-1][j+1]<threshold: #Reverse Diagonal
67              lab[i][j] = np.array([255,127,127])
68
69          elif B[i+1][j-1] < threshold and B[i-1][j+1]>=threshold: #Reverse Diagonal
70              lab[i][j] = np.array([255,127,127])
71
72          else:
73              lab[i][j] = lab[i][j]
74
75  final_image = cv2.cvtColor(lab ,cv2.COLOR_LAB2BGR)
76  cv2.imshow('finalimg',final_image)
77  cv2.waitKey(0)
78  cv2.destroyAllWindows()
```

This is the final image

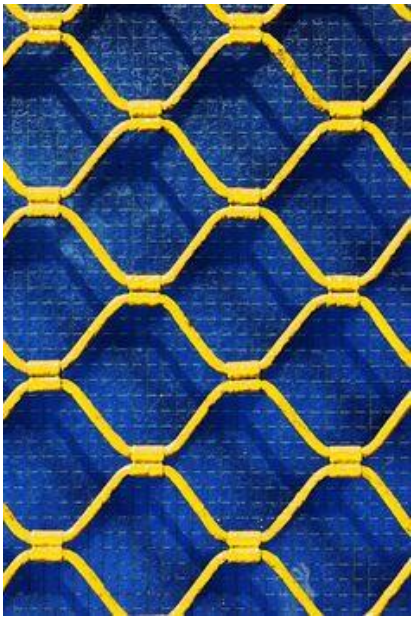
Now I’m going to show you how this algorithm works on different images to make sure that it is effective

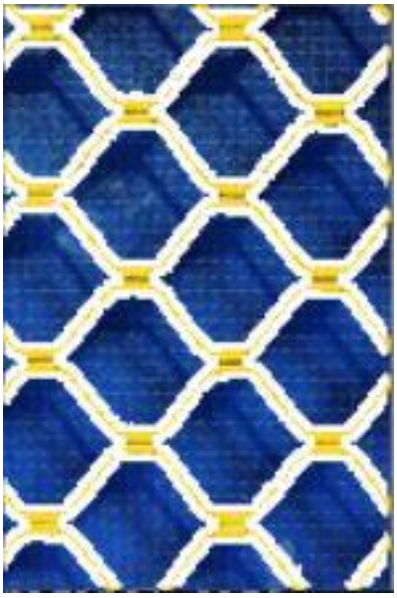
Trial no. 1:

Trial no.2:

Trial no.3:

Trial no.4:





Trial no.5: