**Moez Ahsan Azam**

**Code:**

import os

import cv2

import numpy as np

import cv2

import numpy as np

def find\_optic\_disk(image\_path, min\_radius=20, max\_radius=60):

    img = cv2.imread(image\_path)

    if img is None:

        raise ValueError(f"Could not read image at {image\_path}")

    resized\_img = cv2.resize(img, (720, 720))

    gray\_img = cv2.cvtColor(resized\_img, cv2.COLOR\_BGR2GRAY)

    blurred\_img = cv2.GaussianBlur(gray\_img, (15, 15), 0)

    \_, thresh = cv2.threshold(blurred\_img, 200, 255, cv2.THRESH\_BINARY)

    contours, \_ = cv2.findContours(thresh, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

    optic\_disc\_contour = None

    largest\_area = 0

    optic\_disc\_center = None

    for contour in contours:

        area = cv2.contourArea(contour)

        (x, y), radius = cv2.minEnclosingCircle(contour)

        if min\_radius <= radius <= max\_radius and area > largest\_area:

            largest\_area = area

            optic\_disc\_contour = contour

            optic\_disc\_center = (int(x), int(y))

    if optic\_disc\_contour is not None:

        (x, y), radius = cv2.minEnclosingCircle(optic\_disc\_contour)

        cv2.circle(resized\_img, (int(x), int(y)), int(radius), (0, 255, 0), 3)

    return resized\_img, optic\_disc\_center

def find\_optic\_disk(image\_path, roi\_size=(110, 110)):

    img = cv2.imread(image\_path)

    if img is None:

        raise ValueError(f"Could not read image at {image\_path}")

    resized\_img = cv2.resize(img, (720, 720))

    blurred\_img = cv2.GaussianBlur(resized\_img, (49, 49), 0)

    kernel = np.ones((3, 3), np.uint8)

    opened\_img = cv2.morphologyEx(blurred\_img, cv2.MORPH\_OPEN, kernel)

    gray\_img = cv2.cvtColor(opened\_img, cv2.COLOR\_BGR2GRAY)

    \_, \_, \_, max\_loc = cv2.minMaxLoc(gray\_img)

    top\_left\_x = max(0, max\_loc[0] - roi\_size[0] // 2)

    top\_left\_y = max(0, max\_loc[1] - roi\_size[1] // 2)

    bottom\_right\_x = min(resized\_img.shape[1], top\_left\_x + roi\_size[0])

    bottom\_right\_y = min(resized\_img.shape[0], top\_left\_y + roi\_size[1])

    color = (0, 255, 0)

    thickness = 2

    cv2.rectangle(resized\_img, (top\_left\_x, top\_left\_y), (bottom\_right\_x, bottom\_right\_y), color, thickness)

    return resized\_img

image\_folder = "Fundus\_image"

image\_extensions = ["tif", "jpg", "jpeg"]

for i in range(1, 20):

    image\_file = None

    for ext in image\_extensions:

        temp\_image\_file = f"{i}.{ext}"

        image\_path = os.path.join(image\_folder, temp\_image\_file)

        if os.path.exists(image\_path):

            image\_file = temp\_image\_file

            break

    if not image\_file:

        print(f"No valid image found for index {i}")

        continue

    image\_path = os.path.join(image\_folder, image\_file)

    img = cv2.imread(image\_path)

    if img is None:

        print(f"Failed to read image: {image\_file}")

        continue

    try:

        roi = mark\_brightest\_region(image\_path)

        img1,cord = find\_optic\_disk (image\_path)

        print(cord)

        cv2.imshow("ROI Image",roi)

        cv2.imshow("Image",img1)

        cv2.waitKey(0)

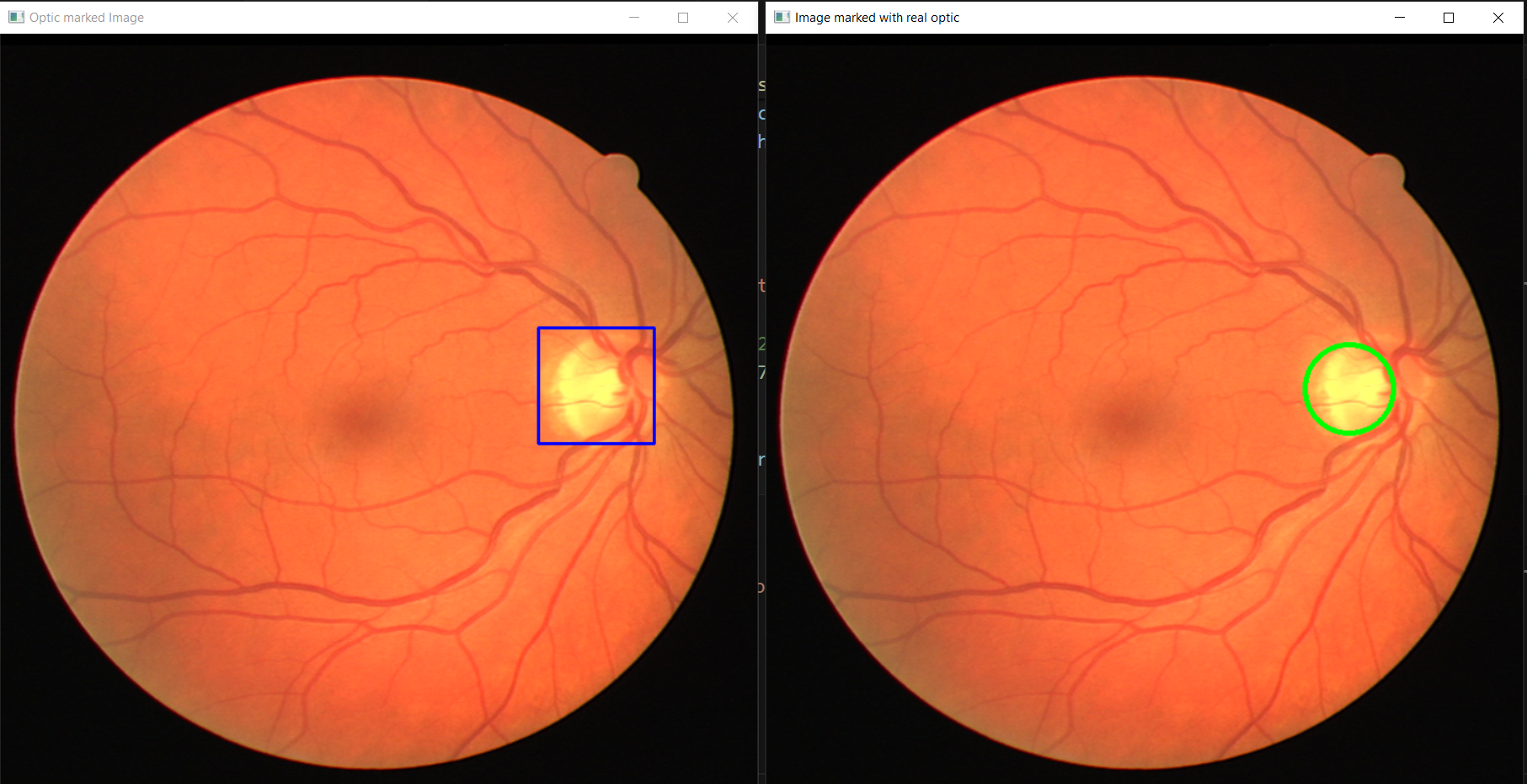
        print(f"Successfully processed image: {image\_file}")

    except Exception as e:

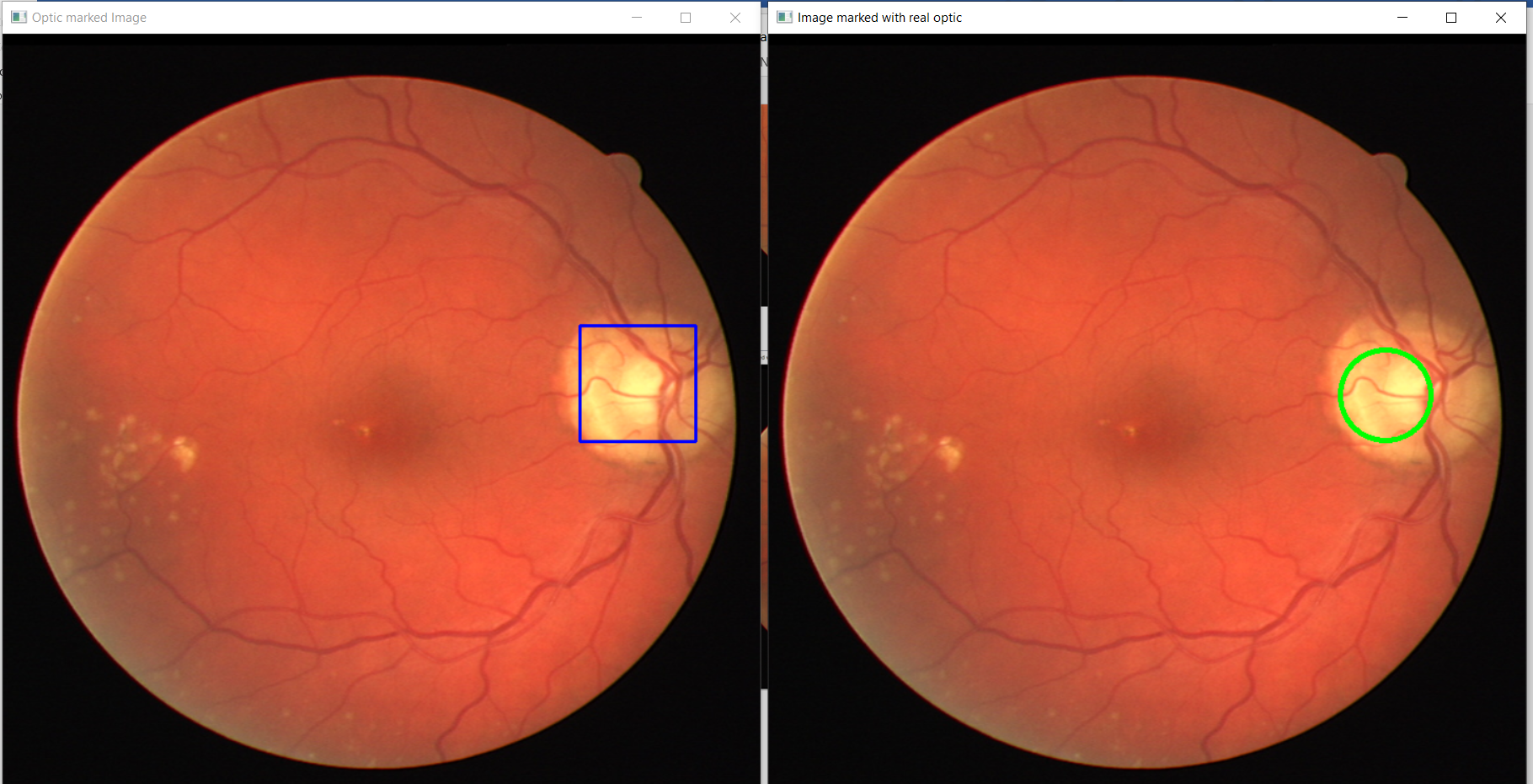
        print(f"Error processing {image\_file}: {e}")

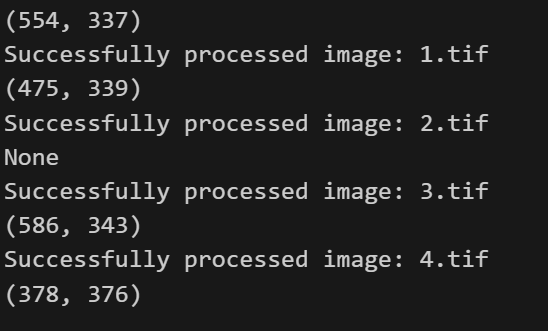
**Output:**

**Examples of some output:**

****

****

****

****

**Pseudo Code:**

* Open and read the image from the given location. If it can't be read, show an error message.
* Adjust the image to a common size to keep things consistent.
* Change the image to black and white.
* Apply a blur effect to smooth out any noise or small details.
* Use a technique to turn the image into just black and white, where white represents the brightest areas.
* I have used opening and guassian filter to remove noise.
* Look for the outlines of the bright shapes in the image.
* Among the shapes you found, locate the one that could be the optic disc based on its size and shape.
* If we find it, get its center point and radius to mark it.
* Draw a circle where the real optic disc is located.
* Try to process each image to identify the optic disc and the brightest region.
* Display the images with the marked optic disc and brightest region.