CN240 Data Science for Signal Processing

Code Explanation: Machine Learning

1. Chonlasit Mooncorn 6110613020
2. Prach Chantasantitam 6110613202
3. Weeraphat Leelawittayanon 6210612823
4. Raned Chuphueak 6210612864

ไฟล์ preprocessing.ipynb

Function: resizeImage

ทำการ resize รูปให้เป็นขนาด 200 * 200

```
# resize image
def resizeImage(img) :

    dim = (200, 200)
    resized = cv2.resize(img, dim, interpolation=cv2.INTER_AREA)

    return resized
```

Function: removeBlackbar

เป็นการลบขอบดำของภาพออก ทำให้ลด error ในการหาค่า threshold ของรูปภาพหนึ่ง ๆ

```
#clear blackbar get only retina retina
def removeBlackbar(img):

    b,g,r = cv2.split(img)

    low = np.array([20])
    high = np.array([255])
    mask = cv2.inRange(g, low, high)

    a = img.shape[0]-1
    middle_y = img.shape[1]/2
    middle_y = int(middle_y)

    first_x = 0
    last_x = 0

    for x in range(img.shape[1]-1) :
        if(mask[middle_y][x] == 255):
            first_x = x
            break
    while a >= 0 :
        if(mask[middle_y][a] == 255):
            last_x = a
            break
        a = a - 1

    img_cropped = img[(0):(199), (first_x):(last_x)]

    return img_cropped
```

ตัวอย่าง

ก่อนทำ

หลังทำ

Function: removeAllArtifact

ลบ artifact ส่วนใหญ่ที่ยังอยู่ขอบของ retina ออก

```python
#clear artifact around retina
def removeAllArtifact(img):

    b,g,r = cv2.split(img)
    h,w = g.shape

    cy = (h/2)-0.5
    cx = (w/2)-0.5

    cy = int(cy)
    cx = int(cx)

    black = np.zeros((h,w), np.uint8)

    center_coordinates = (cx, cy)

    axesLength = (175, 200)

    angle = 0

    startAngle = 0

    endAngle = 360

    # Red color in BGR
    color = (255, 255, 255)

    # Line thickness of 5 px
    thickness = -1

    # Using cv2.ellipse() method
    # Draw a ellipse with red line borders of thickness of 5 px
    image = cv2.ellipse(black, center_coordinates, axesLength,
           angle, startAngle, endAngle, color, thickness)

    res = cv2.bitwise_and(img, img, mask=image)

    return res
```

ตัวอย่าง

ก่อนทำ

หลังทำ

Function: checkRetinaColor

จะช่วยในการตรวจสอบว่ารูปนี้มี retina สีอะไร เพื่อช่วยในการใช้หา optic ใน function ต่อไป

```
#check retina color
def checkRetinaColor(img):

    image = cv2.cvtColor(img, cv2.COLOR_RGB2BGR)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

    original = image.copy()
    lower = np.array([20, 200, 200], dtype="uint8")
    upper = np.array([100, 255, 255], dtype="uint8")
    mask = cv2.inRange(image, lower, upper)

    cnts = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    cnts = cnts[0] if len(cnts) == 2 else cnts[1]

    for c in cnts:
        x,y,w,h = cv2.boundingRect(c)
        cv2.rectangle(original, (x, y), (x + w, y + h), (36,255,12), 2)

    mean = findMean(mask)
    if(mean > 15):

        return "yellow case"
    else :

        return "normal case"
```

Function: getOptic

จะทำการ crop รูปส่วน optic เพื่อจำกัดขอบเขตในการหา CDR โดยการ crop จะขึ้นอยู่กับสีของ retina

```
# get Optic only
def getOptic(img,case):

    if(case == "yellow case"):


        b,_,_ = cv2.split(img)

        max_b = findmaxthreshold(b)
        mean_b = findMean(b)
        sd_b = findSD(b)

        sum_mm_b = (max_b + sd_b )/2


        edge_b = make_threshold(b,sum_mm_b)


        optic , vessel = findOptic(edge_b,img)

        return optic , vessel , case

    elif(case == "normal case"):
        b,g,r  = cv2.split(img)

        max_g = findmaxthreshold(g)
        mean_g = findMean(g)
        sd_g = findSD(g)

        sum_mm_g = (mean_g + max_g + sd_g)/2

        edge_g = make_threshold(g,sum_mm_g)

        optic , vessel = findOptic(edge_g,img)


        return optic , vessel ,case
```

<u>อธิบายเพิ่มเติม</u>

Optic ของ retina ส่วนใหญ่จะมีลักษณะสีขาวสว่าง และรูปส่วนใหญ่ ตัว retina จะเป็นสีแดง เราจะดึง channel สีเขียวออกมาแล้วหาค่า threshold เพื่อทำการ crop ต่อไปในอนาคต

แต่ในกรณีที่ retina เป็นสีเหลือง ค่าสีของ retina จะมีสีแดงและสีเขียวเยอะ หากเราใช้ channel สีเขียวจะเป็นการดึงทั้งรูปออกมา เราจึงจำเป็นต้องไปใช้สีน้ำเงินแทน

Function: findOptic

จะทำการรับรูปที่ผ่านการหาค่า threshold มาแล้ว จากนั้นจะทำการหาวงรีที่ใหญ่ที่สุดแล้วทำการ crop บริเวณรอบ ๆ optic มาเพื่อตัดส่วนที่ไม่จำเป็นออกไป แต่หากไม่สามารถหาวงกลมได้ จะมี function ชื่อ basic_optic รองรับ เพื่อการหา optic ในอีกรูปแบบหนึ่ง

1

```
def findOptic(edge,img) :


    h, w = edge.shape[:2]
    mask = np.zeros((h, w), np.uint8)

    contours, hierarchy = cv2.findContours(edge.copy(), cv2.RETR_EXTERNAL,
    cv2.CHAIN_APPROX_SIMPLE)

    try:
        cnt = max(contours, key=cv2.contourArea)
        extLeft = tuple(cnt[cnt[:, :, 0].argmin()][0])
        extRight = tuple(cnt[cnt[:, :, 0].argmax()][0])
        radius = (extRight[0] - extLeft[0])/2
        r = radius.astype(np.int32)

        M = cv2.moments(cnt)
        cx = int(M['m10']/M['m00'])
        cy = int(M['m01']/M['m00'])
    except (ValueError, TypeError, ZeroDivisionError):
        #print("can't find cx cy")
        img_cropped_optic_basic,vessel_crop = basic_optic(img)

        return img_cropped_optic_basic , vessel_crop


    #rectangle
    start_x = cx - 60
    end_x = cx + 60

    start_y = cy - 60
    end_y = cy + 60
```

2

```
#rectangle
start_x = cx - 60
end_x = cx + 60

start_y = cy - 60
end_y = cy + 60


start_point = (start_x, start_y)

# Ending coordinate, here (125, 80)
# represents the bottom right corner of rectangle
end_point = (end_x, end_y)

# Black color in BGR
color = (255)

# Line thickness of -1 px
# Thickness of -1 will fill the entire shape
thickness = -1


# Create a black image
black = np.zeros((h,w), np.uint8)


# Using cv2.circle() method
# Draw a circle of red color of thickness -1 px
image = cv2.rectangle(black, start_point, end_point, color, thickness)

vessel = extract_bv_circle(img,image)


res = cv2.bitwise_and(img, img, mask=image)
rgb =cv2.cvtColor(res, cv2.COLOR_BGR2RGB)

img_cropped_optic = rgb[(start_y):(end_y), (start_x):(end_x)]
img_cropped_vessel = vessel[(start_y):(end_y), (start_x):(end_x)]
#print(img_cropped.shape)
if(img_cropped_optic.shape[0] == 0 or img_cropped_optic.shape[1] == 0):
    #print("Catch Error height weight")
    img_cropped_optic_basic , vessel_crop = basic_optic(img)

    return img_cropped_optic_basic , vessel_crop

if(img_cropped_optic.shape[0] < 50 or img_cropped_optic.shape[1] < 50):
    #print("Catch Error height weight")
    img_cropped_optic_basic , vessel_crop = basic_optic(img)

    return img_cropped_optic_basic , vessel_crop



return img_cropped_optic , img_cropped_vessel
```

Function: basic_optic

เป็นการหาจุดที่สว่างที่สุดของรูปที่thresholdมาแล้วโดยจะไล่เช็คที่ละpixelจากบนไปล่างและล่าง-บนแล้วทำการcropมา

1

```python
# find optic based on highest color range
def basic_optic(img):

    r,g,b = cv2.split(img)

    h,w = g.shape

    blurred = cv2.GaussianBlur(g, (5, 5), 0)

    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))

    img_clahe = clahe.apply(blurred)

    max_th = (findmaxthreshold(img_clahe) + findMean(img_clahe) )/2


    _, th = cv2.threshold(img_clahe,max_th,255,cv2.THRESH_BINARY)
    R1 = cv2.morphologyEx(th, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(2,2)), iterations = 1)
    r1 = cv2.morphologyEx(R1, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(7,7)), iterations = 1)


    first_x = 0
    first_y = 0
    y,x = r1.shape #y = height x = width

    count = 0

    # first x , y
    for i in range(y) :
        for j in range(x) :
            if(r1[i][j] == 255):

                first_x = j
                first_y = i
                count = 1
            if(count == 1):
                break
        if(count == 1):
            break
```

2

```python
# last x , y
pos_y = y -1
last_x = 0
last_y = 0

while pos_y >= 0 :
    pos_x = x - 1
    while pos_x >= 0 :
        if(r1[pos_y][pos_x] == 255):
            last_y = pos_y
            last_x = pos_x
            count = 2
        if(count == 2):
            break

        pos_x = pos_x - 1

    if(count == 2):
        break
    pos_y = pos_y - 1

#rectangle
start_x = first_x - 30
end_x = last_x + 30
start_y = first_y - 30
end_y = last_y + 30

#rectangle fix
if (start_x < 0):
    end_x = end_x - (start_x)
    start_x = 0
elif (end_x < 0):
    start_x = start_x - (end_x)
    end_x = 0
if (end_y < 0):
    start_y = start_y - end_y
    end_y = 0
elif (start_y < 0):
    end_y = end_y - start_y
    start_y = 0

#fix postion if invert
if(start_x > end_x):
    temp = start_x
    start_x = end_x
    end_x = temp
#print(start_x)
#print(end_x)

start_x, end_x = fixPosition120(start_x, end_x)
start_y, end_y = fixPosition120(start_y, end_y)
```

3

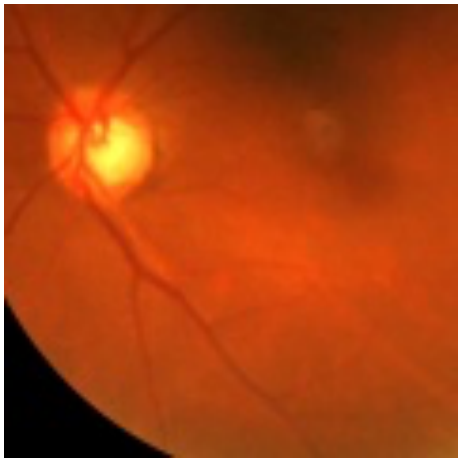
```
    start_point = (start_x, start_y)

    # Ending coordinate, here (125, 80)
    # represents the bottom right corner of rectangle
    end_point = (end_x, end_y)

    # Black color in BGR
    color = (255)

    # Line thickness of -1 px
    # Thickness of -1 will fill the entire shape
    thickness = -1


    # Create a black image
    black = np.zeros((h,w), np.uint8)


    # Using cv2.circle() method
    # Draw a circle of red color of thickness -1 px
    image = cv2.rectangle(black, start_point, end_point, color, thickness)

    res = cv2.bitwise_and(img, img, mask=image)
    rgb =cv2.cvtColor(res, cv2.COLOR_BGR2RGB)



    img_cropped = rgb[(start_y):(end_y), (start_x):(end_x)]
    vessel = extract_bv_basic(img,start_x,end_x,start_y,end_y)


    return img_cropped , vessel

# fix shape image when use basic optic
def fixPosition120(begin, last):
    if (last > begin):
        while (last - begin > 120):
            last = last - 1
        while (last - begin < 120):
            last = last + 1
    else:
        while (begin - last > 120):
            begin = begin - 1
        while (begin - last < 120):
            begin = begin + 1
    return begin, last
```

<u>อธิบายเพิ่มเติม</u>

ขณะเราทำการ crop เอาเฉพาะส่วน optic ไปนั้น เราได้ทำการ crop ส่วนของเส้นเลือดไปด้วย โดย crop ตามส่วนที่ได้จาก optic ด้วยที่ function extract_bv

ตัวอย่าง

ก่อนทำ



หลังทำ

Function: focusOptic

เอาแค่เฉพาะส่วน optic ให้ได้มากที่สุดเพื่อลดค่า threshold ที่อาจจะทำให้หาค่า CDR เกิดความคาดเคลื่อน

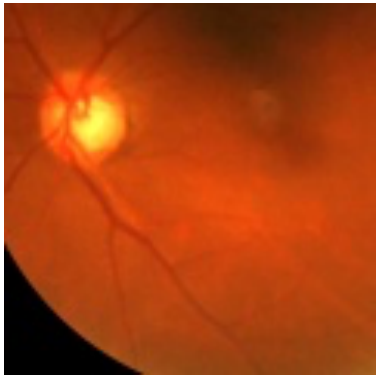
```python
def focusOptic(img,vessel,max_thres) :

    r,g,b = cv2.split(img)
    h, w = img.shape[:2]

    thres = max_thres-20
    blurred = cv2.GaussianBlur(g, (5, 5), 0)
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))

    _, thresh = cv2.threshold(blurred ,thres,255,cv2.THRESH_BINARY)
    kernel = np.ones((10,10),np.uint8)
    opening = cv2.morphologyEx(thresh,cv2.MORPH_CLOSE,kernel, iterations = 2)

    edges = cv2.Canny(opening,250,255)

    edges = np.uint8(edges)
    contours = cv2.findContours(edges, cv2.RETR_EXTERNAL,
    cv2.CHAIN_APPROX_SIMPLE)[0]
    try:
        cnt = max(contours, key=cv2.contourArea)
        extLeft = tuple(cnt[cnt[:, :, 0].argmin()][0])
        extRight = tuple(cnt[cnt[:, :, 0].argmax()][0])
        radius = (extRight[0] - extLeft[0])/2
        r = radius.astype(np.int32)
        blood = r

        M = cv2.moments(cnt)
        cx = int(M['m10']/M['m00'])
        cy = int(M['m01']/M['m00'])
    except (ValueError, TypeError, ZeroDivisionError):

        print("error v2")

        black = np.zeros((h,w), np.uint8)

        return img , black
```

1

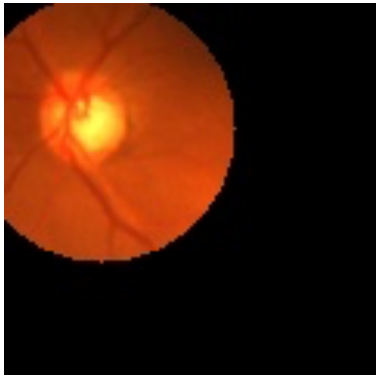
```python
#create circle for mark optic
center_coordinates = (cx, cy)

radius = r+40
radius_blood = r + 20

if(radius > 60):
    radius = 60
elif(radius <= 40):
    radius = 60

color = (255)
thickness = -1
black = np.zeros((h,w), np.uint8)
black_1 = np.zeros((h,w), np.uint8)

image = cv2.circle(black, center_coordinates, radius, color, thickness)
image_blood = cv2.circle(black_1, center_coordinates, radius_blood, color, thickness)
blood = extract_bv_circle(img,image_blood)

res = cv2.bitwise_and(img, img, mask=image)

return res , blood
```

2

ตัวอย่าง



ก่อนทำ



หลังทำ

Function: CDR

เป็นการเอารูปทั้ง optic และ focus optic และเส้นเลือดมาทำการหาค่า CDR ของรูป

```
def CDR(optic, foptic , vessel , name) :
```

1

```
    # split color channel optic foptic
    r_optic,g_optic,b_optic = cv2.split(optic)
    r_foptic,g_foptic,b_foptic = cv2.split(foptic)

    blurred = cv2.GaussianBlur(r_optic,(5, 5), 0)


    r1 = cv2.morphologyEx(blurred, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5)), iterations = 1)
    R1 = cv2.morphologyEx(r1, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5)), iterations = 1)

    #หาthreshold ของ disc
    max_r = findmaxthreshold(R1)

    mean_r = findMean(r_optic)

    sd_r = findSD(r_optic)

    #ทำการหาthresholdสำหรับไว้ใช้หาdisc
    sum_mm_r = ( max_r + mean_r + 0.5*sd_r )/2

    #หากค่าthreshold ของรูปสูงมากกว่า255 จะใช้thresholdของfocus opticแทนเพื่อลดค่าthreshold
    if(sum_mm_r > 255):
        #renai threshold
        print("*****************************************")
        print("foptic threshold")
        mean_r = findMean(r_foptic)
        mean_r = findMean(r_foptic)
        sd_r = findSD(r_foptic)
        sum_mm_r = ( max_r + mean_r + 0.5*sd_r )/2

    #threshold รูป
    img_r = pixelminus(r_optic,sum_mm_r)

    #ทำการรวมdiscกับเส้นเลือดเข้าด้วยกันเพื่อให้เกิดความสมบูรณ์ของdisc
    op_v = mergeVesselWithOptic(img_r,vessel)

    #หาthreshold ของ cup
    max_g = findmaxthreshold(g_foptic)
    mean_g = findMean(g_foptic)
    sd_g = findSD(g_foptic)

    #ทำการหาthresholdสำหรับไว้ใช้หาcup
    sum_mm_g = (mean_g + max_g + sd_g)/2

    #threshold รูป
    img_g = pixelminus(g_foptic,sum_mm_g)
```

2

```
    #นำไปเข้าfunction เพื่อหาค่าdiameter radius แลนด่อ
    disc_diameter,disc_area,radius_disc,pixel_disc = disc(optic,op_v,name)

    cup_diameter,cup_area,radius_cup,pixel_cup,draw  = cup(optic,img_g,name)

    #ทำการคำนวณหาค่าfeatureต่างๆ
    if(cup_diameter == -1 or disc_diameter == -1 ):
        pixel_ratio =  0
        diameter = 0
        area = 0
        radius = 0
    else :
        pixel_ratio =  pixelCal(pixel_disc,pixel_cup)
        diameter = diameterCal(disc_diameter,cup_diameter)
        area = areaCal(disc_area,cup_area)
        radius = radiusCal(radius_disc,radius_cup)



    return diameter,area,radius,pixel_ratio,draw
```

Function: disc

ทำการหาค่า diameter radius area และ pixel ของ disc และทำการวาดวงรีที่แสดงส่วนของ disc ใน optic

```
def disc(optic,img_thres,name) :

    kernel = np.ones((9,9),np.uint8)

    # make disc look more complete
    r1 = cv2.morphologyEx(img_thres, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5)), iterations = 1)
    R1 = cv2.morphologyEx(r1, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5)), iterations = 1)
    r2 = cv2.morphologyEx(R1, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(11,11)), iterations = 1)
    R2 = cv2.morphologyEx(R1, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(11,11)), iterations = 1)
    dilation = cv2.dilate(R2,kernel,iterations = 1)

    edges = cv2.Canny(R2,200,255)

    # find the largest ellipse like shape based on contours to draw a ellipse
    contours,hierarchy = cv2.findContours(edges, cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE) #Getting all possible contours in the segmented image
    try:
        cup_diameter = 0
        largest_area = 0
        el_cup = contours[0]
        if len(contours) != 0:
            for i in range(len(contours)):
                if len(contours[i]) >= 5:
                    area = cv2.contourArea(contours[i]) #Getting the contour with the largest area
                    if (area>largest_area):
                        largest_area=area
                        index = i
                        el_cup = cv2.fitEllipse(contours[i])

        #pixel count
        pixel_disc = count_pixel(dilation)

        #find radius
        cnt = contours[index]
        extLeft = tuple(cnt[cnt[:, :, 0].argmin()][0])
        extRight = tuple(cnt[cnt[:, :, 0].argmax()][0])
        radius_disc = (extRight[0] - extLeft[0])/2

        cv2.ellipse(optic,el_cup,(0,0,255),1)
        x,y,w,h = cv2.boundingRect(contours[index]) #fitting a rectangle on the ellipse to get the length of major axis

        disk_diameter = max(w,h) #major axis is the diameter
        disk_area = disk_diameter**2 * math.pi
        #print("disc_dia",disk_diameter)
        #print("disc_area",disk_area)
        #print("disc_rad", radius_disc)
        return disk_diameter,disk_area,radius_disc,pixel_disc

    except:
        #print("can't draw Disc Ellipse")

        return -1,-1,-1,-1
```

Function: cup

ทำการหาค่า diameter radius area และ pixel ของ cup และทำการวาดวงรีที่แสดงส่วนของ cup ใน optic

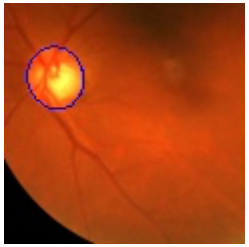
```
def cup(optic,img_threshold,name) :

    kernel =  np.ones((3,3),np.uint8)
    kernel_1 = np.ones((5,5),np.uint8)
    kernel_2 = np.ones((8,8),np.uint8)
    kernel_3 = np.ones((10,10),np.uint8)

    # make cup look more complete
    opening = cv2.morphologyEx(img_threshold, cv2.MORPH_OPEN, kernel)
    dilation1 = cv2.dilate(opening ,kernel_1,iterations = 1)
    dilation2 = cv2.dilate(dilation1,kernel_2,iterations = 1)
    erosion = cv2.erode(dilation2,kernel_3,iterations = 1)

    #save
    cv2.imwrite("opening_cup.jpg", opening)
    cv2.imwrite("dilation_cup.jpg", dilation1)
    cv2.imwrite("erosion_cup.jpg", erosion)

    canny = cv2.Canny(erosion,0,255)

    # find the largest ellipse like shape based on contours to draw a ellipse
    contours,hierarchy = cv2.findContours(canny, cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE) #Getting all possible contours in the segmented image

    try :
        cup_diameter = 0
        largest_area = 0
        el_cup = contours[0]
        if len(contours) != 0:
            for i in range(len(contours)):
                if len(contours[i]) >= 5:
                    area = cv2.contourArea(contours[i]) #Getting the contour with the largest area
                    if (area>largest_area):
                        largest_area=area
                        index = i
                        el_cup = cv2.fitEllipse(contours[i])


        # find CDR

        # pixel count
        pixel_cup = count_pixel(erosion)

        # find radius
        cnt = contours[index]
        extLeft = tuple(cnt[cnt[:, :, 0].argmin()][0])
        extRight = tuple(cnt[cnt[:, :, 0].argmax()][0])
        radius_cup = (extRight[0] - extLeft[0])/2

        cv2.ellipse(optic,el_cup,(255,0,0),1)
```

1

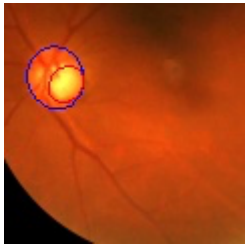
2

```
        x,y,w,h = cv2.boundingRect(contours[index])

        cup_diameter = max(w,h) #major axis is the diameter
        cup_area = cup_diameter**2 * math.pi

    except:
        print("can't draw Cup Ellipse")

        return -1,-1,-1,-1,optic

    return cup_diameter,cup_area,radius_cup,pixel_cup,optic
```

ตัวอย่าง



Disc



Cup

ค่าที่ได้จาก function CDR

```
****************************************
CDR
****************************************
area 0.33149678604224053
diameter 0.5757575757575758
radius 0.6071428571428571
pixel_ratio 0.11523046092184369
```

Function: cataractClassification

ทำการเพิ่มความสว่างของรูปเพื่อหารายละเอียดหากมีเยอะแสดงว่าไม่เป็น cataract

```python
# other diseases features
def cataractClassification(img,name):

    gray = cv2.cvtColor(img , cv2.COLOR_BGR2GRAY)
    rgb = cv2.cvtColor(img , cv2.COLOR_BGR2RGB)
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
    cl1 = clahe.apply(gray)
    cl2 = clahe.apply(cl1)
    #cl3 = clahe.apply(cl2)
    edges = cv2.Canny(cl2,200,400)

    '''
    fig, axs = plt.subplots(nrows=2, ncols = 2, sharex=True, figsize=(10, 10))
    axs[0][0].set_title('ori')
    axs[0][0].imshow(rgb)

    axs[0][1].set_title('cataract')
    axs[0][1].imshow(edges)

    plt.suptitle("cataract filename " + name, fontsize=16)

    plt.show()
    '''

    count = count_pixel(edges)

    return count , edges
```

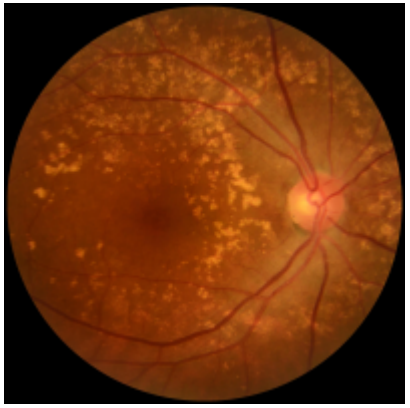
Function: count_Microaneurysm

ทำการดึงจุดสะเก็ดต่าง ๆ ออกมาและลบส่วนอื่น ๆ ออกไป

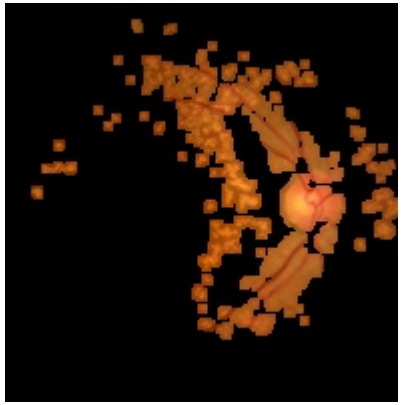
```
def count_Microaneurysm(img,name):

    rgb = cv2.cvtColor(img,cv2.COLOR_RGB2BGR)
    ori = rgb
    r,g,b = cv2.split(img)

    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
    g = clahe.apply(g)

    # Morphological Transformations
    kernel_1 = np.ones((3,3),np.uint8)
    kernel_2 = np.ones((5,5),np.uint8)
    d1 = cv2.dilate(g,kernel_1,iterations = 1)
    e1 = cv2.erode(d1,kernel_1,iterations = 1)
    d2 = cv2.dilate(e1,kernel_2,iterations = 1)

    max_g = findmaxthreshold(d2)
    mean_g = findMean(d2)
    sd_g = findSD(d2)

    sum_mm_g = (max_g + sd_g )/2

    _,thres = cv2.threshold(d2,sum_mm_g,255,cv2.THRESH_BINARY)

    pixel = count_pixel(thres)

    res = cv2.bitwise_and(rgb, rgb, mask=thres)

    return pixel , res
```

ตัวอย่าง

Microaneurysm

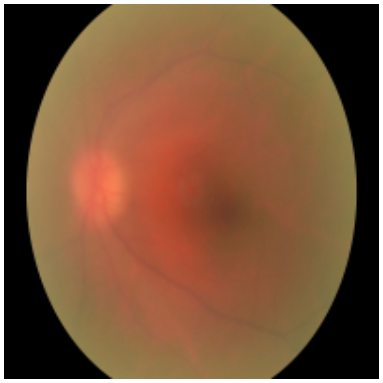


ก่อน

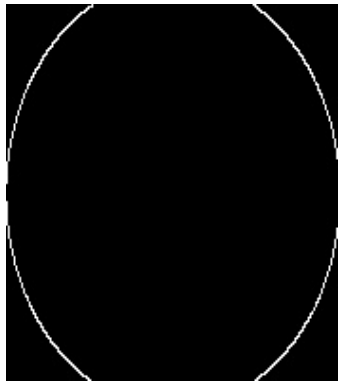


หลัง

Cataract



ก่อน



หลัง