MODULE 10
THRESHOLDING

Arranged By:

Rajendra Rakha Arya Prabaswara

(1941720080/20)

PROGRAM STUDI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

1. **Importing Library & Save To Github**

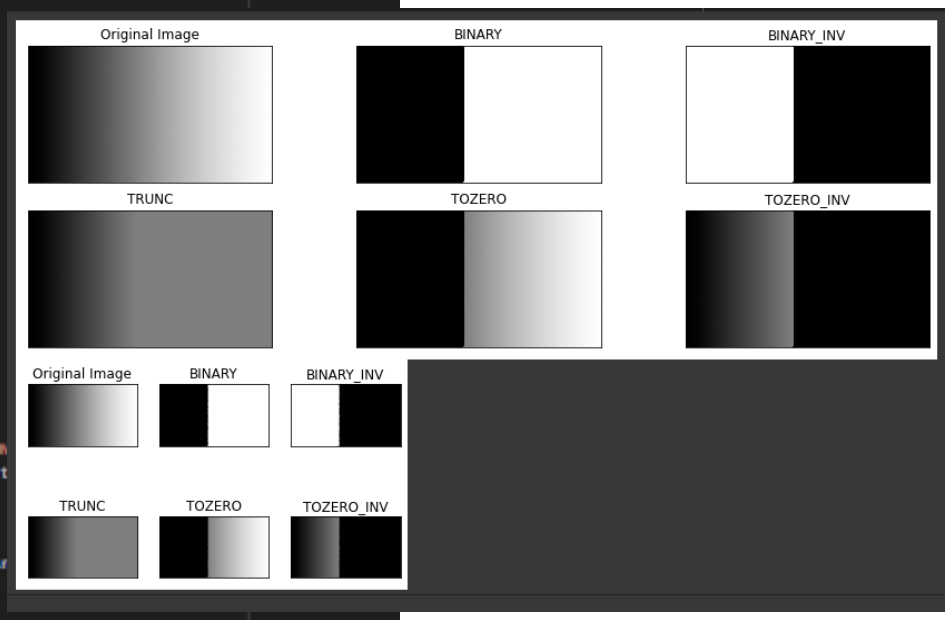
```
[ ] from google.colab import drive

    drive.mount('/content/drive')

    Mounted at /content/drive
```

```
import cv2
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt
```

2. Manually create a Global Threshold (BINARY, BINARY_INV, TRUNC, TOZERO,TOZERO_INV) according to the description from the chart shown above.

```
[ ] # No 3
    # tanpa menggunakan library
    filename_1 = ('/content/drive/MyDrive/PCVK_MM/Images/gradient.jpg')
    img_g = cv2.imread(filename_1, 0)
    thresh1 = cv2.imread(filename_1)
    thresh2 = cv2.imread(filename_1)
    thresh3 = cv2.imread(filename_1)
    thresh4 = cv2.imread(filename_1)
    thresh5 = cv2.imread(filename_1)
    thresh = 127    #nilai Threshold yang ditentukan

    #1. Original Image
    thresh1[img_g>thresh] = 255
    thresh1[img_g<=thresh] = 0

    #2. Binary threshold inverse
    thresh2 = 255 - thresh1
    #3. Threshold Truncate
    thresh3[img_g>thresh] = thresh
    #4. Threshold Tozero
    thresh4[img_g<=thresh] = 0
    #5. Threshold Tozero Inverse
    thresh5[img_g>thresh] = 0

    titles = ['Original Image','BINARY','BINARY_INV','TRUN
    images = [img_g, thresh1, thresh2, thresh3, thresh4, t

    plt.figure(figsize = (15,5))
    for i in range(len(images)):
      plt.subplot(2,3,i+1),plt.imshow(images[i],'gray', ir
      plt.title(titles[i])
      plt.xticks([]),plt.yticks([])

    plt.show()

    # menggunakan library
    img_g = cv2.imread(filename_1)
    ret,thresh1 = cv2.threshold(img_g,127,255,cv2.THRESH_BINARY)
    ret,thresh2 = cv2.threshold(img_g,127,255,cv2.THRESH_BINARY_INV)
    ret,thresh3 = cv2.threshold(img_g,127,255,cv2.THRESH_TRUNC)
    ret,thresh4 = cv2.threshold(img_g,127,255,cv2.THRESH_TOZERO)
    ret,thresh5 = cv2.threshold(img_g,127,255,cv2.THRESH_TOZERO_INV)

    titles = ['Original Image','BINARY','BINARY_INV','TRUNC','TOZERO','TOZERO_INV']
    images = [img_g, thresh1, thresh2, thresh3, thresh4, thresh5]

    for i in range(6):
        plt.subplot(2,3,i+1),plt.imshow(images[i],'gray',vmin=0,vmax=255)
        plt.title(titles[i])
        plt.xticks([]),plt.yticks([])

    plt.show()
```

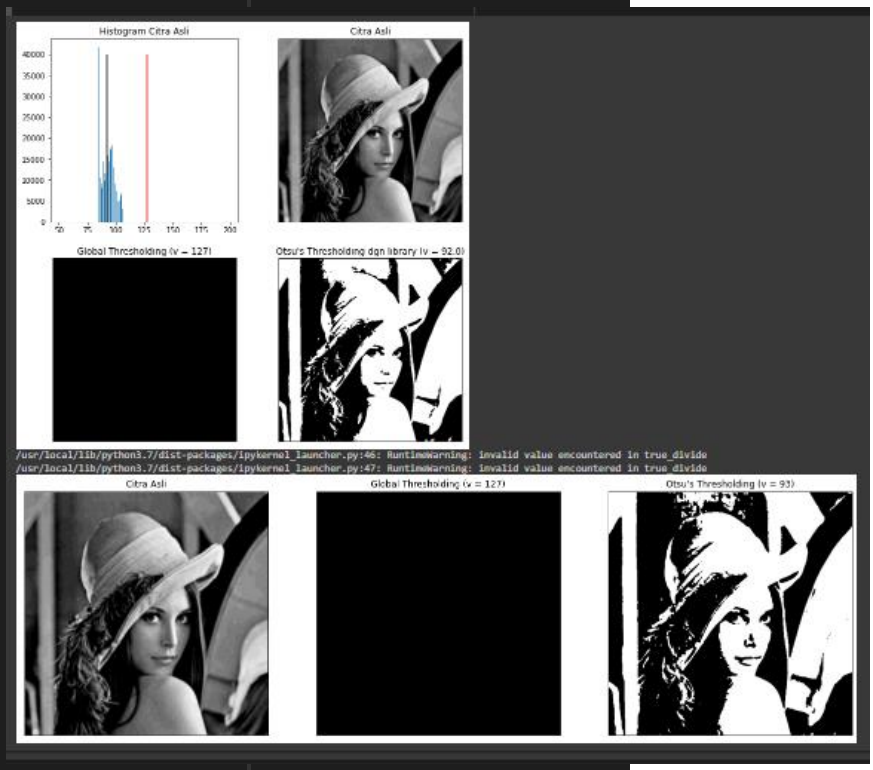
RESULT



3. Create Otsu Thresholding without using the Library. Also display the threshold value when you use Otsu's, as shown in the following image . (use the image lena_gs_lc2.jpg so that it really looks different between the otsu's results and the usual global threshold)

```
# No 4
# menggunakan library
filename_3 = ('/content/drive/MyDrive/PCVK_PM/images/lena_gs_lc2.jpg')
img_lgs = cv2.imread(filename_3, 0)
blur = cv2.GaussianBlur(img_lgs,(5,5),0)
thresh = 127

ret,thi = cv2.threshold(blur,thresh,255,cv2.THRESH_BINARY)
ret2,thi2 = cv2.threshold(blur,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

x = ("Otsu's thresholding dgn library (v = "+str(ret2)+")"
titles = ['Citra Asli', 'Global Thresholding (v = 127)', x]
citrai = [blur, thi, thi2]
plt.figure(figsize = (30,10))
plt.subplot(2,3,1),plt.hist(blur.ravel(),256,[50,200])

plt.vlines(ret,0,40000,colors='red') #garis vertical merah menunjukan threshold global 127
plt.vlines(ret2,0,40000,colors='black') #garis vertical hitam menunjukkan threshold hasil otsu's
plt.title('Histogram Citra Asli')

for i in range(len(citrai)):
  plt.subplot(2,3,i+2),plt.imshow(citrai[i],'gray')
  plt.title(titles[i])
  plt.xticks([]),plt.yticks([])

plt.show()

# tanpa menggunakan library
img_lgs = cv2.imread(filename_3, 0)
blur = cv2.GaussianBlur(img_lgs, (5,5),0)
hist = cv2.calcHist([blur],[0],None,[256], [0,256])

hist_norm = hist.ravel()/hist.max()
Q = hist_norm.cumsum()
bins = np.arange (256)
ab = 0
amb = 0
varmax = 0
fn_min = np.inf
thresh = -1

for i in range(0,255):
  ab += hist[i]
  af = (img_lgs.shape[0]*img_lgs.shape[1]) - ab
  amb += (i*hist[i])
  ab = amb / ab
  af = (amb - amb)/af
  betweenvar = ab * af * (ab - af) * (ab - af)

  if (betweenvar>varmax):
    varmax = betweenvar
    thresh = i

x = ("Otsu's thresholding (v = "+str(thresh)+")"
ret, thi = cv2.threshold(blur, 127, 255, cv2.THRESH_BINARY)
ret, thi = cv2.threshold (blur, thresh, 255, cv2. THRESH_BINARY)

titles = ['Citra Asli', 'Global Thresholding (v = 127)', x]
citrai = [blur, thi, thi]
plt.figure(figsize = (30,15))

for i in range(len(citrai)):
    plt. subplot(2,3,i+1),plt.imshow(citrai[i], 'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])

plt.show()
```

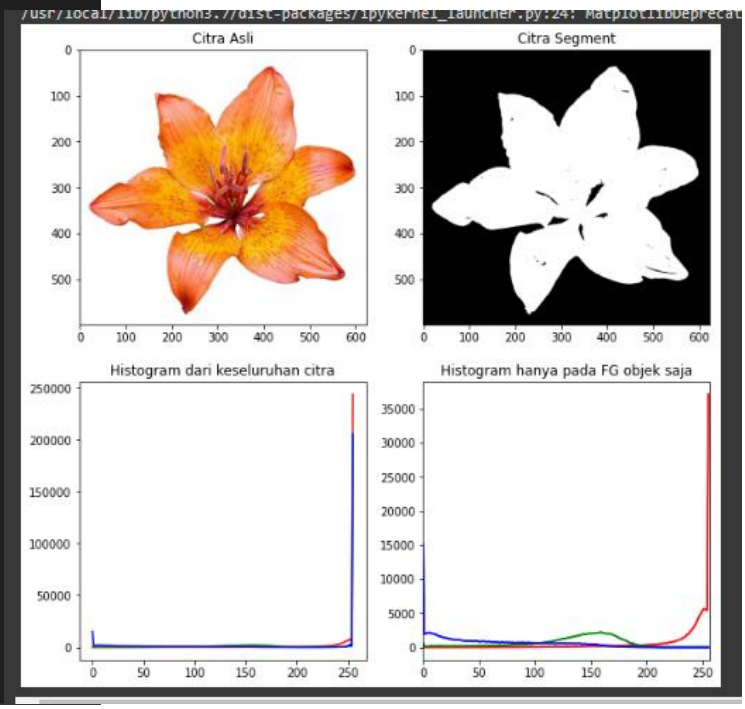
OUTPUT



4. Create a histogram from a segmented image, the histogram is only in the foreground image. Use the Lily or Leaves image provided in the images folder . Instructions:

```
# No 5
filename_3 = ('/content/drive/MyDrive/PCVK_MM/Images/lily.jpg')
img_1 = cv2.imread(filename_3)
img_rgb = cv2.cvtColor(img_1,cv2.COLOR_BGR2RGB)
img_gray = cv2.cvtColor(img_rgb,cv2.COLOR_BGR2GRAY)

ret3,mask = cv2.threshold(img_gray,200,255,cv2.THRESH_BINARY_INV)

plt.figure(figsize = (10,10))
plt.subplot(2,2,1),plt.imshow(img_rgb)
plt.title('Citra Asli')
plt.subplot(2,2,2),plt.imshow(mask, 'gray')
plt.title('Citra Segment')
color = ('r','g','b')

for i,col in enumerate(color):
    hist0 = cv2.calcHist([img_rgb],[i],None,[256],[0,256])
    plt.subplot(2,2,3),plt.plot(hist0,color = col)
    plt.title('Histogram dari keseluruhan citra')
    hist1 = cv2.calcHist([img_rgb],[i],mask,[256],[0,256])
    plt.subplot(2,2,4),plt.plot(hist1,color = col)
    plt.title('Histogram dari keseluruhan citra')
    hist1 = cv2.calcHist([img_rgb],[i],mask,[256],[0,256])
    plt.subplot(2,2,4),plt.plot(hist1,color = col)
    plt.title('Histogram hanya pada FG objek saja')
    plt.xlim([0,256])

plt.show()
```

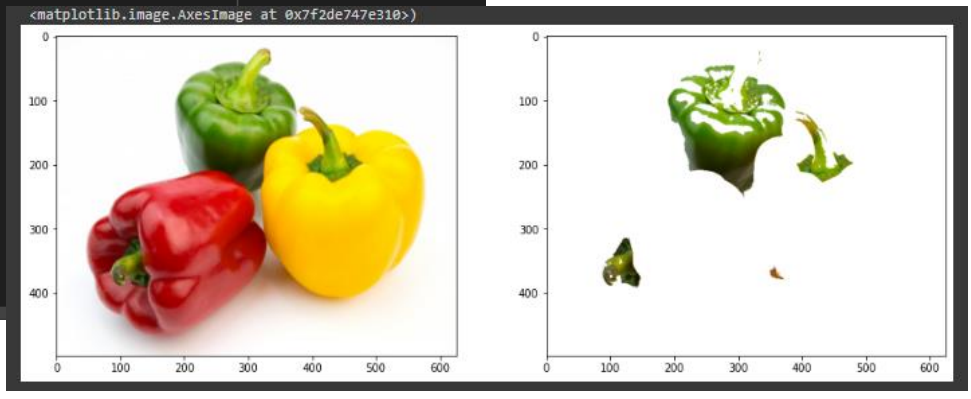
OUTPUT



5. Perform color segmentation on the image "peppers.jpg", display only yellow colors . (Hint: you can use K-Means to display only certain colors)

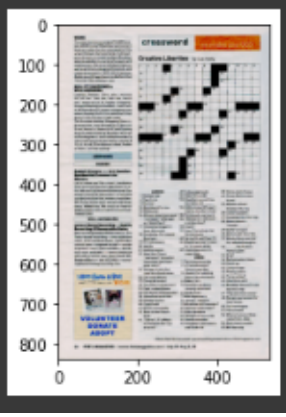
```
# No 6
filename_4 = ('/content/drive/MyDrive/PCVK_MM/Images/peppers.jpg')
img_p = cv2.imread(filename_4)
img_rgb2 = cv2.cvtColor(img_p,cv2.COLOR_BGR2RGB)

pixel_values = img_rgb2.reshape((-1, 3))
pixel_values = np.float32(pixel_values)

# syarat berhenti iterasi dr KMeans adalah jika centroid sudah tidak terlalu banyak pergeseran posisi antara

criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 100, 0.2)

# Jika diperhatikan pada image asli, terdapat 3 warna utama (hijau, biru, dan putih/orange). Untuk percobaan

k = 6
_, labels, (centers) = cv2.kmeans(pixel_values, k, None, criteria, 10, cv2.KMEANS_RANDOM_CENTERS)

# konversi titik centroid ke dalam integer
centers = np.uint8(centers)

# flattening label array
labels = labels.flatten()

masked_image = np.copy(img_rgb2)
masked_image0 = masked_image.reshape((-1, 3))
masked_image0[labels != 1] = [255, 255, 255]
masked_image0 = masked_image0.reshape(img_rgb2.shape)

plt.figure(figsize = (15,12))
plt.subplot(2,2,1),plt.imshow(img_rgb2)
plt.subplot(2,2,2),plt.imshow(masked_image0)
```

OUTPUT



6. Open the crossword.jpg file. With the knowledge of thresholding that you have learned. Perform binary thresholding with the best results in your opinion. Copy the code and image results in this module.

```
# No 7
img_c = cv2.imread('/content/drive/MyDrive/PCVK_MM/Images/crossword.jpg')
img_rgb3 = cv2.cvtColor(img_c, cv2.COLOR_BGR2RGB)

ret, thresh = cv2.threshold(img_rgb3, 125, 255, cv2.THRESH_TOZERO)

plt.imshow(thresh)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



https://github.com/Rjndrkha/PCVK_Genap_2022