شماره فصل: ۷	تعداد جلسات مورد نیاز: ۱ جلسه
اهداف: – آشنایی با انواع روش‌های قطعه‌بندی تصویر	

قطعه بندی با روش آستانه‌گذاری ساده:

```
## Simple thresholding ##
import cv2
from matplotlib import pyplot as plt # or use: import matplotlib.pyplot as
plt
img=cv2.imread('E:/UOK/UOK_ Presentations/UOK_Digital Image
Processing/Python/Images/Fruits.jpeg')
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) #Convert BGR image to
grayscale
ret,thr1 = cv2.threshold(gray_img,127,255,cv2.THRESH_BINARY) #If pixel
intensity is greater than the set threshold, value set to 255 (white), else
set to 0 (black).
ret,thr2 = cv2.threshold(gray_img,127,255,cv2.THRESH_BINARY_INV) #Inverted
or Opposite case of cv2.THRESH_BINARY
ret,thr3 = cv2.threshold(gray_img,127,255,cv2.THRESH_TRUNC) #If pixel
intensity value is greater than threshold, it is truncated to the
threshold. The pixel values are set to be the same as the threshold. All
other values remain the same.
ret,thr4 = cv2.threshold(gray_img,127,255,cv2.THRESH_TOZERO) #Pixel
intensity is set to 0, for all the pixels intensity less than the threshold
value. All other values remain the same.
ret,thr5 = cv2.threshold(gray_img,127,255,cv2.THRESH_TOZERO_INV) #Inverted
or Opposite case of cv2.THRESH_TOZERO.
titles = ['Grayscale
image','BINARY','BINARY_INV','TRUNC','TOZERO','TOZERO_INV']
images = [gray img, thr1, thr2, thr3, thr4, thr5]
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()
###
```

قطعه بندی با روش آستانه‌گذاری اوتسو:

```
## Otsu's thresholding ##
import cv2
img=cv2.imread('E:/UOK/UOK_ Presentations/UOK_Digital Image
Processing/Python/Images/Fruits.jpeg')
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) #Convert BGR image to
grayscale
ret,thr = cv2.threshold(gray_img, 0, 255, cv2.THRESH_OTSU) # Otsu's
thresholding
cv2.imshow('Grayscale image', gray_img)
cv2.imshow('Image segmented by Otsu thresholding', thr)
cv2.waitKey(0)
###
```

قطعه بندی با روش آستانه‌گذاری الگوریتم مثلث:

```
## Triangle algorithm thresholding ##
import cv2
img=cv2.imread('E:/UOK/UOK_ Presentations/UOK_Digital Image
Processing/Python/Images/Fruits.jpeg')
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) #Convert BGR image to
grayscale
ret,thr = cv2.threshold(gray_img, 0, 255, cv2.THRESH_TRIANGLE) #
thresholding by triangle algorithm
cv2.imshow('Grayscale image', gray_img)
cv2.imshow('Image segmented by triangle algorithm', thr)
cv2.waitKey(0)
###
```

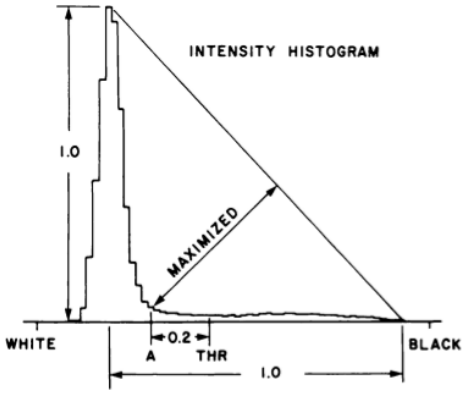

FIG. 2. Determination of the global search threshold for each picture. The threshold (THR) was selected by normalizing the height and dynamic range of the intensity histogram, locating point A as shown, and then adding a fixed offset.

قطعه بندی با روش آستانه‌گذاری تطبیقی:

```
## Adaptive thresholding ##
import cv2
from matplotlib import pyplot as plt # or use: import matplotlib.pyplot as
plt
img=cv2.imread('E:/UOK/UOK_ Presentations/UOK_Digital Image
Processing/Python/Images/Paper.jpg')
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) #Convert BGR image to
grayscale
ret1,thr1 = cv2.threshold(gray_img,127,255,cv2.THRESH_BINARY) #Simple
thresholding
ret2,thr2 = cv2.threshold(gray_img,0,255,cv2.THRESH_OTSU) #Otsu's
thresholding
ret3,thr3 = cv2.threshold(gray_img,0,255,cv2.THRESH_TRIANGLE) #Thresholding
by triangle algorithm
thr4 =
cv2.adaptiveThreshold(gray_img,255,cv2.ADAPTIVE_THRESH_MEAN_C,cv2.THRESH_BI
NARY,11,2) #Adaptive thresholding with arithmetic mean of the local pixel
neighborhood to compute the threshold value
thr5 =
cv2.adaptiveThreshold(gray_img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRES
H_BINARY,11,2) #Adaptive thresholding with Gaussian mean of the local pixel
neighborhood to compute the threshold value
```

```
titles = ['Grayscale image', 'Global thresholding (v = 127)', 'Otsu
thresholding', 'Triangle algorithm thresholding', 'Adaptive mean
thresholding', 'Adaptive Gaussian thresholding']
images = [gray_img, thr1, thr2, thr3, thr4, thr5]
for i in range(6):
    plt.subplot(2,3,i+1),plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])
plt.show()
###
```