2019

Digital Image Processing

LAB 8 REPORT

MISHAAL SAFEER

SAMEEN ABBAS

FOHA KHALID

HASHIR SHOAIB

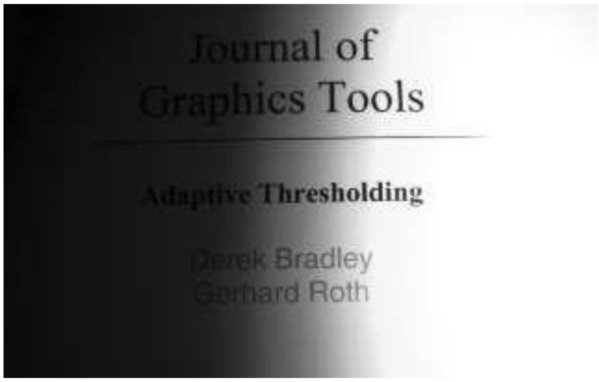
LAB # 08: Image Segmentation

Lab Objective:
The objective of this lab is to apply differential spatial filters to segment an image.

Lab Task 1:
Write a program that threshold the provided image using global mean and median.

Code:

```
import cv2 as cv
import numpy as np

img = cv.imread('Lab8Img.jpg', 0)


def task1(x):
    m = np.mean(x)
    n = np.median(x)
    print(m)
    print(n)
    retval, threshold_m = cv.threshold(x, m, 255, cv.THRESH_BINARY)

    retval, threshold_n = cv.threshold(x, n, 255, cv.THRESH_BINARY)

    cv.imshow('original', x)
    cv.imshow('Mean', threshold_m)
    cv.imshow('Median', threshold_n)

    cv.waitKey(0)
    cv.destroyAllWindows()


task1(img)
```

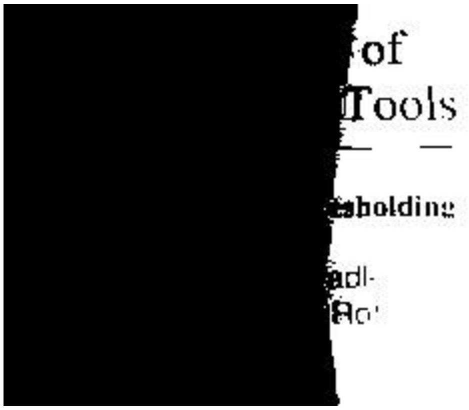


Journal of
Graphics Tools

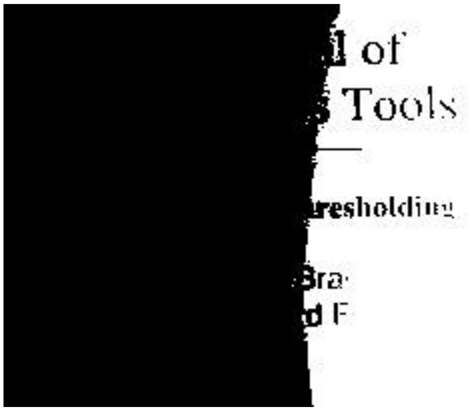
Adaptive Thresholding

Derek Bradley
Gerhard Roth

Original Image



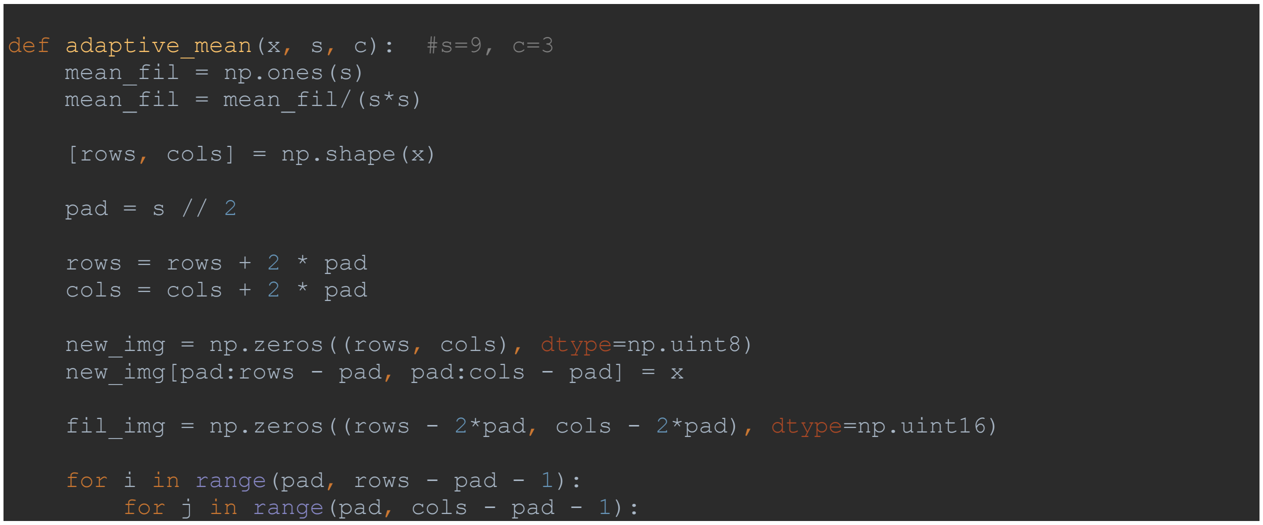
Global Mean Filter



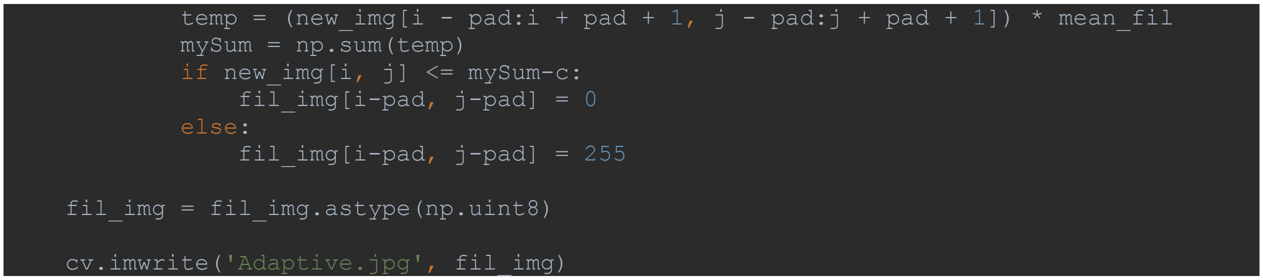
Global Median Filter

Lab Task 2:
Now threshold the image by taking threshold value mean of 3x3 block locally.

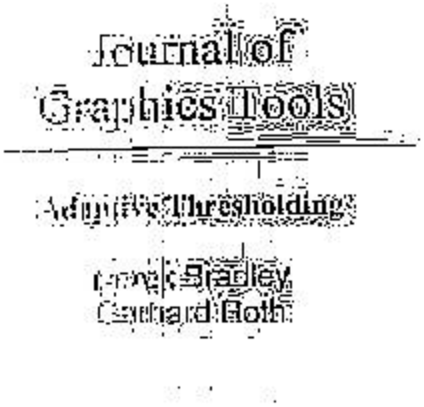
Code:



```
def adaptive_mean(x, s, c):  #s=9, c=3
    mean_fil = np.ones(s)
    mean_fil = mean_fil/(s*s)

    [rows, cols] = np.shape(x)

    pad = s // 2

    rows = rows + 2 * pad
    cols = cols + 2 * pad

    new_img = np.zeros((rows, cols), dtype=np.uint8)
    new_img[pad:rows - pad, pad:cols - pad] = x

    fil_img = np.zeros((rows - 2*pad, cols - 2*pad), dtype=np.uint16)

    for i in range(pad, rows - pad - 1):
        for j in range(pad, cols - pad - 1):
```



```
            temp = (new_img[i - pad:i + pad + 1, j - pad:j + pad + 1]) * mean_fil
            mySum = np.sum(temp)
            if new_img[i, j] <= mySum-c:
                fil_img[i-pad, j-pad] = 0
            else:
                fil_img[i-pad, j-pad] = 255

    fil_img = fil_img.astype(np.uint8)

    cv.imwrite('Adaptive.jpg', fil_img)
```

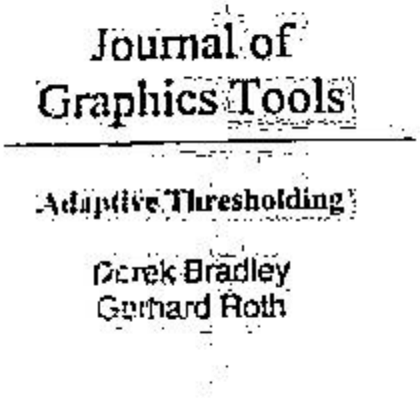


Journal of
Graphics Tools

Adaptive Thresholding

Derek Bradley
Gerhard Roth

Adaptive mean 3x3 with constant 2



Journal of
Graphics Tools

Adaptive Thresholding

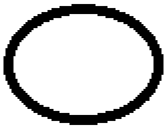
Derek Bradley
Gerhard Roth

Adaptive mean 9x9 with constant 3

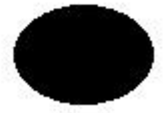
Lab Task 3:
Write a program that take image of the circle as input and applies region growing on it. Take center
of the image as seed point. Display the resultant image.

Code:

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

img = cv.imread('hello.png', 0)

size = np.shape(img)
M = np.zeros(size, dtype=np.uint8)
thresh = 6


plt.figure()
plt.imshow(img)
plt.gray()

print('\nPlease select the initial seed point')

pseed = plt.ginput(1)
p = int(pseed[0][0])
q = int(pseed[0][1])

seed = []
seed.append(p)
seed.append(q)

print('you clicked:', seed)

plt.close()

my_region = np.zeros(size, dtype=np.uint8)
my_region[p][q] = 255
M[p][q] = 1
check_list = [seed]


def check_neighbours(x, y):
    for i in range(-1, 2):
        for j in range(-1, 2):
            if x+i<size[0] and y+j<size[1]:
                if M[x + i][y + j] != 1:
                    if abs(int(img[x + i][y + j]) - int(img[p][q])) <= thresh:
                        z = [x + i, y + j]
                        my_region[x + i][y + j] = 255
                        check_list.append(z)
                M[x + i][y + j] = 1
    check_list.remove([x, y])


def region_grow():
    while len(check_list) > 0:
        z = check_list[0]
        x = z[0]
        y = z[1]
        check_neighbours(x, y)


region_grow()
my_region = np.invert(my_region)
cv.imwrite('Region.jpg', my_region)
```



Original Image



After Region Growing