Pengolahan Citra

**Pekan 4. Filtering LPH dan HPF in domain spatial**

**Dosen Pengampu**

Hero Yudo Martono ST, MT



**Disusun Oleh :**

Nama : M. Faza Nur Husain

Nrp : 3121550004

**D3 PJJ AK TEKNIK INFORMATIKA**

**POLITEKNIK ELEKTRONIKA NEGERI SURABAYA**

**TAHUN AKADEMIK 2021/2022**

**Source Code :**

def convolution2D():

    img1 = cv2.imread('gambar/kuda.jpg')

    img1 = cv2.cvtColor(img1, cv2.COLOR\_BGR2RGB)

    kernel = np.ones((3, 3), np.float32) / 9

    #print (kernel)

    img2 = cv2.filter2D(img1, -1, kernel)

    kernel = np.array([[0,  -1, 0],

                       [-1, 5, -1],

                       [0, -1, 0]])

    img3 = cv2.filter2D(img1, -1, kernel)

    img4 = cv2.blur(img1, (5, 5))

    img5 = cv2.GaussianBlur(img1, (3, 3), 0)

    img6 = cv2.medianBlur(img1, 3)

    titles = ['Original Image', 'Filter 1/9',

              'Sharpen', 'Blur', 'Gaussian', 'Median Blur']

    images = [img1, img2, img3, img4, img5, img6]

    for i in range(6):

        plt.subplot(3, 2, i+1), plt.imshow(images[i], 'gray', vmin=0, vmax=255)

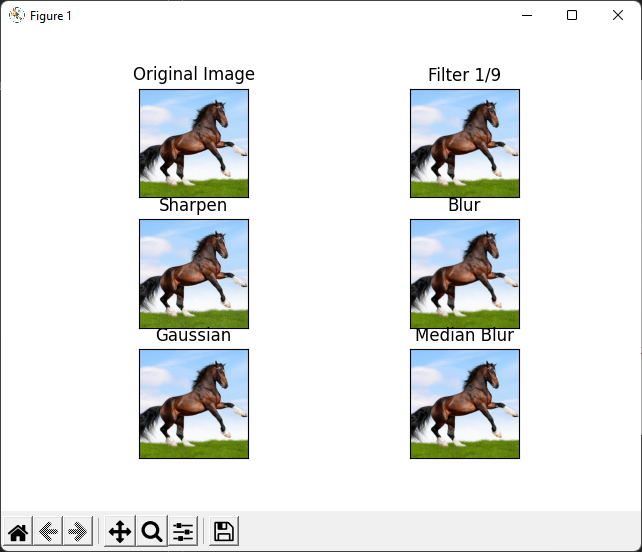
        plt.title(titles[i])

        plt.xticks([]), plt.yticks([])

    plt.show()

    return

convolution2D()



**Source Code :**

def dilatation():

    img1 = cv2.imread('gambar/kuda.jpg')

    # convert to black and white

    img1 = cv2.cvtColor(img1, cv2.COLOR\_BGR2GRAY)

    r, img1 = cv2.threshold(img1, 150, 255, cv2.THRESH\_BINARY)

    # create kernel

    kernel = np.ones((5, 5), np.uint8)

    img2 = cv2.erode(img1, kernel)

    img3 = cv2.dilate(img1, kernel)

    img4 = cv2.morphologyEx(img1, cv2.MORPH\_GRADIENT, kernel)

    img5 = cv2.GaussianBlur(img1, (3, 3), 0)

    img6 = cv2.medianBlur(img1, 3)

    titles = ['Original Image', 'Erosion', 'Dilatation',

              'morphologyEx', 'Gaussian', 'Median Blur']

    images = [img1, img2, img3, img4, img5, img6]

    for i in range(6):

        plt.subplot(3, 2, i+1), plt.imshow(images[i], 'gray', vmin=0, vmax=255)

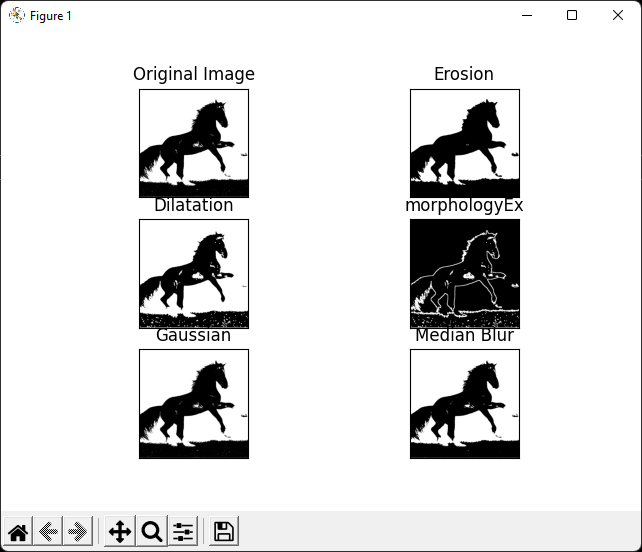
        plt.title(titles[i])

        plt.xticks([]), plt.yticks([])

    plt.show()

    return

dilatation()

****

**Source Code :**

def filtering():

    img1 = cv2.imread('gambar/kuda.jpg')

    kernel = np.array([[1, 1, 1, 1, 1],

                      [1, 1, 1, 1, 1],

                       [1, 1, 1, 1, 1],

                       [1, 1, 1, 1, 1],

                       [1, 1, 1, 1, 1]])

    kernel = kernel/25

    img2 = cv2.filter2D(img1, -1, kernel)

    kernel = np.array([[0.0, -1.0, 0.0],

                      [-1.0, 4.0, -1.0],

                       [0.0, -1.0, 0.0]])

    kernel = kernel/(np.sum(kernel) if np.sum(kernel) != 0 else 1)

    img3 = cv2.filter2D(img1, -1, kernel)

    kernel = np.array([[0.0, -1.0, 0.0],

                      [-1.0, 5.0, -1.0],

                       [0.0, -1.0, 0.0]])

    kernel = kernel/(np.sum(kernel) if np.sum(kernel) != 0 else 1)

    img4 = cv2.filter2D(img1, -1, kernel)

    # img4= cv2.morphologyEx(img1, cv2.MORPH\_GRADIENT, kernel)

    # img5= cv2.GaussianBlur(img1, (3,3), 0)

    # img6= cv2.medianBlur(img1, 3)

    kernel = np.array([[-1.0, -1.0, ],

                      [2.0, 2.0],

                       [-1.0, -1.0]])

    kernel = kernel/(np.sum(kernel) if np.sum(kernel) != 0 else 1)

    img5 = cv2.filter2D(img1, -1, kernel)

    titles = ['original image', 'low pass', 'high pass',

              'high pass', 'cusom kernel', 'normal']

    images = [img1, img2, img3, img4, img5, img1]

    for i in range(6):

        plt.subplot(

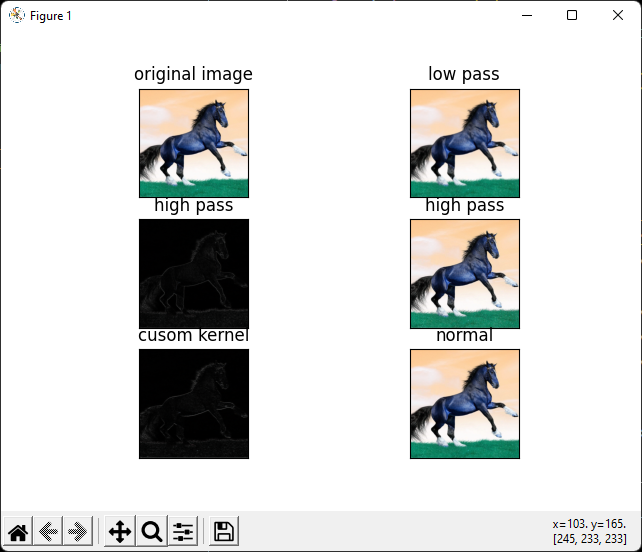
            3, 2, i+1), plt.imshow(images[i], 'gray', vmin=-0, vmax=255)

        plt.title(titles[i])

        plt.xticks([]), plt.yticks([])

    plt.show()

    return

****