Tugas 2 Pengolahan Citra Video
Filter Spasial

Kenanya Keandra Adriel Prasetyo
5024211004

September 17, 2023

1 Pendahuluan

Pada Tugas 2 mata kuliah Pengolahan Citra Video, penulis diminta untuk membuat program yang sama dengan yang diberikan namun dengan menggunakan library opencv dan membandingkan kecepatan dari 2 program tersebut.

2 Pembahasan Program

Kode keseluruhan pada Tugas 2 Pengolahan Citra Video ini dapat diakses pada https://github.com/KenanyaKAP/pcv_2

2.1 Low Pass Filter Manual

```
import time
import cv2
import numpy as np

# Convolution Function
def Convolusi(f,w):
    f = np.float64(f)/255
    baris = f.shape[0]
    kolom = f.shape[1]
    dkernel = w.shape[0]
    dkernel2= np.int32(np.floor(dkernel/2))
    g =np.zeros((baris,kolom,3))
    for y in range(baris):
        for x in range(kolom):
```

```
            g[y,x] = 0
            for i in range(dkernel):
                yy =y+i-dkernel2
                if (yy<0)|(yy>=baris-1):
                    continue
                for j in range(dkernel):
                    xx =x+j - dkernel2
                    if (xx<0)|(xx>=kolom-1):
                        continue
                    g[y,x]=g[y,x]+f[yy,xx]*w[i,j]
    return g

# Read image
img = cv2.imread("image.jpg")

# Making kernel
dkernel = 5
kernel = np.ones((dkernel,dkernel))/(dkernel*dkernel)

# Start time
start = time.time()

# Apply convolution
imgResult = Convolusi(img, kernel)

# End time
end = time.time()

print("Execution time: ",(end-start), "second")

# Show images
cv2.imshow('Citra Asli', img)
cv2.imshow('Hasil Convolusi', imgResult)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Program di atas akan melakukan Low Pass Filter pada citra yang telah dipilih. Filter dilakukan secara manual dengan melakukan for loop pada tiap-tiap nilai pada pixel baris dan kolom gambar. Program di atas menggunakan lebar kernel sebesar 5x5

Untuk dapat menghitung waktu yang diperlukan program, digunakan library time. Waktu dihitung dimulai dari sebelum melakukan convolusi pada gambar. Yakni dengan menggunakan variable start untuk mengambil waktu saat program dijalankan

```
# Start time
start = time.time()

# Apply convolution
imgResult = Convolusi(img, kernel)

# End time
end = time.time()
```

```
9
10 print ( " Execution time : " ,( end - start ) , " second " )
```

Kemudian setelah convolusi sudah dilakukan pada citra, kembali dilakukan pengambilan waktu, yang ditampung pada variable end. Untuk menampilkan jumlah waktu yang diperlukan program, hanya tinggal mengurangi waktu akhir dan waktu awal.

Berikut adalah waktu yang diperlukan program untuk melakukan convolusi pada citra

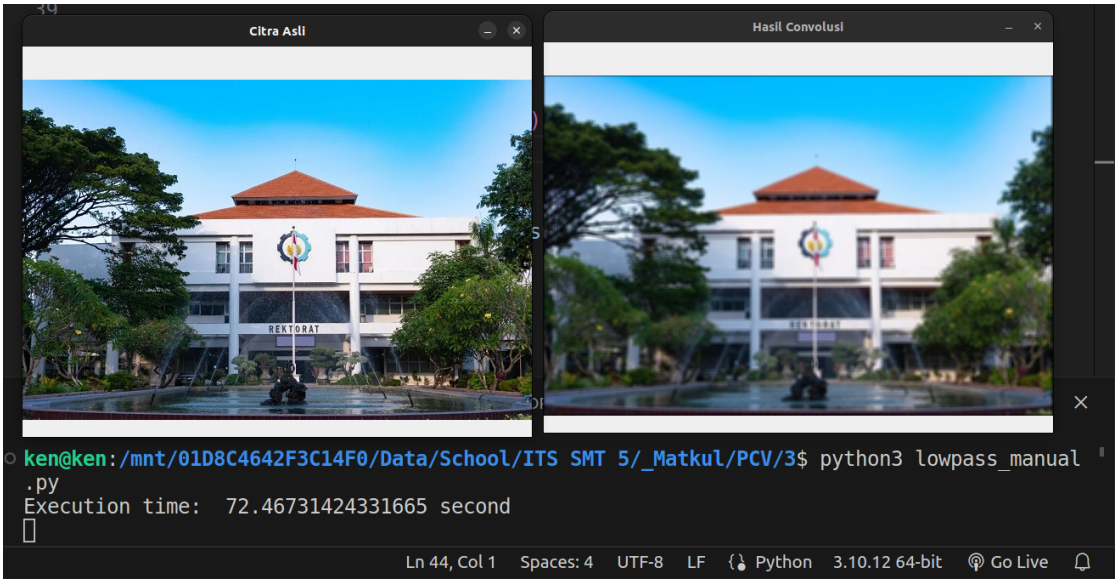

Figure 1: LP Manual

Dapat dilihat dengan menggunakan cara looping manual, gambar rgb, lebar kernel 5x5, program memerlukan waktu selama 72 detik

2.2 Low Pass Filter Opencv

```
1  import time
2  import cv2
3  import numpy as np
4
5  # Read image
6  img = cv2.imread('image.jpg')
7
8  # Making kernel
9  kernel = np.ones((5, 5))
10 kernel = kernel/(np.sum(kernel) if np.sum(kernel)!=0 else 1)
11
12 # Start time
```

```
start = time.time()

# Apply convolution
imgResult = cv2.filter2D(img,-1,kernel)

# End time
end = time.time()

print("Execution time: ",(end-start), "second")

# Show images
cv2.imshow('Citra Asli', img)
cv2.imshow('Hasil Convolosi', imgResult)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Program di atas akan melakukan Low Pass Filter yang sama dengan sebelumnya. Hanya saja filter dilakukan dengan library opencv. Program di atas juga tetap menggunakan lebar kernel yang sama yakni sebesar 5x5

Untuk dapat menghitung waktu yang diperlukan program, digunakan library time. Waktu dihitung dimulai dari sebelum melakukan convolusi pada gambar. Yakni dengan menggunakan variable start untuk mengambil waktu saat program dijalankan

```
# Start time
start = time.time()

# Apply convolution
imgResult = cv2.filter2D(img,-1,kernel)

# End time
end = time.time()

print ( " Execution time : " ,( end - start ) , " second " )
```

Kemudian setelah convolusi sudah dilakukan pada citra, kembali dilakukan pengambilan waktu, yang ditampung pada variable end. Untuk menampilkan jumlah waktu yang diperlukan program, hanya tinggal mengurangi waktu akhir dan waktu awal.

Berikut adalah waktu yang diperlukan program untuk melakukan convolusi pada citra

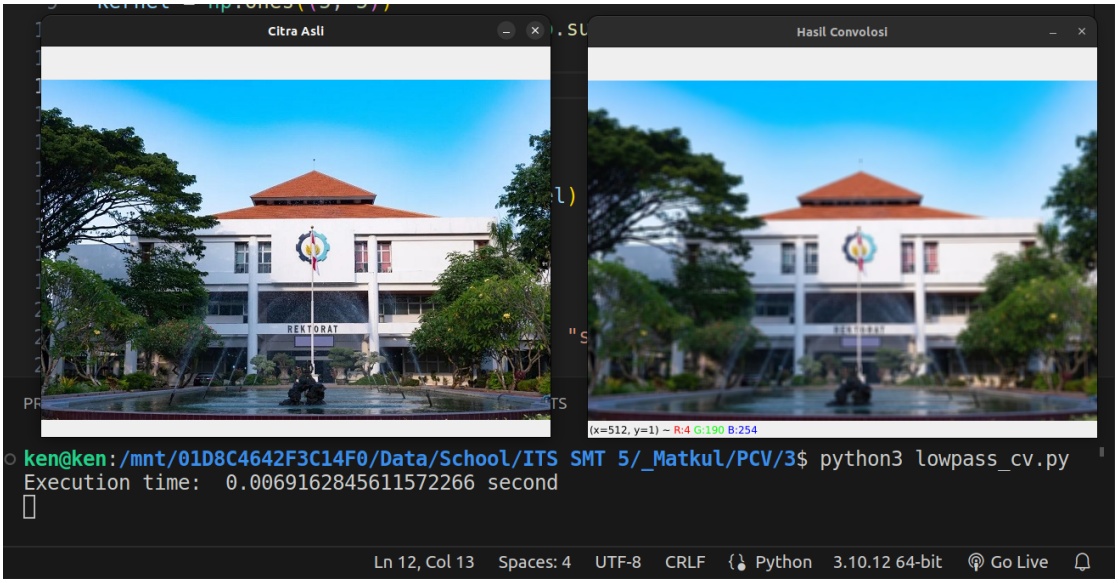

Figure 2: LP Opencv

Dapat dilihat dengan menggunakan library opencv pada gambar yang sama, lebar kernel 5x5, program hanya memerlukan waktu selama 0.0067 detik atau sekitar 7 milisecond. Dapat dilihat perbandingan kecepatan yang sangat signifikan apabila menggunakan library opencv untuk melakukan convolusi citra.

2.3 High Pass Filter Manual

```
import time
import cv2
import numpy as np

# Convolution Function
def Convolusi(f,w):
    f = np.float64(f)/255
    baris = f.shape[0]
    kolom = f.shape[1]
    dkernel = w.shape[0]
    dkernel2= np.int32(np.floor(dkernel/2))
    g =np.zeros((baris,kolom,3))
    for y in range(baris):
        for x in range(kolom):
            g[y,x] = 0
            for i in range(dkernel):
                yy =y+i-dkernel2
                if (yy<0)|(yy>=baris-1):
                    continue
                for j in range(dkernel):
```

```
                    xx =x+j - dkernel2
                    if (xx<0)|(xx>=kolom-1):
                        continue
                    g[y,x]=g[y,x]+f[yy,xx]*w[i,j]
    return g

# Read image
img = cv2.imread("image.jpg")

# Making kernel
kernel = np.array([[1, 1, 1],
                   [1,-8, 1],
                   [1, 1, 1]])

# Start time
start = time.time()

# Apply convolution
imgResult = Convolusi(img, kernel)

# End time
end = time.time()

print("Execution time: ",(end-start), "second")

# Show images
cv2.imshow('Citra Asli', img)
cv2.imshow('Highpass Filter', imgResult)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Program di atas akan melakukan High Pass Filter pada citra yang telah dipilih. Filter dilakukan secara manual dengan melakukan for loop pada tiap-tiap nilai pada pixel baris dan kolom gambar. Program di atas menggunakan lebar kernel sebesar 3x3 dan dengan filter yakni [[1, 1, 1],[1,-8, 1],[1, 1, 1]].

Untuk dapat menghitung waktu yang diperlukan program, digunakan library time. Waktu dihitung dimulai dari sebelum melakukan convolusi pada gambar. Yakni dengan menggunakan variable start untuk mengambil waktu saat program dijalankan

```
# Start time
start = time.time()

# Apply convolution
imgResult = Convolusi(img, kernel)

# End time
end = time.time()

print ( " Execution time : " ,( end - start ) , " second " )
```

Kemudian setelah convolusi sudah dilakukan pada citra, kembali dilakukan pengam-

bilan waktu, yang ditampung pada variable end. Untuk menampilkan jumlah waktu yang diperlukan program, hanya tinggal mengurangi waktu akhir dan waktu awal.

Berikut adalah waktu yang diperlukan program untuk melakukan convolusi pada citra

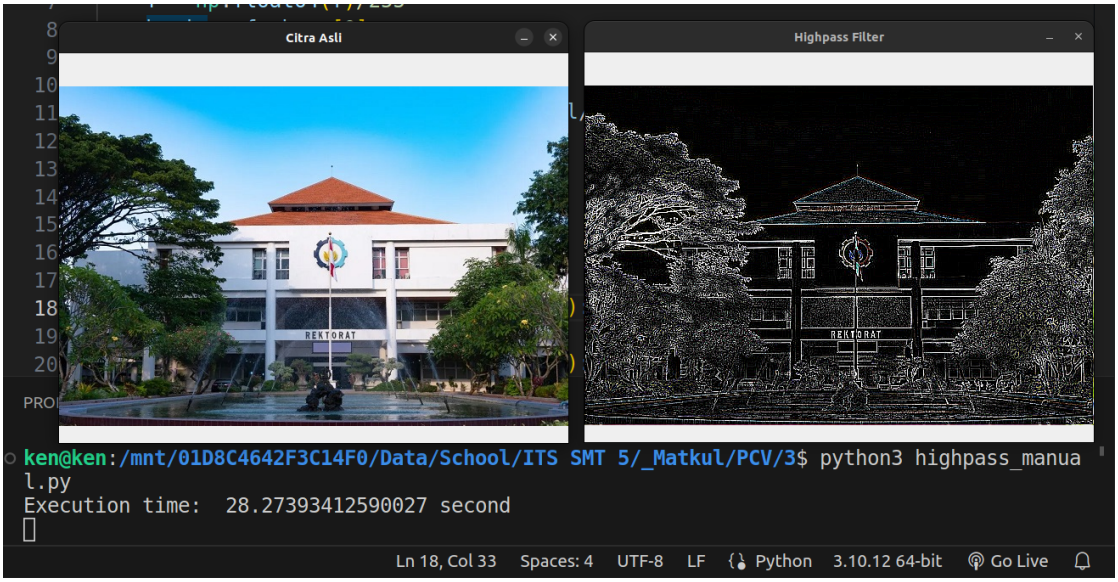

Figure 3: HP Manual

Dapat dilihat dengan menggunakan cara looping manual, gambar rgb, lebar kernel 3x3, program memerlukan waktu selama 28 detik

2.4 High Pass Filter Opencv

```
import time
import cv2
import numpy as np

# Read image
img = cv2.imread('image.jpg')

# Making kernel
kernel = np.array([[1, 1, 1],
                   [1,-8, 1],
                   [1, 1, 1]])
kernel = kernel/(np.sum(kernel) if np.sum(kernel)!=0 else 1)

# Start time
start = time.time()

```

```
# Apply convolution
imgResult = cv2.filter2D(img,-1,kernel)

# End time
end = time.time()

print("Execution time: ",(end-start), "second")

# Show images
cv2.imshow('Citra Asli', img)
cv2.imshow('Hasil Convolosi', imgResult)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Program di atas akan melakukan High Pass Filter yang sama dengan sebelumnya. Hanya saja filter dilakukan dengan library opencv. Program di atas juga tetap menggunakan lebar kernel yang sama yakni sebesar 3x3 dan dengan filter yang sama yakni [[1, 1, 1],[1,-8, 1],[1, 1, 1]].

Untuk dapat menghitung waktu yang diperlukan program, digunakan library time. Waktu dihitung dimulai dari sebelum melakukan convolusi pada gambar. Yakni dengan menggunakan variable start untuk mengambil waktu saat program dijalankan

```
# Start time
start = time.time()

# Apply convolution
imgResult = cv2.filter2D(img,-1,kernel)

# End time
end = time.time()

print ( " Execution time : " ,( end - start ) , " second " )
```

Kemudian setelah convolusi sudah dilakukan pada citra, kembali dilakukan pengambilan waktu, yang ditampung pada variable end. Untuk menampilkan jumlah waktu yang diperlukan program, hanya tinggal mengurangi waktu akhir dan waktu awal.

Berikut adalah waktu yang diperlukan program untuk melakukan convolusi pada citra

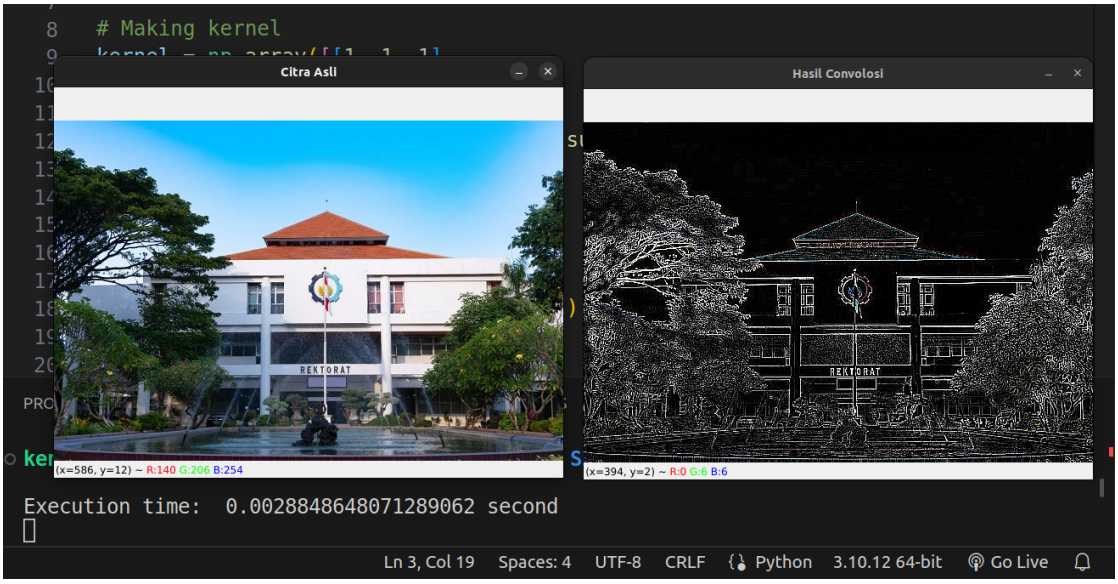

Figure 4: HP Opencv

Dapat dilihat dengan menggunakan library opencv pada gambar yang sama, lebar kernel 3x3, program hanya memerlukan waktu selama 0.0028 detik atau sekitar 3 milisecond. Dapat dilihat perbandingan kecepatan yang sangat signifikan apabila menggunakan library opencv untuk melakukan convolusi citra.