

# PENGOLAHAN CITRA DIGITAL



NAMA MAHASISWA	: NOFI ABDI NURHIKMAH
NIM / ROMBEL	: 5301414056 / 2
NAMA DOSEN PENGAMPU	: Dr. Hadi Wibawanto, M.T Kuntoro Adi Nugroho, S.T.,M.Eng.

PENDIDIKAN TEKNIK ELEKTRO  
JURUSAN TEKNIK ELEKTRO FAKULTAS TEKNIK  
UNIVERSITAS NEGERI SEMARANG

2017

## Tugas !

Buatlah filter image menggunakan Low Pass Filter dan High Pass Filter kemudian buat Histogram dari hasil filter tersebut!

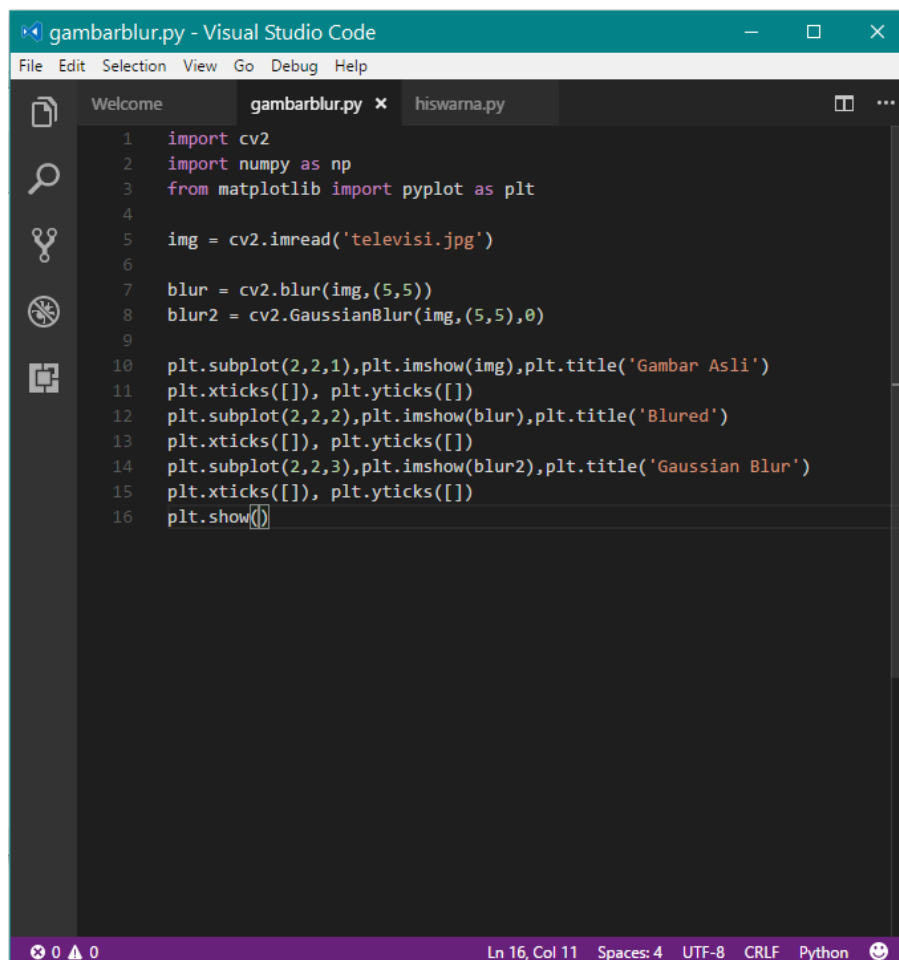
## Jawaban !

Image Filtering dapat dikelompokkan menjadi dua jenis berdasarkan efeknya:

### 1. Low pass filters (Smoothing)

Low pass filter (smoothing) bekerja dengan menggapus ruang derau frekuensi tinggi dari gambar digital. Low pass filters digunakan untuk menggeser jendela penghubung antara efek pixel pertama dari gambar pada waktu yang sama, mengubah nilai dari suatu fungsi lokal region piksel. LPF (low pass filter) digunakan untuk membantu menghapus noises, menghaburkan (blurring) gambar.

Fungsi low pass filter saya gunakan untuk mengaburkan gambar (blurring) dengan efek Average dan Gaussian. Saya menggunakan program OpenCV. Untuk file pythonnya saya simpan dengan nama **gambarblur.py**. Berikut source code nya:



```
gambarblur.py - Visual Studio Code
File Edit Selection View Go Debug Help

Welcome gambarblur.py x hiswarna.py

1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4
5 img = cv2.imread('televisi.jpg')
6
7 blur = cv2.blur(img,(5,5))
8 blur2 = cv2.GaussianBlur(img,(5,5),0)
9
10 plt.subplot(2,2,1),plt.imshow(img),plt.title('Gambar Asli')
11 plt.xticks([], plt.yticks([]))
12 plt.subplot(2,2,2),plt.imshow(blur),plt.title('Blured')
13 plt.xticks([], plt.yticks([]))
14 plt.subplot(2,2,3),plt.imshow(blur2),plt.title('Gaussian Blur')
15 plt.xticks([], plt.yticks([]))
16 plt.show()
```

Ln 16, Col 11 Spaces: 4 UTF-8 CRLF Python

Penjelasan script:

```
1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4
```

Digunakan untuk mengimpor modul atau library yang akan digunakan

```
5 img = cv2.imread('televisi.jpg')
6
```

Digunakan untuk melakukan inisialisasi pada gambar. Gambar yang akan digunakan diletakkan dalam satu folde dengan file python.

```
7 blur = cv2.blur(img,(5,5))
```

Menggunakan fungsi `cv2.blur()` untuk mengganti setiap nilai pixel didalam sebuah gambar dengan rata-rata nilai dari pixel tetanggannya, dan mencakup dirinya sendiri. Sehingga didapatkan efek gambar kabur. Fungsi ini juga dikenal dengan Averaging blurred, dimana gambar digulung dengan filter ternormal kotak. Penggunaan (5,5) adalah kernel lebar dan tinggi 5x5. Ini digunakan untuk merepresentasikan bentuk dan ukuran dari sekitarnya sebagai sample kemudian mengkalkulasi dengan rata-rata.

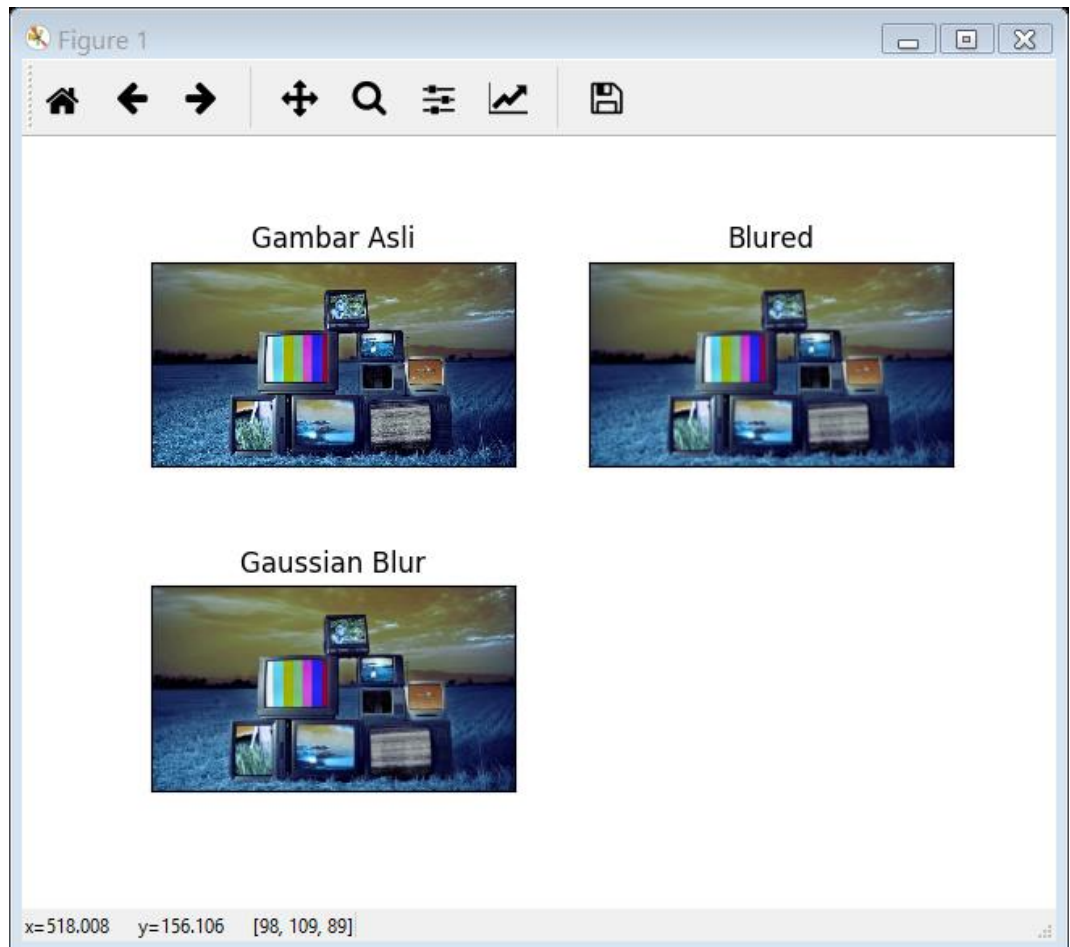
```
8 blur2 = cv2.GaussianBlur(img,(5,5),0)
```

Salah satu contoh aplikasi low pass filter adalah efek Gaussian bluring. Menggunakan fungsi `cv2.GaussianBlur()`. Dalam teori Gaussian, setiap titik dari gambar akan dihitung tidak-nol, sehingga seluruh gambar harus dihitung setiap pixelnya. (5,5) adalah ukuran kernel lebar dan tinggi.

```
10 plt.subplot(2,2,1),plt.imshow(img),plt.title('Gambar Asli')
11 plt.xticks([], plt.yticks([]))
12 plt.subplot(2,2,2),plt.imshow(blur),plt.title('Blured')
13 plt.xticks([], plt.yticks([]))
14 plt.subplot(2,2,3),plt.imshow(blur2),plt.title('Gaussian Blur')
15 plt.xticks([], plt.yticks([]))
16 plt.show()
```

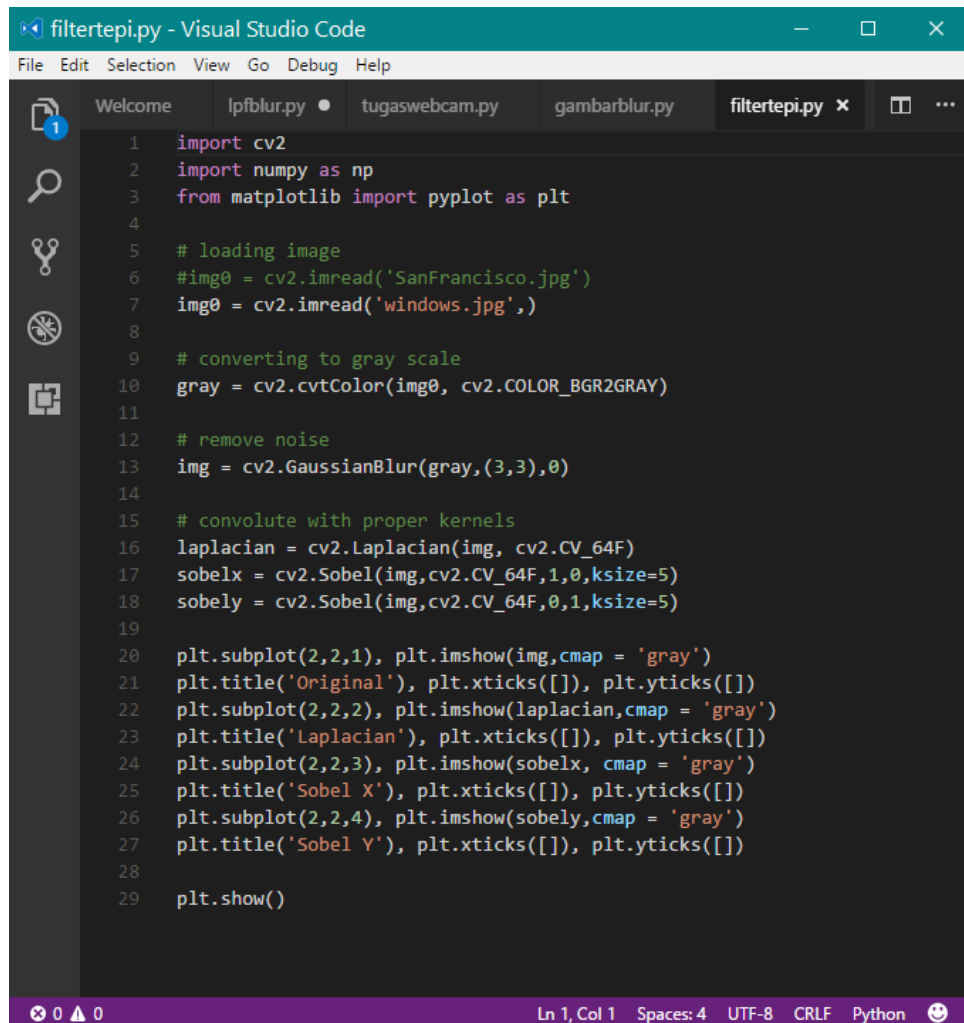
Digunakan untuk menampilkan gambar hasil kedalam satu figure.

Berikut hasil menggunakan low pass filter dengan efek Average dan Gaussian:



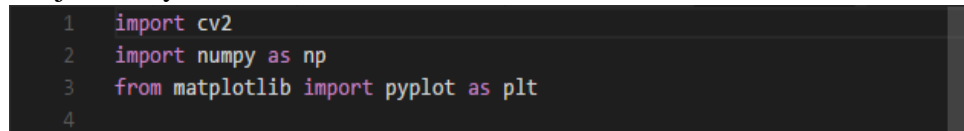
2. High pass filter bisa digunakan untuk membuat gambar terlihat tajam. HPF bekerja sama dengan low pass filter, hanya saja yang berbeda ada pada konvolusi kernel. Salah satu penggunaan HPF adalah menemukan tepi dari sebuah gambar.

HPF saya gunakan untuk menemukan tepi gambar. Edge detection (deteksi garis tepi) biasanya digunakan untuk mendeteksi tepi sebuah skema (sobel) penggunaan dasar deteksi tepi dan Laplacian. File python edge detection ini saya simpan dengan nama **filtertepi.py**. Berikut source code nya:



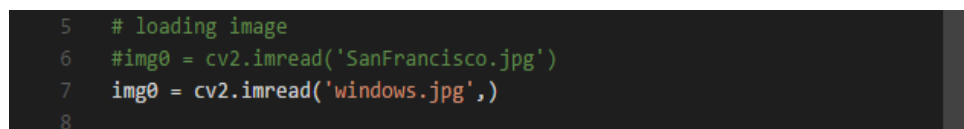
```
1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4
5 # loading image
6 #img0 = cv2.imread('SanFrancisco.jpg')
7 img0 = cv2.imread('windows.jpg',)
8
9 # converting to gray scale
10 gray = cv2.cvtColor(img0, cv2.COLOR_BGR2GRAY)
11
12 # remove noise
13 img = cv2.GaussianBlur(gray,(3,3),0)
14
15 # convolute with proper kernels
16 laplacian = cv2.Laplacian(img, cv2.CV_64F)
17 sobelx = cv2.Sobel(img,cv2.CV_64F,1,0,ksize=5)
18 sobely = cv2.Sobel(img,cv2.CV_64F,0,1,ksize=5)
19
20 plt.subplot(2,2,1), plt.imshow(img,cmap = 'gray')
21 plt.title('Original'), plt.xticks([], plt.yticks([]))
22 plt.subplot(2,2,2), plt.imshow(laplacian,cmap = 'gray')
23 plt.title('Laplacian'), plt.xticks([], plt.yticks([]))
24 plt.subplot(2,2,3), plt.imshow(sobelx, cmap = 'gray')
25 plt.title('Sobel X'), plt.xticks([], plt.yticks([]))
26 plt.subplot(2,2,4), plt.imshow(sobely,cmap = 'gray')
27 plt.title('Sobel Y'), plt.xticks([], plt.yticks([]))
28
29 plt.show()
```

Penjelasannya :



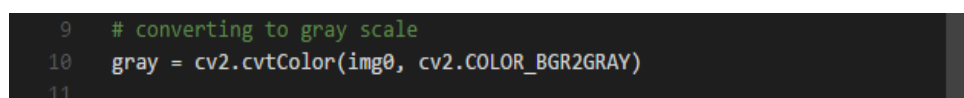
```
1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4
```

Digunakan untuk mengimpor modul atau library yang akan digunakan.



```
5 # loading image
6 #img0 = cv2.imread('SanFrancisco.jpg')
7 img0 = cv2.imread('windows.jpg',)
8
```

Digunakan untuk menginisialisasi gambar.



```
9 # converting to gray scale
10 gray = cv2.cvtColor(img0, cv2.COLOR_BGR2GRAY)
11
```

Digunakan untuk menginisialisasi gambar dan merubah gambar warna menjadi abu-abu.

```

12  # remove noise
13  img = cv2.GaussianBlur(gray,(3,3),0)
14

```

Digunakan menginisialisai gambar dan merubahnya menjadi efek kabur Gaussian, karena efek gaussian sangat baik digunakan untuk menghilangkan noise.

```

15  # convolute with proper kernels
16  laplacian = cv2.Laplacian(img, cv2.CV_64F)
17  sobelx = cv2.Sobel(img,cv2.CV_64F,1,0,ksize=5)
18  sobely = cv2.Sobel(img,cv2.CV_64F,0,1,ksize=5)
19

```

Digunakan untuk mengkonvolusi gambar dengan kernel yang tepat. Pada efek laplacian, cv2.CV\_64F digunakan untuk meminta ukuran gambar tujuan. Sedangkan sobelx menginisialisasi gambar menjadi efek sobel x-dir, dan sobely gambar di inialisasi menjadi efek sobel y-dir terhadap axes Y.

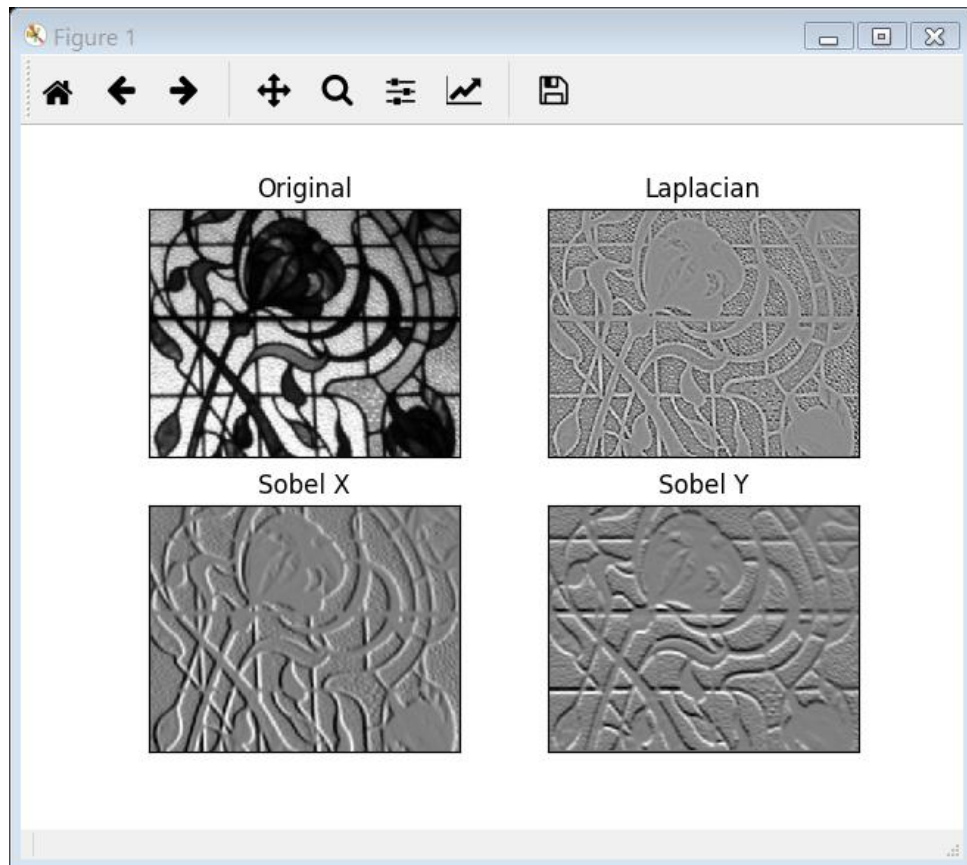
```

20  plt.subplot(2,2,1), plt.imshow(img,cmap = 'gray')
21  plt.title('Original'), plt.xticks([]), plt.yticks([])
22  plt.subplot(2,2,2), plt.imshow(laplacian,cmap = 'gray')
23  plt.title('Laplacian'), plt.xticks([]), plt.yticks([])
24  plt.subplot(2,2,3), plt.imshow(sobelx, cmap = 'gray')
25  plt.title('Sobel X'), plt.xticks([]), plt.yticks([])
26  plt.subplot(2,2,4), plt.imshow(sobely,cmap = 'gray')
27  plt.title('Sobel Y'), plt.xticks([]), plt.yticks([])
28
29  plt.show()

```

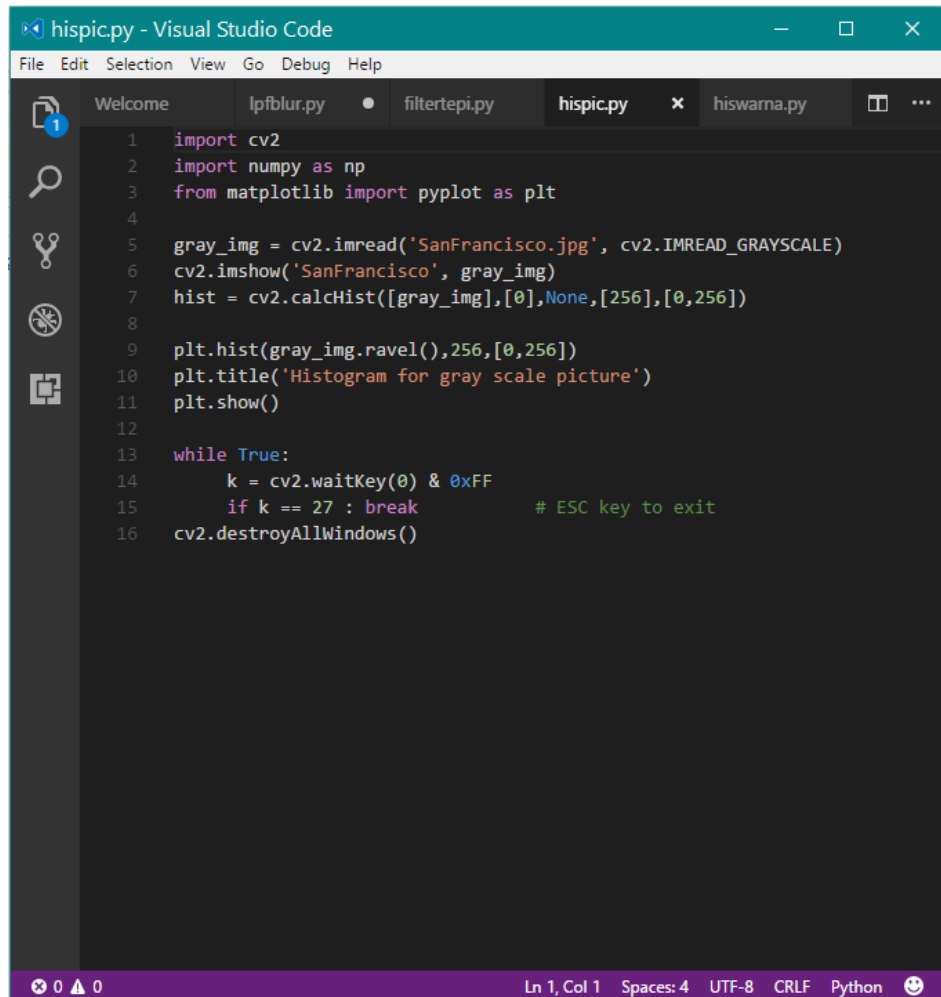
Digunakan untuk menampilkan hasil gambar kedalam satu figure.

Hasil :



3. Histogram digunakan untuk merepresentasikan graphical dari distribusi intensitas gambar. Digunakan pula untuk mengkuantisasi nomor pixel tiap nilai intensitas.

File python untuk histogram saya simpan dengan nama **hispic.py**. Berikut source code nya:



```
1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4
5 gray_img = cv2.imread('SanFrancisco.jpg', cv2.IMREAD_GRAYSCALE)
6 cv2.imshow('SanFrancisco', gray_img)
7 hist = cv2.calcHist([gray_img],[0],None,[256],[0,256])
8
9 plt.hist(gray_img.ravel(),256,[0,256])
10 plt.title('Histogram for gray scale picture')
11 plt.show()
12
13 while True:
14     k = cv2.waitKey(0) & 0xFF
15     if k == 27 : break          # ESC key to exit
16 cv2.destroyAllWindows()
```

Penjelasan :

```
1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4
```

Digunakan untuk mengimpor modul atau library yang akan digunakan

```
5 gray_img = cv2.imread('SanFrancisco.jpg', cv2.IMREAD_GRAYSCALE)
6 cv2.imshow('SanFrancisco', gray_img)
```

Digunakan untuk membaca gambar dan merubahnya menjadi gambar gray scale. Kemudian menampilkan nya kedalam figure.

```
7 hist = cv2.calcHist([gray_img],[0],None,[256],[0,256])
8
```

cv2.calcHist() adalah fungsi untuk histogram. [gray\_img], gambar yang digunakan adalah type uint8 atau folat32. [0] digunakan karena input gambar adalah grayscale, jika menggunakan gambar warna dapat menggunakan [0],[1] atau [2] untuk mengkalkulasi histogram channel biru, hijau dan merah, berturut-turut.



None, adalah mask image yang digunakan untuk menemukan histogram gambar penuh. [256] adalah histSize untuk skala penuh. [0,256] adalah range normal.

```
9 plt.hist(gray_img.ravel(),256,[0,256])
10 plt.title('Histogram for gray scale picture')
11 plt.show()
```

Digunakan untuk menampilkan gambar kedalam figure dengan nama 'Histogram for gray scale picture'

```
13 while True:
14     k = cv2.waitKey(0) & 0xFF
15     if k == 27 : break          # ESC key to exit
16     cv2.destroyAllWindows()
```

Perintah untuk menutup figure jika tombol ESC ditekan.

Hasil



