

# **MAKALAH PENGOLAHAN CITRA DIGITAL**



**NIM** : 2103065  
**Nama** : Deo Ananda Rizky  
**Kelas** : D3TI.2C

**PROGRAM STUDI D3 TEKNIK INFORMATIKA  
JURUSAN TEKNIK INFORMATIKA POLITEKNIK  
NEGERI INDRAMAYU  
2023**

## Tugas1

```
import cv2 as cv
import sys
img = cv.imread(cv.samples.findFile("fE:/admPerkuliahanSemGenap2022_2023/PCD/
sample_images/starry_night.jpg"))
if img is None: sys.exit("Could not read the image.")
cv.imshow("Display window", img)
k = cv.waitKey(0) if k == ord("s"):
cv.imwrite("starry_night.png", img)
```

**Jelaskan fungsi baris instruksi-instruksi pada kode program tersebut.**

Baris pertama import cv2 as cv berfungsi untuk mengimport library opencv yang sudah kita install lalu "as cv" digunakan untuk penamaan cv2 yang nantinya kita cukup memakai nama cv saja

Dan baris ke dua import sys berfungsi untuk menjalankan kode file python dilingkungan direktorinya sendiri.

Lalu baris selanjutnya img =

cv.imread(cv.samples.findFile("fE:/admPerkuliahanSemGenap2022\_2023/PCD/sample\_images/starry\_night.jpg")) kita menaruh fungsi open cv untuk mencari gambar yang ada di dalam direktori kita ke dalam variable img

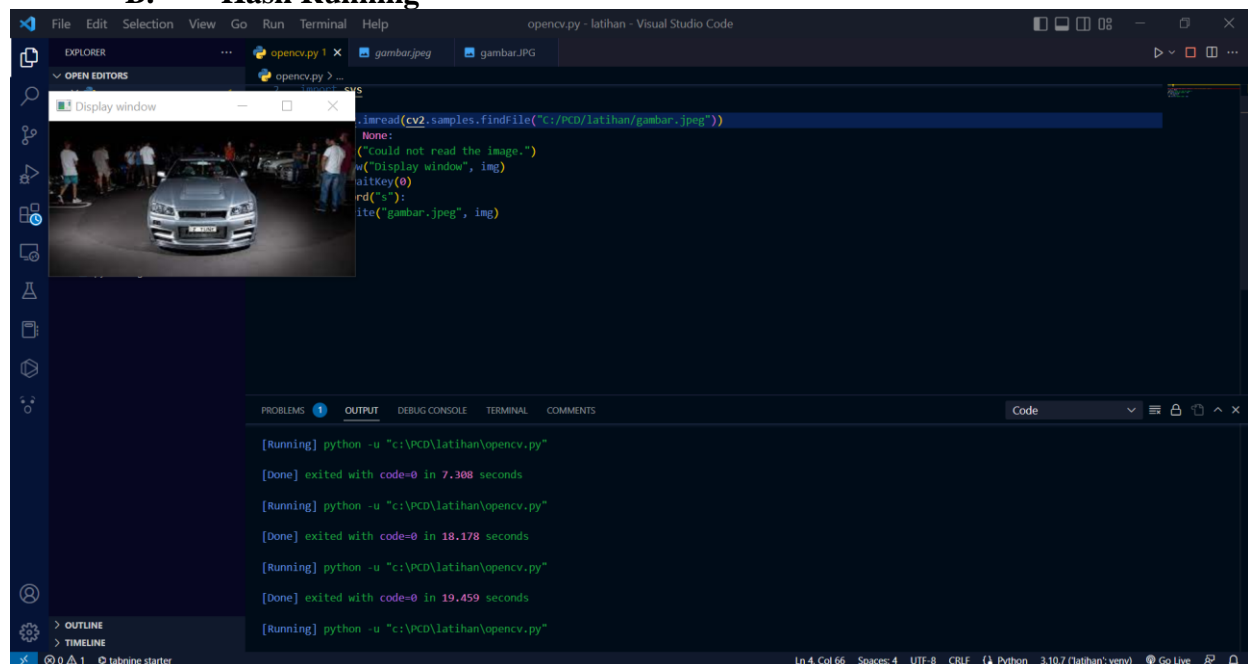
Lalu selanjutnya if img is None: sys.exit("Could not read the image.") berguna untuk pengecekan apakah ada atau tidak ada gambar yang kita cari, jika gambarnya tidak ada maka akan muncul pemberitahuan Could not read the image

Lalu cv.imshow("Display window", img) berfungsi untuk memunculkan gambar yang kita cari tadi ke layar monitor kita

Lalu k = cv.waitKey(0) if k == ord("s"): berfungsi untuk penghitung waktu dalam milidetik maksudnya adalah jika angka 0 dilewati maka akan menunggu tanpa batas waktu untuk stroke kunci

Lalu cv.imwrite("starry\_night.png", img) untuk menyimpan gambarnya pertama adalah nama file yang kedua adalah gambar yang ingin kita simpan

## B. Hasil Running



## Tugas2

### Tugas 2a

`import numpy as np` → Untuk Memanggil package numpy yang sebelumnya sudah pernah kita install

`import cv2 as cv` → Untuk Memanggil package opencv yang sebelumnya sudah pernah kita install

`cap = cv.VideoCapture("C:/PCD/latihan/yoi.mp4")` → Untuk memanggil file video yang sudah ada lalu kita simpan ke dalam variable cap

`while cap.isOpened():`

`ret, frame = cap.read()`

`# if frame is read correctly ret is True`

`if not ret:`

`print("Can't receive frame (stream end?). Exiting ...")`

`break`

`gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)`

`cv.imshow('frame', gray)`

`if cv.waitKey(111) == ord('j'):`

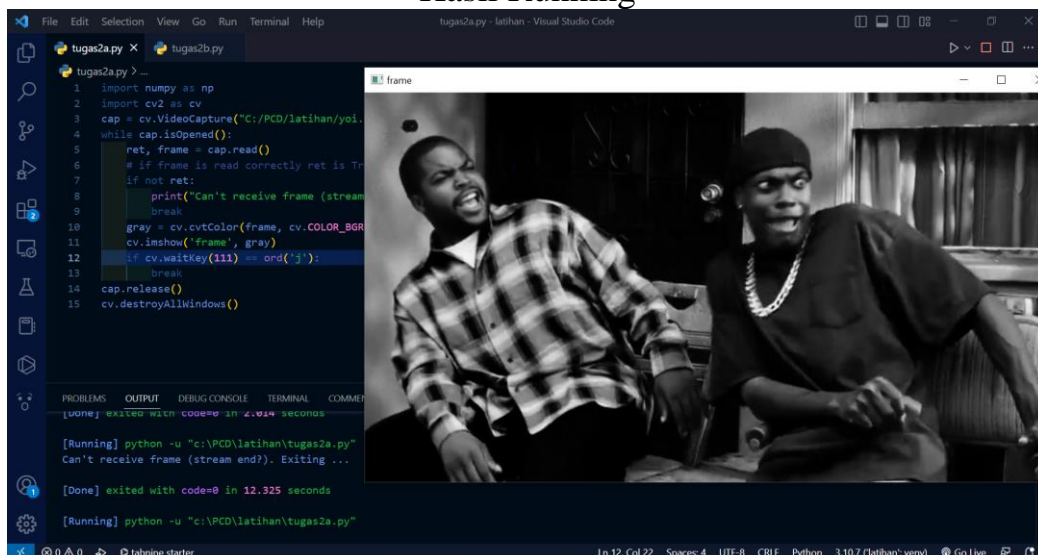
`break`

`cap.release()`

Program di atas digunakan untuk perulangan. Jadi selama video yang kita panggil tadi ada maka akan di tampilkan ke layar dengan warna videonya abu abu dan jika videonya itu tidak ada maka akan ada pesan "Can't receive frame (stream end?). Exiting ...")

`cv.destroyAllWindows()` → untuk mengilangkan video yang tampil di layar

### Hasil Running



## Tugas 2b

`fourcc = cv.VideoWriter_fourcc(*'XVID')` → Membuat library dan mendefinisikan codec yang akan digunakan untuk merekam video, dalam hal ini codec XVID yang disimpan pada variable `fourcc`.”

`out = cv.VideoWriter('C:/New folder(2), fourcc, 20.0, (640, 480))` → merekam video dalam file 'output.avi atau video ber extensi lain', dengan menggunakan codec XVID, frame rate 20 fps, dangan resolusi 640x480 yang disimpan pada variable `out`.

`while cap.isOpened():`

`ret, frame = cap.read()`

    if not ret:

`print("Can't receive frame (stream end?). Exiting ...")`

`break`

`frame = cv.flip(frame, 1)` → akan membuka kamera laptop dan jika nilainya kita kasih 0 maka kameranya akan berposisi terbalik

    ➔ Program di atas akan dilakukan perulangan selama fungsi `isOpened()` berjalan maka akan membaca yang simpan pada variable `ret` dengan isi Boolean yang nantinya akan disimpan pada variable `frame` dan jika tidak ada saat proses membaca berjalan maka akan tampil pesan “ Can't receive frame (stream end?). Exiting ...”) “

# write the flipped frame

`out.write(frame)`

`cv.imshow('frame', frame)`

    if `cv.waitKey(1) == ord('q')`: ➔ if `cv.waitKey(1) == ord('q')`: break “ Menunggu tombol ‘q’ ditekan dan loop berakhir dan video juga berakhir jika ditekan.

`break`

# Release everything if job is finished

`cap.release()`

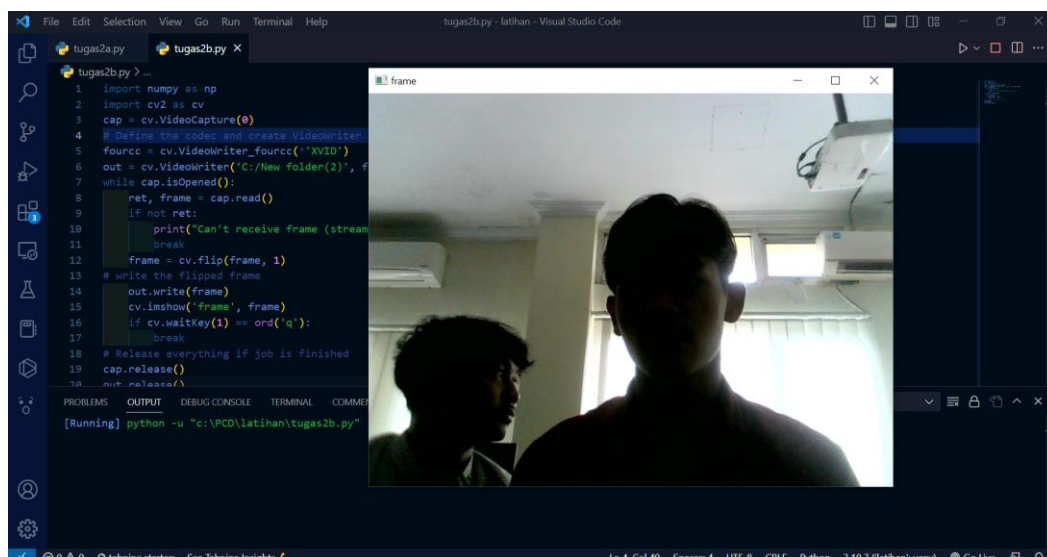
`out.release()`

➔ **cap.release()** “ digunakan untuk menghentikan stream pada webcam

`cv.destroyAllWindows()`

➔ `cv.destroyAllWindows()`: Menutup semua jendela OpenCV yang terbuka.

## Hasil Running



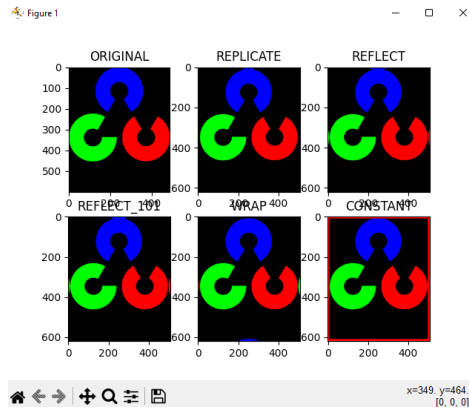
## Tugas3

### Tugas 3a

```
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt BLUE = [255,0,0]
img1 = cv.imread('img/openCV.png')
replicate = cv.copyMakeBorder(img1,10,10,10,10,cv.BORDER_REPLICATE)
reflect = cv.copyMakeBorder(img1,10,10,10,10,cv.BORDER_REFLECT)
reflect101 = cv.copyMakeBorder(img1,10,10,10,10,cv.BORDER_REFLECT_101)
wrap = cv.copyMakeBorder(img1,10,10,10,10,cv.BORDER_WRAP)
constant=
cv.copyMakeBorder(img1,10,10,10,10,cv.BORDER_CONSTANT,value=BLUE)
plt.subplot(231),plt.imshow(img1,'gray'),plt.title('ORIGINAL')
plt.subplot(232),plt.imshow(replicate,'gray'),plt.title('REPLICATE')
plt.subplot(233),plt.imshow(reflect,'gray'),plt.title('REFLECT')
plt.subplot(234),plt.imshow(reflect101,'gray'),plt.title('REFLECT_101')
plt.subplot(235),plt.imshow(wrap,'gray'),plt.title('WRAP')
plt.subplot(236),plt.imshow(constant,'gray'),plt.title('CONSTANT')
plt.show()
```

- Import library OpenCV dan NumPy untuk pengolahan citra dan manipulasi array.
- Import pyplot dari library matplotlib untuk menampilkan gambar.
- Mendefinisikan warna biru dengan RGB (255,0,0).
- Membaca gambar dengan nama file 'openCV.png' menggunakan fungsi imread dari OpenCV.
- Menambahkan tepi pada gambar menggunakan fungsi copyMakeBorder dari OpenCV dengan parameter (img1,10,10,10,10,cv.BORDER\_REPLICATE).
- Menambahkan tepi pada gambar menggunakan fungsi copyMakeBorder dari OpenCV dengan parameter (img1,10,10,10,10,cv.BORDER\_REFLECT).
- Menambahkan tepi pada gambar menggunakan fungsi copyMakeBorder dari OpenCV dengan parameter (img1,10,10,10,10,cv.BORDER\_REFLECT\_101).
- Menambahkan tepi pada gambar menggunakan fungsi copyMakeBorder dari OpenCV dengan parameter (img1,10,10,10,10,cv.BORDER\_WRAP).
- Menambahkan tepi pada gambar menggunakan fungsi copyMakeBorder dari OpenCV dengan parameter (img1,10,10,10,10,cv.BORDER\_CONSTANT,value=BLUE).
- Menampilkan beberapa gambar pada satu frame menggunakan pyplot dari matplotlib.
- Menampilkan gambar pertama pada grid 2x3 pada posisi 1.
- Menampilkan gambar kedua pada grid 2x3 pada posisi 2.

- Menampilkan gambar ketiga pada grid 2x3 pada posisi 3.
- Menampilkan gambar keempat pada grid 2x3 pada posisi 4.
- Menampilkan gambar kelima pada grid 2x3 pada posisi 5.
- Menampilkan gambar keenam pada grid 2x3 pada posisi 6.
- Menampilkan gambar pada jendela dengan fungsi



show() dari pyplot. Hasil Run :

### Tugas 3b

```
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt
x = np.uint8([250])
y = np.uint8([10])
print(cv.add(x, y)) # 250+10 = 260 => 255

print(x+y) # 250+10 = 260 % 256 = 4
```

- impor cv2 sebagai cv: impor library OpenCV dengan alias 'cv'.
  - impor numpy sebagai np: impor library NumPy dengan alias 'np'.
  - from matplotlib import pyplot as plt: impor modul pyplot dari library Matplotlib dengan alias 'plt'.
  - x = np.uint8([250]): buat array NumPy 'x' bertipe uint8 dengan nilai 250.
  - y = np.uint8([10]): buat array NumPy 'y' bertipe uint8 dengan nilai 10.
  - print(cv.add(x,y)): tambahkan array 'x' dan 'y' menggunakan fungsi cv2.add() dan cetak hasilnya.
  - print(x+y): tambahkan array 'x' dan 'y' menggunakan operator penjumlahan NumPy dan cetak hasilnya
- Output :

```
[Running] python -u "c:\PCD\latihan\tugas4\tugas2a2.py"
[[255]]
[4]
```

### Tugas 3c

```
import cv2 as cv import numpy as np
from matplotlib import pyplot as plt x = np.uint8([250])
y = np.uint8([10])
print(cv.add(x, y)) # 250+10 = 260 => 255
```

```
print(x+y) # 250+10 = 260 % 256 = 4
```

- import cv2 as cv: memuat library OpenCV dengan alias 'cv'.
- import numpy as np: memuat library NumPy dengan alias 'np'.
- img1 = cv.imread('img/smile.png'): membaca gambar 'smile.png' dan menyimpannya ke dalam variabel 'img1'.
- img2 = cv.imread('img/pixelart.jpg'): membaca gambar 'pixelart.jpg' dan menyimpannya ke dalam variabel 'img2'.
- dst = cv.addWeighted(img1,0.7,img2,0.3,0): melakukan operasi blending pada dua gambar 'img1' dan 'img2' dengan koefisien alpha 0.7 dan 0.3, serta gamma 0. Hasil dari operasi blending tersebut disimpan dalam variabel 'dst'.
- cv.imshow('dst',dst): menampilkan gambar hasil blending yang disimpan dalam variabel 'dst' dengan judul 'dst' pada jendela pop-up.
- cv.waitKey(0): menunggu tombol keyboard ditekan untuk menutup jendela pop-up. Jika nilai parameter adalah  $n > 0$ , maka jendela akan tertutup setelah  $n$  milidetik.
- cv.destroyAllWindows(): menutup semua jendela pop-up yang dibuka.

Hasik Running



### Tugas 3d

```
import cv2 as cv import numpy as np
# Load two images
img1 = cv.imread('img/smile.png') img2 = cv.imread('img/pixelart.jpg')
# I want to put logo on top-left corner, So I create a ROI rows, cols, channels =
img2.shape
roi = img1[0:rows, 0:cols]
# Now create a mask of logo and create its inverse mask also img2gray =
cv.cvtColor(img2, cv.COLOR_BGR2GRAY)
ret, mask = cv.threshold(img2gray, 10, 255, cv.THRESH_BINARY) mask_inv =
cv.bitwise_not(mask)
# Now black-out the area of logo in ROI
img1_bg = cv.bitwise_and(roi, roi, mask=mask_inv)
# Take only region of logo from logo image. img2_fg = cv.bitwise_and(img2, img2,
mask=mask)
# Put logo in ROI and modify the main image dst = cv.add(img1_bg, img2_fg)
img1[0:rows, 0:cols] = dst
cv.imshow('res', img1)
cv.waitKey(0)
cv.destroyAllWindows()
```



- `import cv2 as cv` : mengimport library OpenCV dan memberikan alias 'cv'.
- `import numpy as np` : mengimport library NumPy dan memberikan alias 'np'.
- `img1 = cv.imread('img/smile.png')` : membaca file gambar 'smile.png' menggunakan `cv2.imread()` dan menyimpannya di variabel 'img1'.
- `img2 = cv.imread('img/pixelart.jpg')` : membaca file gambar 'pixelart.jpg' menggunakan `cv2.imread()` dan menyimpannya di variabel 'img2'.
- `rows, cols, channels = img2.shape` : mendapatkan dimensi gambar 'img2' dan menyimpannya di variabel 'rows', 'cols', dan 'channels'.
- `roi = img1[0:rows, 0:cols]` : mengambil bagian gambar 'img1' sesuai dengan ukuran gambar 'img2' dan menyimpannya di variabel 'roi' sebagai region of interest (ROI).
- `img2gray = cv.cvtColor(img2, cv.COLOR_BGR2GRAY)` : mengubah gambar 'img2' dari warna BGR menjadi grayscale menggunakan `cv2.cvtColor()` dan menyimpannya di variabel 'img2gray'.
- `ret, mask = cv.threshold(img2gray, 10, 255, cv.THRESH_BINARY)` : menghasilkan threshold binary menggunakan `cv2.threshold()` pada gambar 'img2gray' dengan nilai threshold 10 dan menyimpannya di variabel 'mask'.
- `mask_inv = cv.bitwise_not(mask)` : menghasilkan inverse mask dari 'mask' menggunakan `cv2.bitwise_not()` dan menyimpannya di variabel 'mask\_inv'.
- `img1_bg = cv.bitwise_and(roi, roi, mask=mask_inv)` : menggabungkan gambar ROI 'roi' dan inverse mask 'mask\_inv' menggunakan `cv2.bitwise_and()` dan menyimpannya di variabel 'img1\_bg'.
- `img2_fg = cv.bitwise_and(img2, img2, mask=mask)` : menggabungkan gambar 'img2' dan mask 'mask' menggunakan `cv2.bitwise_and()` dan menyimpannya di variabel 'img2\_fg'.
- `dst = cv.add(img1_bg, img2_fg)` : menggabungkan gambar 'img1\_bg' dan 'img2\_fg' menggunakan `cv2.add()` dan menyimpannya di variabel 'dst'.
- `img1[0:rows, 0:cols] = dst` : menempatkan hasil gabungan 'dst' ke dalam ROI 'img1' dan menyimpannya kembali ke variabel 'img1'.
- `cv.imshow('res', img1)` : menampilkan hasil gabungan ke dalam jendela OpenCV dengan judul 'res'.
- `cv.waitKey(0)` : menunggu input keyboard dari pengguna.
- `cv.destroyAllWindows()` : menutup semua jendela OpenCV yang terbuka.

### Hasil Running



### Tugas 3e

```
import cv2 as cv import numpy as np
e1 = cv.getTickCount() # your code execution e2 = cv.getTickCount()
time = (e2 - e1)/ cv.getTickFrequency()
img1 = cv.imread('img/spiderman.jpg') e1 = cv.getTickCount()
for i in range(5,49,2):
    img1 = cv.medianBlur(img1,i)
    e2 = cv.getTickCount()
    t = (e2 - e1)/cv.getTickFrequency()
print( t )
```

- Import library cv2 as cv dan library numpy as np.
- Menggunakan fungsi cv.getTickCount() untuk mendapatkan waktu saat ini dalam siklus clock CPU dan menyimpannya dalam variabel e1.
- Tidak ada kode yang dieksekusi di sini. Komentar "your code execution" hanya sebagai placeholder untuk menunjukkan bahwa ini adalah tempat di mana kode yang ingin diukur waktu eksekusinya harus ditempatkan.
- Menggunakan kembali fungsi cv.getTickCount() untuk mendapatkan waktu saat ini dalam siklus clock CPU setelah kode yang ingin diukur waktu eksekusinya selesai dieksekusi dan menyimpannya dalam variabel e2.
- Menghitung selisih waktu antara variabel e2 dan variabel e1, kemudian membaginya dengan frekuensi clock CPU menggunakan fungsi cv.getTickFrequency() untuk mendapatkan waktu eksekusi dalam detik, dan menyimpan hasilnya dalam variabel time.
- Menggunakan fungsi cv.imread() untuk membaca gambar 'img/spiderman.jpg' dan menyimpannya dalam variabel img1.
- Menggunakan kembali fungsi cv.getTickCount() untuk mendapatkan waktu saat ini dalam siklus clock CPU sebelum melakukan loop for.
- Melakukan loop for dengan variabel i yang berisi bilangan ganjil antara 5 dan 49 (termasuk 5 dan 49) dengan kenaikan nilai sebesar 2. Setiap iterasi loop, gambar img1 akan dikenai operasi median blur dengan kernel size sebesar i menggunakan fungsi cv.medianBlur().
- Menggunakan kembali fungsi cv.getTickCount() untuk mendapatkan waktu saat ini dalam siklus clock CPU setelah loop for selesai dieksekusi dan menyimpannya dalam variabel e2.
- Menghitung selisih waktu antara variabel e2 dan variabel e1, kemudian membaginya dengan frekuensi clock CPU menggunakan fungsi cv.getTickFrequency() untuk mendapatkan waktu eksekusi dalam detik, dan menyimpan hasilnya dalam variabel t.
- Mencetak nilai variabel t yang merupakan waktu eksekusi dari

```
[Running] python -u "c:\PCD\latihan\tugas4\tugas2c1.py"
29.1736574
```

loop for dalam detik. Output :

### Tugas 3f

```
import cv2 as cv
import numpy as np
import time

# membaca gambar
img1 = cv.imread('img/spiderman.jpg')
# mengambil waktu sebelum proses start_time = time.time()
# melakukan operasi pada gambar
img1 = cv.imread('img/spiderman.jpg')
for i in range(5,49,2):
    img1 = cv.medianBlur(img1,i)
# mengambil waktu setelah proses end_time = time.time()
#mencetak waktu eksekusi
print ("Waktu eksekusi: ", end_time - start_time)
```

- Import library cv2 as cv dan library numpy as np.
- Import library time.
- Menggunakan function cv.imread() untuk membaca gambar dengan path 'img/spiderman.jpg' dan menyimpannya dalam variabel img1.
- Menggunakan fungsi time.time() untuk mengambil waktu sekarang sebelum proses berlangsung dan menyimpannya dalam variabel start\_time.
- Kembali membaca gambar 'img/spiderman.jpg' dan menyimpannya dalam variabel img1.
- Melakukan pengolahan gambar dengan memanggil fungsi cv.medianBlur() dengan parameter img1 dan i, dimana i adalah bilangan ganjil yang nilainya berubah dari 5 hingga 49 dengan increment 2 pada setiap iterasi. Operasi ini dilakukan untuk menghilangkan noise pada gambar.
- Menggunakan kembali fungsi time.time() untuk mengambil waktu sekarang setelah proses berlangsung dan menyimpannya dalam variabel end\_time.
- Mencetak waktu eksekusi dengan menghitung selisih waktu end\_time dan start\_time dan menyimpannya dalam variabel end\_time - start\_time.
- Sehingga pada perbarisannya program ini melakukan operasi pengolahan gambar menggunakan fungsi medianBlur() untuk menghilangkan noise pada gambar, kemudian mencetak waktu eksekusi untuk menunjukkan seberapa cepat atau lambat program tersebut bekerja.

### Hasil Running

```
[Running] python -u "c:\PCD\latihan\tugas4\tugas2c2.py"
Waktu eksekusi: 29.023895502090454
```

### Tugas 3g

```
import numpy as np
import cv2 as cv
cv.useOptimized()
True
img = cv.imread('img/pixelart.jpg')
res = cv.medianBlur(img, 49)
print (res) cv.setUseOptimized(False)
cv.useOptimized()
False
res = cv.medianBlur(img,49) print(res)
# cv.imshow("Display window", img) # if cv.waitKey(0) & 0xff == 27:
# cv.destroyAllWindows()
```

- Import library cv2 as cv dan library numpy as np.
- Menggunakan fungsi cv.useOptimized() untuk mengecek apakah mode optimasi sudah diaktifkan atau belum. Fungsi ini mengembalikan nilai True jika mode optimasi sudah aktif.
- Membaca gambar 'img/pixelart.jpg' menggunakan fungsi cv.imread() dan menyimpannya dalam variabel img.
- Menggunakan fungsi cv.medianBlur() untuk melakukan operasi median blur pada gambar img dengan kernel size sebesar 49 dan menyimpan hasilnya dalam variabel res.
- Mencetak hasil operasi median blur pada gambar img yang disimpan dalam variabel res.
- Menggunakan fungsi cv.setUseOptimized(False) untuk menonaktifkan mode optimasi.
- Menggunakan kembali fungsi cv.useOptimized() untuk mengecek apakah mode optimasi sudah dinonaktifkan atau belum. Fungsi ini mengembalikan nilai False jika mode optimasi sudah tidak aktif.
- Menggunakan kembali fungsi cv.medianBlur() untuk melakukan operasi median blur pada gambar img dengan kernel size sebesar 49 dan menyimpan hasilnya dalam variabel res.
- Mencetak hasil operasi median blur pada gambar img yang disimpan dalam variabel res setelah mode optimasi dimatikan.
- Komentar pada baris ke-11 dan 12 menunjukkan bahwa program ini seharusnya menampilkan gambar dengan menggunakan fungsi cv.imshow() dan menunggu input dari pengguna dengan menggunakan fungsi cv.waitKey(). Jika pengguna menekan tombol esc (ASCII code 27), maka jendela gambar akan ditutup. Namun, kedua baris tersebut telah dijadikan komentar dan tidak dieksekusi. Sehingga program hanya melakukan pengolahan gambar dan mencetak hasilnya saja.

## Hasil Running



	...	...	...
[Running] python		[ 19 5 8]	
[[[141 38 141]		[ 20 5 8]	[[ 5 1 3]
[141 38 141]	[[ 5 1 3]	[ 20 5 8]]	[ 5 1 3]
[141 38 141]	[ 5 1 3]	[[[141 38 141]	[ 5 1 3]
...	[ 5 1 3]	[141 38 141]	...
[ 2 1 1]	...	[141 38 141]	[ 19 5 8]
[ 1 1 1]	[ 19 5 8]	...	[ 19 5 8]
[ 1 1 1]]	[ 19 5 8]	[ 2 1 1]	[ 20 5 8]]
	[ 20 5 8]]	[ 1 1 1]	
[[[141 38 141]		[[[141 38 141]	[[ 5 1 3]
[141 38 141]		[141 38 141]	[ 5 1 3]
[141 38 141]	[[ 5 1 3]	[141 38 141]	[ 5 1 3]
...	[ 5 1 3]	...	...
[ 2 1 1]	[ 5 1 3]	[ 2 1 1]	[ 19 5 8]
[ 1 1 1]	[ 5 1 3]	[ 1 1 1]	[ 19 5 8]
[ 1 1 1]]	...	[ 1 1 1]]	[ 20 5 8]]
	[ 19 5 8]		
[[[141 38 141]	[ 19 5 8]	[[[141 38 141]	[[ 5 1 3]
[141 38 141]	[ 20 5 8]]	[141 38 141]	[ 5 1 3]
[141 38 141]		[141 38 141]	[ 5 1 3]
...		...	...
[ 2 1 1]	[[ 5 1 3]	[ 2 1 1]	[ 19 5 8]
[ 1 1 1]	[ 5 1 3]	[ 1 1 1]	[ 20 5 8]
[ 1 1 1]]	[ 5 1 3]	[ 1 1 1]]	[ 20 5 8]]

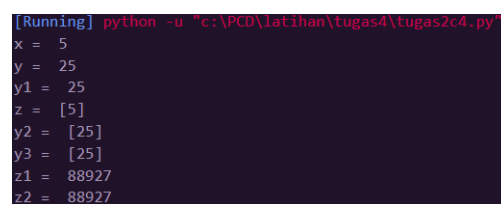
### Tugas 3h

```
import cv2 as cv import numpy as np
img = cv.imread('img/pixelart.jpg', 1)
img = cv.cvtColor(img, cv.COLOR_RGB2GRAY) type(img)
x = 5
y = x**2 y1 = x*x
z = np.uint8([5]) y2 = z*z
y3 = np.square(z)
z1 = cv.countNonZero(img)
z2 = np.count_nonzero(img)
```

- Import library cv2 as cv dan library numpy as np.
- Menggunakan fungsi cv.imread() untuk membaca gambar 'img/pixelart.jpg' dalam mode 1 (membaca gambar dengan mode aslinya) dan menyimpannya dalam variabel img.
- Menggunakan fungsi cv.cvtColor() untuk mengubah gambar dari mode RGB menjadi grayscale menggunakan konstanta cv.COLOR\_RGB2GRAY dan menyimpannya kembali ke dalam variabel img.
- Menggunakan fungsi bawaan Python type() untuk mengetahui tipe data dari variabel img dan mencetak hasilnya.
- Memberikan nilai 5 ke variabel x.
- Menghitung nilai kuadrat dari variabel x menggunakan operator \*\* dan menyimpan hasilnya ke dalam variabel y.
- Menghitung nilai kuadrat dari variabel x menggunakan operator \* dan menyimpan hasilnya ke dalam variabel y1.
- Membuat array numpy dengan tipe data uint8 berisi satu nilai 5 dan menyimpannya ke dalam variabel z.
- Mengalikan array z dengan dirinya sendiri menggunakan operator \* dan menyimpan hasilnya ke dalam variabel y2.
- Menggunakan fungsi numpy square() untuk menghitung kuadrat dari setiap elemen dalam array z dan menyimpan hasilnya ke dalam variabel y3.
- Menggunakan fungsi cv.countNonZero() untuk menghitung jumlah piksel yang bukan nol dalam gambar img dan menyimpan hasilnya ke dalam variabel z1.
- Menggunakan fungsi numpy count\_nonzero() untuk menghitung jumlah elemen yang bukan nol dalam array img dan menyimpan hasilnya ke dalam variabel z2.

Mencetak nilai variabel x, y, y1, z, y2, y3, z1, dan z2

### Hasil Running



```
[Running] python -u "c:\PCD\latihan\tugas4\tugas2c4.py"
x = 5
y = 25
y1 = 25
z = [5]
y2 = [25]
y3 = [25]
z1 = 88927
z2 = 88927
```

## Tugas 4

### Tugas 4a

```
import numpy as np
import cv2 as cv
img = cv.imread('img/smile.png')
res = cv.resize(img, None, fx=2, fy=2, interpolation = cv.INTER_CUBIC) #
cv.imshow('res', res)
#OR
height, width = img.shape[:2]
res = cv.resize(img, (2*width, 2*height), interpolation = cv.INTER_CUBIC)
cv.imshow('res', res)
cv.waitKey(0)
```

Baris pertama mengimpor 2 package

Baris kedua membaca gambar smile.png

menggunakan cv.imread Baris ketiga mengubah

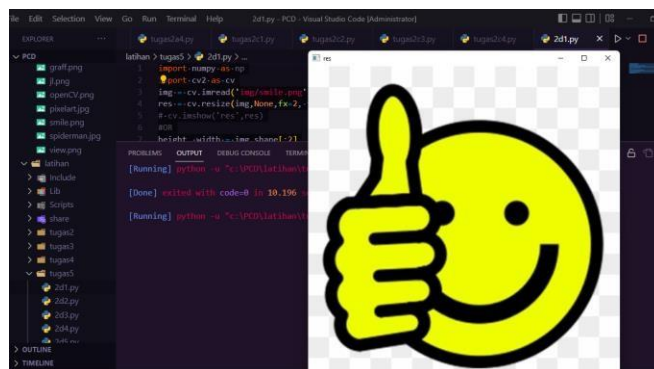
ukuran gambar menggunakan cv.resize

Baris keempat dan kelima melakukan hal yg sama dengan baris ketiga, namun menggunakan metode yang berbeda. Ukuran gambar 2x ukuran gambar asli

Baris ke enam menampilkan gambar yang telah di resize menggunakan cv.imshow

Bris terakhir menggunakan cv.waitKey untuk menunggu user menekan tombol pada keyboard untuk keluar program

### Hasil Running



### Tugas 4b

```
import numpy as np
import cv2 as cv
img = cv.imread('img/smile.png', 0)
rows, cols = img.shape
M = np.float32([[1, 0, 100], [0, 1, 50]])
dst = cv.warpAffine(img, M, (cols, rows))
cv.imshow('img', dst)
cv.waitKey(0)
cv.destroyAllWindows()
```

Baris pertama mengimpor 2 package

Baris kedua membaca gambar smile.png dan mengubah citra menjadi grayscale dengan menambahkan param 0  
Baris ketiga dan keempat mendapatkan dimensi citra menggunakan shape

Baris kelima menciptakan sebuah matrix mempresentasikan transformasi affine  
Baris keenam menggunakan transformasi affine pada citra

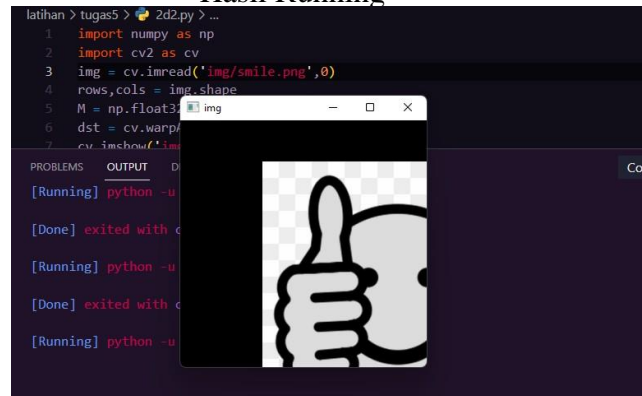
Baris ketujuh menampilkan hasil

menggunakan cv.imshow

Baris kedelapan menunggu ser menekan tombol di keyboard

Baris terakhir menggunakan cv.destroyAllWindows untuk menghapus semua jendela

### Hasil Running





#### Tugas 4c

```
import numpy as np
import cv2 as cv
img = cv.imread('img/smile.png',0)
rows,cols = img.shape
# cols-1 and rows-1 are the coordinate limits.
M = cv.getRotationMatrix2D(((cols-1)/2.0,(rows-1)/2.0),90,1)
dst = cv.warpAffine(img,M,(cols,rows))
print(dst) cv.imshow('dst', dst) cv.waitKey(0)
```

Baris pertama mengimpor 2 package  
Baris kedua membaca gambar smile.png dan diubah menjadi grayscale dengan param0  
Baris ketiga dan keempat mendapatkan dimensi citra menggunakan shape  
Baris kelima menciptakan sebuah matrix 2x3 M yang merepresentasikan transformasi affine  
Baris keenam menggunakan cv.warpAffine untuk menerapkan transformasi pada citra  
Baris ketujuh menampilkan hasil  
Baris kedelapan menunggu keyboard ditekan  
Baris ke Sembilan mencetak hasil transformasi yang

disimpan pada-variable dst. Hasil :



#### Tugas 4d

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

img = cv.imread('C:\PCD\pcd4\Spiderman1.jpg')
rows, cols, ch = img.shape

pts1 = np.float32([[50, 50], [200, 50], [50, 200]])
pts2 = np.float32([[10, 100], [200, 50], [100, 250]])

M = cv.getAffineTransform(pts1, pts2)
dst = cv.warpAffine(img, M, (cols, rows))

plt.subplot(121), plt.imshow(cv.cvtColor(img, cv.COLOR_BGR2RGB)), plt.title('Input')
plt.subplot(122), plt.imshow(cv.cvtColor(dst, cv.COLOR_BGR2RGB)),
```

```
plt.title('Output')
```

```
plt.show()
```

Baris pertama mengimpor 3 package

Baris kedua membaca gambar

Baris ketiga sampai kelima mendapatkan dimensi citra

menggunakan shape

Baris keenam dan ketujuh membuat

2 buah matrix 2x3, pts1 dan pts2

Baris kedelapan menggunakan cv.getAffineTransform untuk mendapatkan matriks

transformasi Affine M

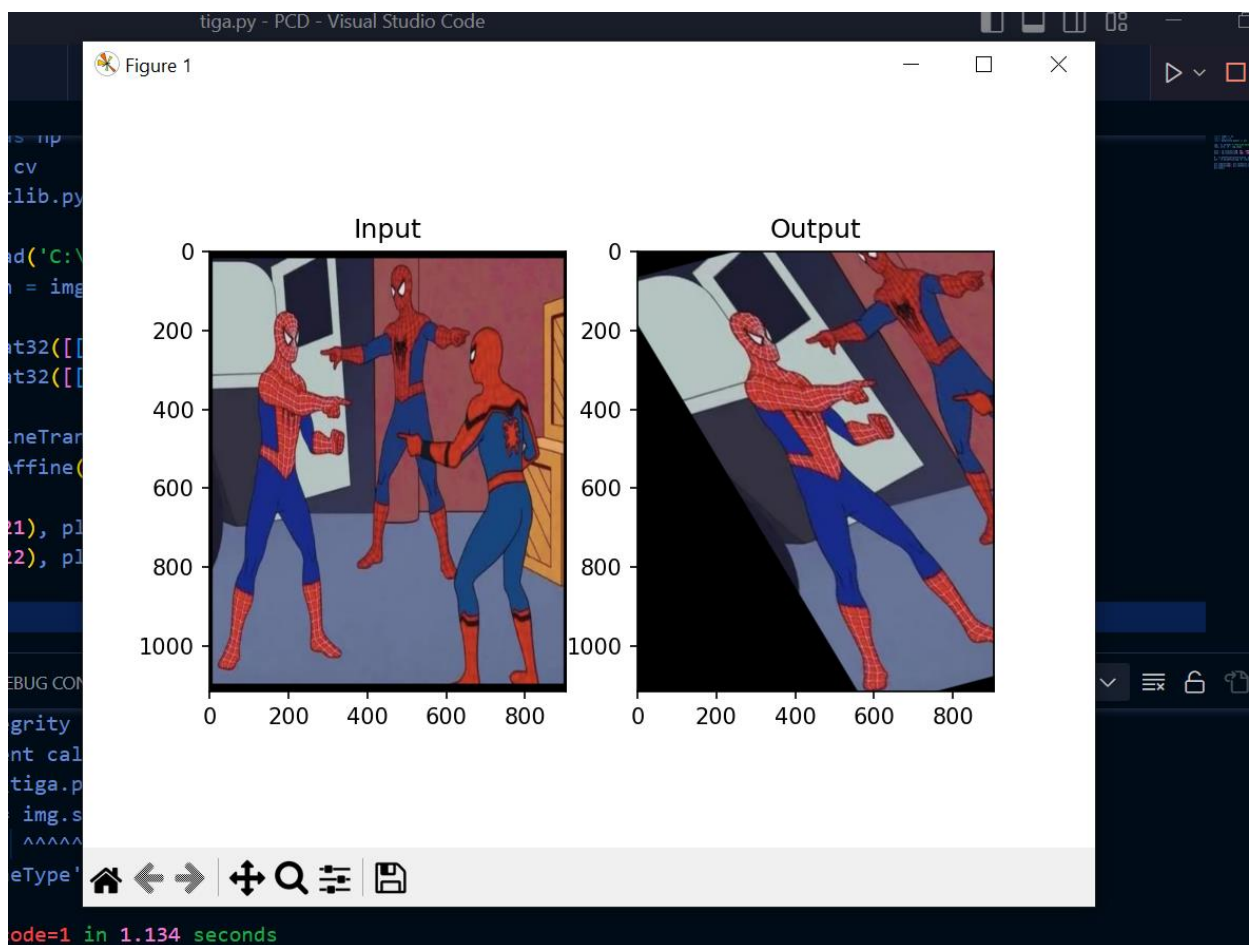
Baris kesembilan menggunakan cv.warpAffine untuk

menerapkan transformasi Affine

Baris kesepuluh sampai ke dua belas menggunakan pyplot untuk menampilkan

citra asli dan citra hasil transformasi dalam satu window

Hasil running



## Tugas 4e

```
import numpy as np
```

```
import cv2 as cv
```

```
from matplotlib import pyplot as plt
```

```
img = cv.imread('C:\PCD\pcd4\Spiderman1.jpg')
```

```
rows,cols,ch = img.shape
```

```
pts1 = np.float32([[56,65],[368,52],[28,387],[389,390]])
```

```
pts2 = np.float32([[0,0],[300,0],[0,300],[300,300]])
```

```
M = cv.getPerspectiveTransform(pts1,pts2)
```

```
dst = cv.warpPerspective(img,M,(300,300))
```

```
plt.subplot(121),plt.imshow(img),plt.title('Input')
```

```
plt.subplot(122),plt.imshow(dst),plt.title('Output')
```

plt.show()

Baris pertama

mengimpor library

kedua membaca gambar

Baris ketiga mendapatkan dimensi baris, kolom dan channel dari gambar

mengunakan `img.shape`

Baris keempat menentukan 4 titik sudut pada gambar sumber

Menggunakan `cv.getPerspectiveTransform` untuk mendapatkan matriks transformasi perspektif yang diperlukan untuk memetakan titik sudut pada gambar

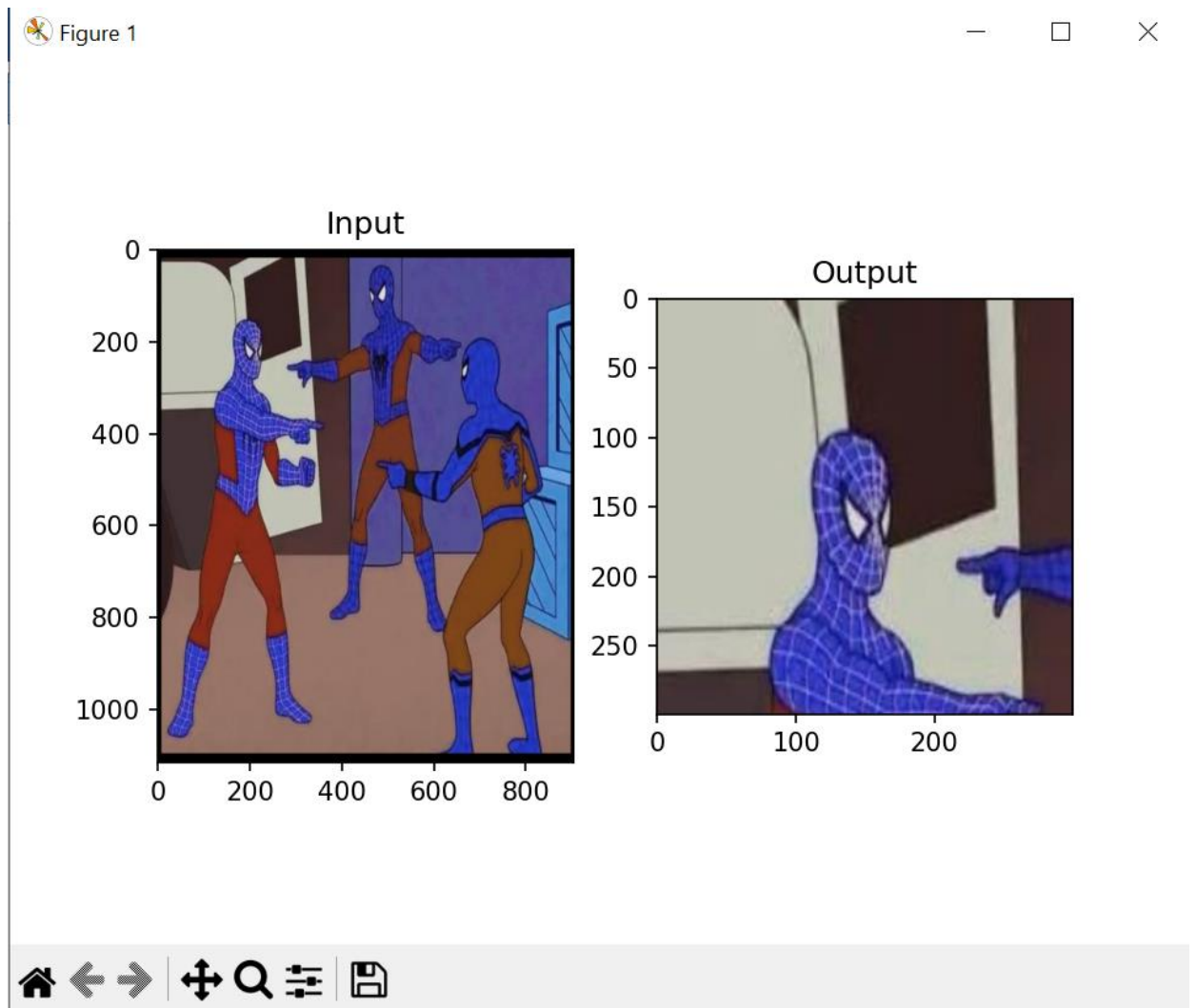
Menggunakan `cv.warpPerspective` untuk memetakan gambar sumber ke gambar tujuan

Menampilkan gambar asli dan gambar hasil pemetaan

menggunakan `pyplot.subplot` dan `imshow`

Menampilkan plot yang telah dibuat menggunakan `pyplot.imshow`

Hasil :



## Tugas

### Tugas 3a

`import numpy as np` - Mengimpor library numpy dan memberikan alias np.

`import cv2 as cv` - Mengimpor library OpenCV dan memberikan alias cv.

`filename = 'catur.jpeg'` - Mendefinisikan variabel filename dengan nama file gambar yang akan diproses.

`img = cv.imread(filename)` - Membaca gambar dari file yang ditentukan di filename menggunakan fungsi `imread` dari OpenCV.

`gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)` - Mengubah gambar warna img menjadi grayscale menggunakan fungsi `cvtColor` dari OpenCV.

`gray = np.float32(gray)` - Mengubah gambar grayscale gray ke format floating point menggunakan fungsi `float32` dari numpy.

`dst = cv.cornerHarris(gray, 2, 3, 0.04)` - Mendeteksi sudut-sudut pada gambar menggunakan algoritma Harris Corner Detection dan fungsi `cornerHarris` dari OpenCV. Parameter 2, 3, dan 0.04 adalah ukuran tetangga yang dipertimbangkan untuk deteksi sudut, ukuran kernel Sobel yang digunakan untuk menghitung turunan, dan parameter bebas detektor Harris, secara berturut-turut.

`dst = cv.dilate(dst, None)` - Membesarkan titik sudut pada gambar menggunakan fungsi `dilate` dari OpenCV.

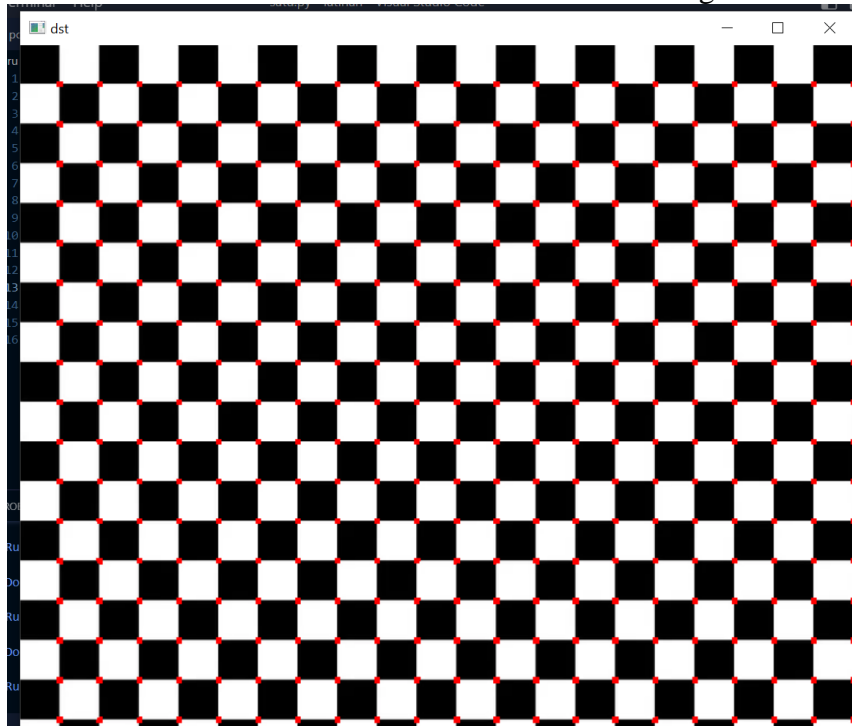
`img[dst > 0.01 * dst.max()] = [0, 0, 255]` - Menandai sudut-sudut yang terdeteksi pada gambar dengan mengubah warna piksel menjadi merah. Sudut-sudut dideteksi berdasarkan ambang 0.01 kali nilai maksimum dari dst.

`cv.imshow('dst', img)` - Menampilkan gambar akhir dengan sudut-sudut yang terdeteksi menggunakan fungsi `imshow` dari OpenCV.

`if cv.waitKey(0) & 0xff == 27:` - Menunggu suatu event keyboard dan memeriksa apakah itu adalah tombol 'Esc' (kode ASCII 27).

`cv.destroyAllWindows()` - Menutup semua jendela OpenCV yang terbuka selama eksekusi program.

Hasil running



### Tugas 3b

import numpy as np

import cv2 as cv -> Kode di atas mengimpor dua modul yaitu NumPy dan OpenCV.

filename = 'catur.jpeg'

img = cv.imread(filename) -> membaca gambar 'chessboard2.jpg' dan menyimpannya dalam variabel img.

gray = cv.cvtColor(img, cv.COLOR\_BGR2GRAY)->mengonversi gambar berwarna menjadi gambar grayscale (grayscale adalah gambar dengan tingkat keabuan atau hanya memiliki satu channel warna). Konversi ini dilakukan dengan menggunakan fungsi cvtColor() dari modul OpenCV. Dalam kasus ini, cvtColor() digunakan untuk mengubah gambar img dari mode warna BGR (Blue, Green, Red) menjadi mode grayscale.

gray = np.float32(gray)->mengubah tipe data dari gambar grayscale menjadi tipe data float32 menggunakan fungsi float32() dari NumPy.

dst = cv.cornerHarris(gray, 2, 3, 0.04) -> menghitung sudut-sudut sudut Harris pada gambar grayscale menggunakan fungsi cornerHarris() dari OpenCV. Parameter pertama gray adalah gambar grayscale yang telah dikonversi sebelumnya, parameter kedua dan ketiga adalah ukuran kernel sobel untuk menghitung gradien gambar, dan parameter terakhir adalah konstanta sudut Harris

dst = cv.dilate(dst, None) -> melakukan operasi dilasi pada gambar sudut Harris menggunakan fungsi dilate() dari OpenCV. Tujuan dilasinya gambar adalah untuk memperkuat nilai sudut pada gambar.

ret, dst = cv.threshold(dst, 0.01\*dst.max(), 255, 0) -> melakukan thresholding pada gambar sudut Harris dengan nilai ambang sebesar 1% dari nilai maksimum sudut Harris menggunakan fungsi threshold() dari OpenCV.

dst = np.uint8(dst) ->mengubah tipe data gambar sudut Harris menjadi tipe data uint8 (8-bit unsigned integer) menggunakan fungsi uint8() dari NumPy.

ret, labels, stats, centroids = cv.connectedComponentsWithStats(dst) ->menghitung komponen-komponen terhubung dari gambar sudut Harris menggunakan fungsi connectedComponentsWithStats() dari OpenCV. Hasil dari fungsi ini adalah jumlah komponen terhubung, label-label komponen, statistik statistik komponen, dan koordinat-koordinat centroid dari setiap komponen.

criteria = (cv.TERM\_CRITERIA\_EPS + cv.TERM\_CRITERIA\_MAX\_ITER, 100, 0.001)

corners = cv.cornerSubPix(gray, np.float32(centroids), (5,5), (-1,-1), criteria) ->melakukan sub-pixel corner refinement pada setiap sudut sudut sudut Harris menggunakan fungsi cornerSubPix() dari OpenCV. Fungsi ini mengambil gambar grayscale, posisi-posisi sudut Harris yang telah dihitung sebelumnya, ukuran kernel sobel, posisi

for i in range(len(corners)):

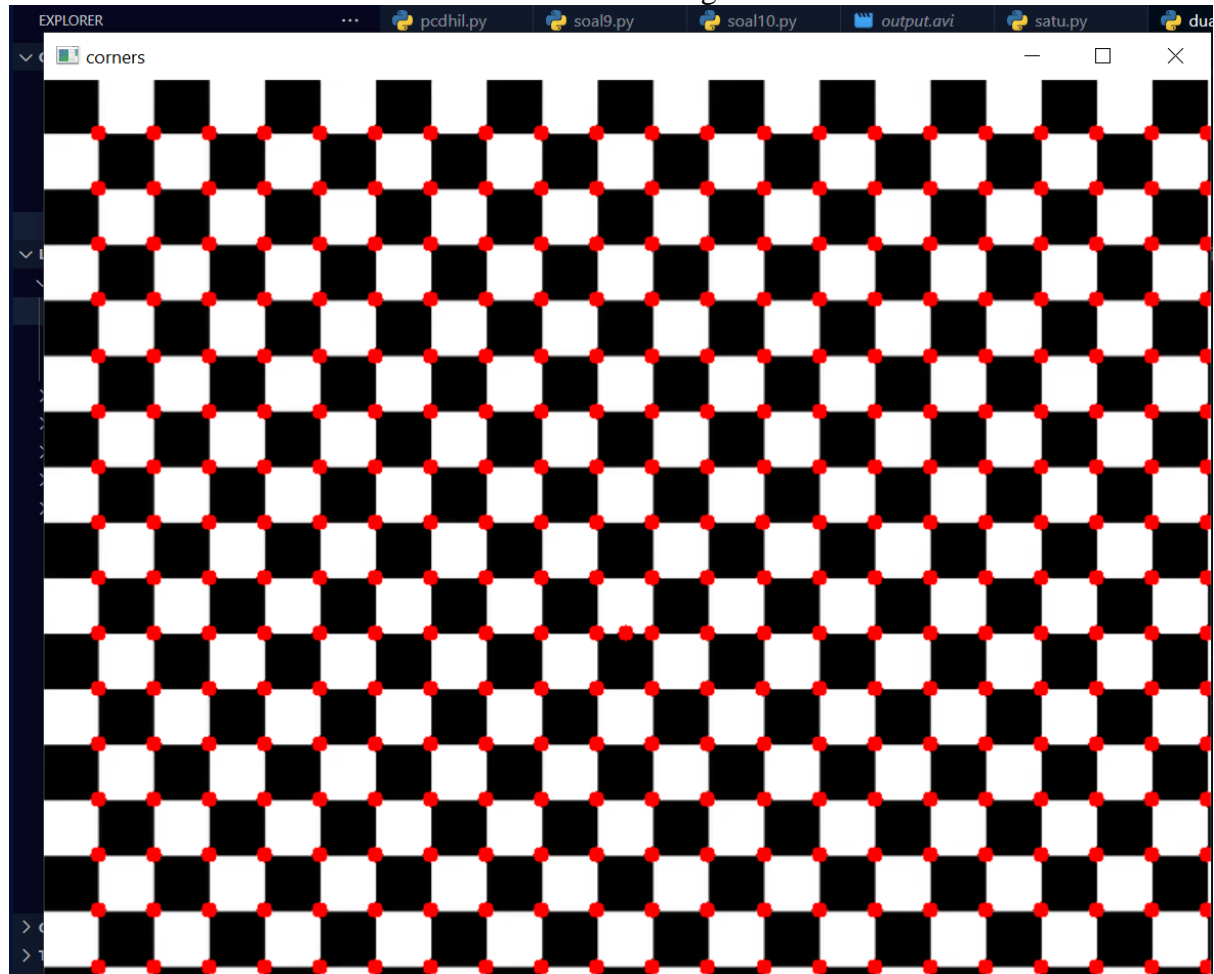
    x, y = corners[i]

    cv.circle(img, (int(x), int(y)), 5, (0, 0, 255), -1) ->Menggambar titik sudut pada gambar menggunakan fungsi cv.circle() dari OpenCV dan mengubah warna titik sudut menjadi merah.  
cv.imshow('corners', img)

if cv.waitKey(0) & 0xff == 27:

    cv.destroyAllWindows() ->Menampilkan gambar hasil pada jendela pop-up menggunakan fungsi cv.imshow() dari OpenCV dan menunggu input keyboard. Jika tombol escape (kode ASCII 27) ditekan, maka jendela akan ditutup menggunakan fungsi cv.destroyAllWindows() dari OpenCV.

## Hasil running



### C. Tugas UTS

Cari dan ambil salah satu image atau citra dan beri nama photo1.jpg. Jelaskan masing-masing baris script padaprogram berikut, dan untuk analisis image apa yang dapat diselesaikan melalui program

```
import cv2
```

```
import numpy as np
```

```
from matplotlib import pyplot as plt
```

```
img = cv2.imread('C:/PCD/latihan/gambar.JPG', 0)
```

```
hist, bins = np.histogram(img.flatten(), 256, [0, 256])
```

```
cdf = hist.cumsum()
```

```
cdf_normalized = cdf * hist.max() / cdf.max()
```

```
plt.plot(cdf_normalized, color='b')
```

```
plt.hist(img.flatten(), 256, [0, 256], color='r')
```

```
plt.xlim([0, 256])
```

```
plt.legend(('cdf', 'histogram'), loc='upper left')
```

```
plt.show()
```

Cari dan ambil salah satu image atau citra dan beri nama photo2.jpg. Jelaskan masing-masing baris script pada program berikut, dan untuk analisis image apa yang dapat diselesaikan melalui program

```
import cv2
# import matplotlib
import numpy as np
from matplotlib import pyplot as plt
img = cv2.imread('C:/PCD/latihan/subpixel4.png', 0)
hist, bins = np.histogram(img.flatten(), 256, [0,256])
cdf = hist.cumsum()
cdf_normalized = cdf * hist.max() / hist.max()
plt.plot(cdf_normalized, color = 'b')
plt.hist(img.flatten(), 256, [0,256], color = 'r')
plt.xlim([0,256])
plt.legend(('cdf','histogram'), loc = 'upper left')
plt.show()
```

**Code pada ke 2 program di atas sama...yang membedakan hanyalah beda file gambarnya saja. Maka penjelasan dari ke 2 program di atas adalah sebagai berikut:'**

---

`import cv2:` → Mengimpor Library Opencv

`import numpy as np:` Mengimpor Library NumPy yang diberi alias np untuk digunakan untuk operasi numerik.

`from matplotlib import pyplot as plt:` Mengimpor modul Pyplot dari Matplotlib alias plt yang digunakan untuk membuat visualisasi grafik.

`img = cv2.imread('C:/PCD/latihan/gambar.JPG', 0);` → Membaca file gambar yang ada di folder latihan di dalam direktori C dengan nama file 'gambar.JPG' dengan format grayscale (mode 0) dari lokasi yang ditentukan dan menyimpannya dalam variabel `img`.

`hist, bins = np.histogram(img.flatten(), 256, [0, 256]):` → Menghitung histogram dari grayscale dan menyimpan nilai histogramnya ke variabel `hist` dan batas histogramnya ke variabel `bins`.

`cdf = hist.cumsum():` → Menghitung jumlah kumulatif dari nilai histogram dan menyimpannya dalam variabel `cdf`.

`cdf_normalized = cdf * hist.max() / cdf.max():` → Normalisasi cdf agar nilainya antara 0 dari nilai maksimum histogram dan menyimpannya kedalam variabel `cdf_normalized`.

`plt.plot(cdf_normalized, color='b'):` → Membuat plot garis biru dengan nilai `cdf_normalized`.

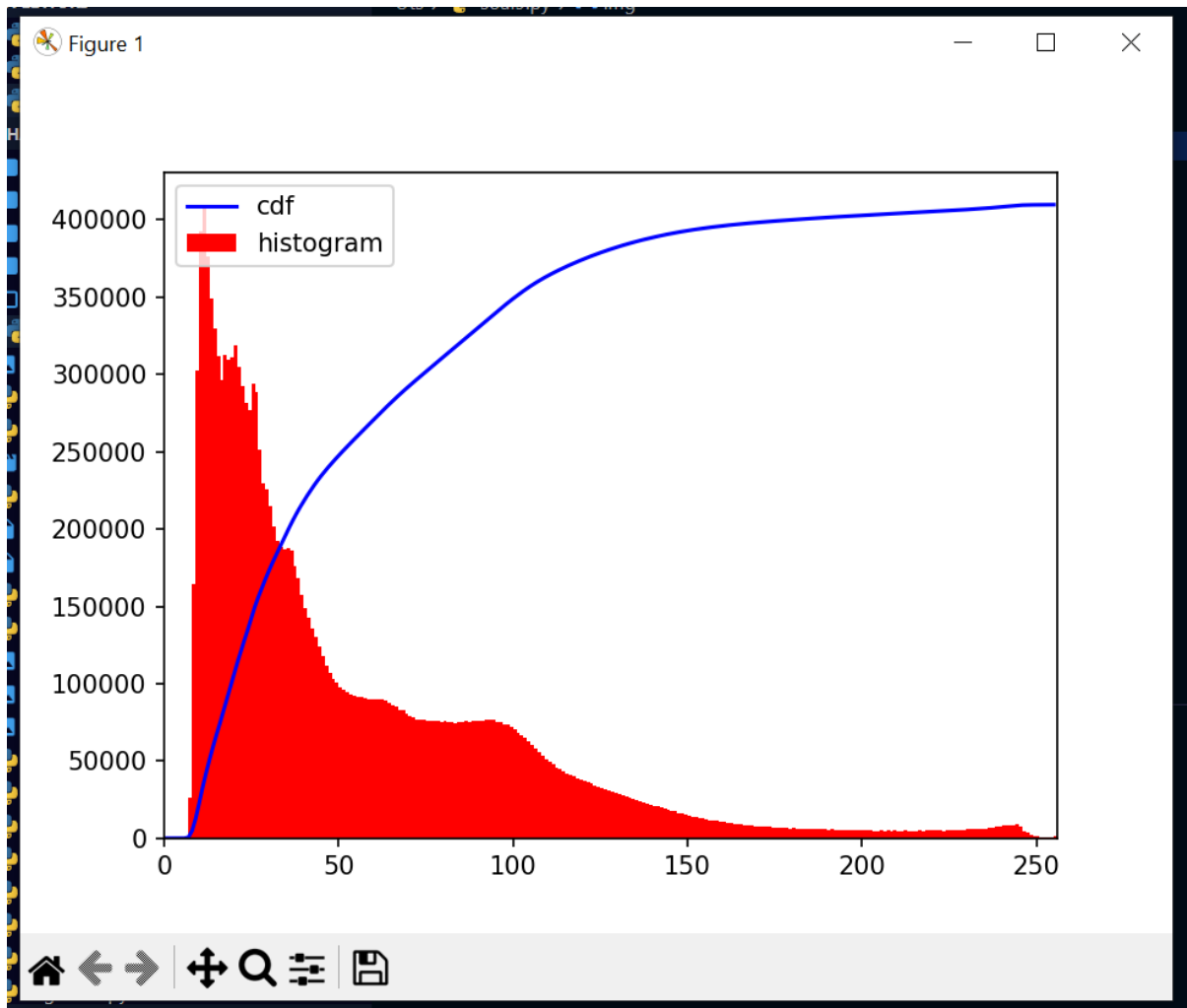
`plt.hist(img.flatten(), 256, [0, 256], color='r'):` → Membuat histogram dari grayscale dengan batas histogram yang sama dengan nilai bins dan warna merah untuk 'r'.

`plt.xlim([0, 256]):` → Mengatur batas sumbu x dari plot menjadi 0 dan 256.

`plt.legend(('cdf', 'histogram'), loc='upper left'):` → Menambahkan legenda untuk plot dengan label "cdf" dan "histogram" yang ditempatkan di kiri atas.

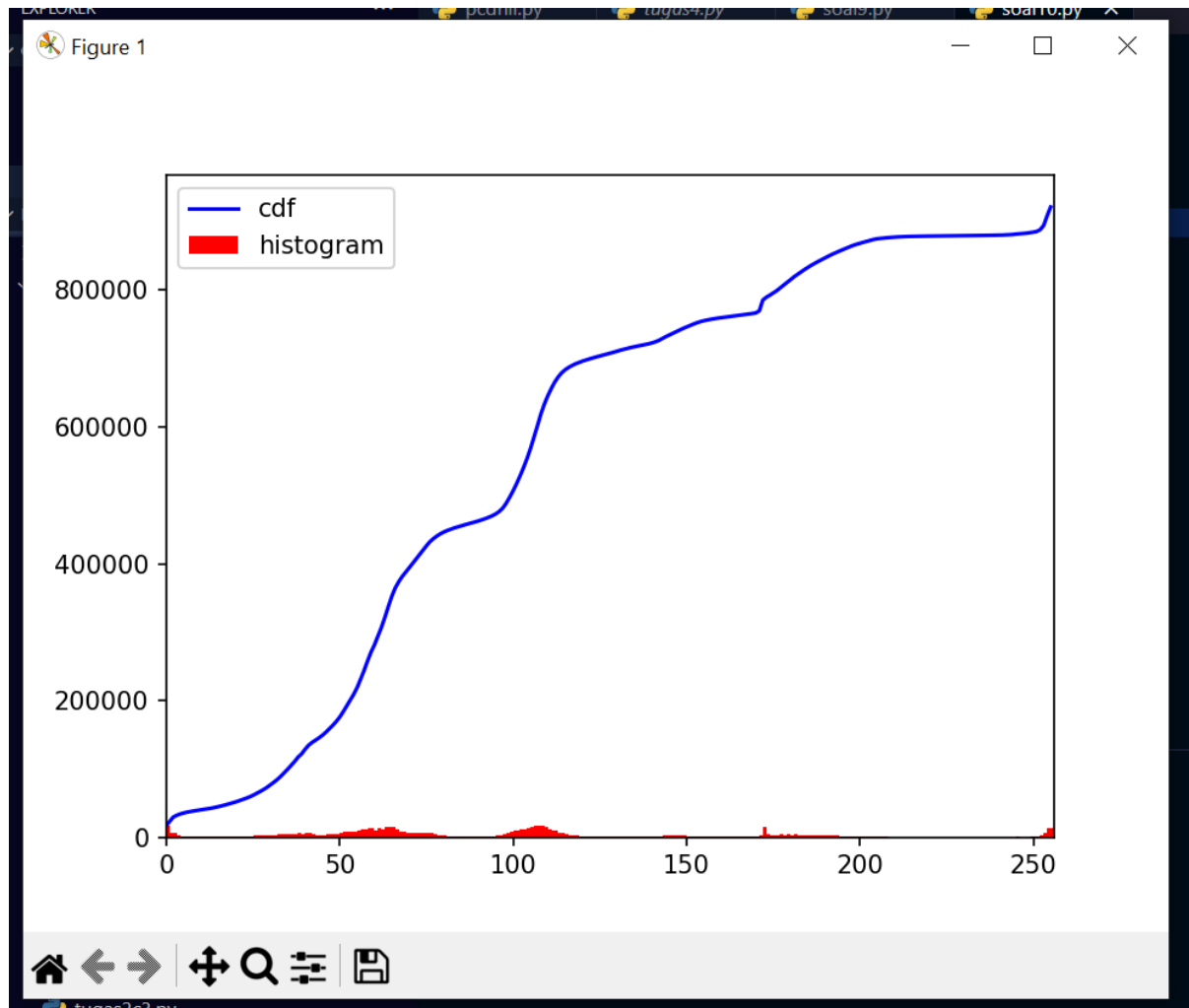
`plt.show():` → Untuk Menampilkan plot grafik.

Hasil Dari Program Script NO 1 / Soal UTS NO 9 dari gambar 1 yaitu “Gambar”





Hasil Dari Program Script NO 2 / Soal UTS NO 10 dari gambar 2 yaitu “subpixel4”



## Tugas Makalah

### 2.1 Pengertian Citra

Yang dimaksud dengan citra adalah hasil penginderaan yang telah dicetak atau berupa gambaran suatu objek yang tampak pada cermin melalui lensa kamera. Penginderaan yang dimaksud lebih tepatnya adalah penginderaan jauh yang dilakukan terhadap permukaan bumi.

Image atau gambar secara umum dapat diartikan sebagai representasi visual dari suatu objek atau situasi yang direkam atau diciptakan menggunakan berbagai media seperti kamera, pensil, kuas, atau perangkat lunak pengolahan gambar. Image dapat berupa gambar statis atau gambar yang bergerak seperti video. Image atau gambar memiliki berbagai jenis, seperti gambar digital, foto, ilustrasi, dan grafik.

Image memiliki berbagai fungsi seperti untuk mengkomunikasikan ide, merekam momen atau peristiwa, atau untuk menghibur. Image juga sering digunakan dalam industri kreatif seperti desain grafis, film, dan seni visual. Dalam pengolahan citra digital, image digunakan untuk mengolah data visual dan mendapatkan informasi yang berguna.

Image dalam pengolahan citra digital mengacu pada representasi digital dari suatu gambar atau visual. Image biasanya direpresentasikan dalam bentuk matriks bilangan yang mewakili piksel-piksel yang membentuk gambar. Setiap piksel pada gambar direpresentasikan oleh nilai intensitas yang menunjukkan kecerahan atau warna dari piksel tersebut.

Image digital dapat diperoleh dari berbagai sumber seperti kamera digital, scanner, dan bahkan dapat dibuat menggunakan perangkat lunak pengolahan citra digital. Image dapat diolah menggunakan berbagai teknik pengolahan citra digital seperti filtering, enhancement, segmentation, dan recognition.

Dalam pengolahan citra digital, image sering digunakan untuk menganalisis, mengubah, dan memproses data visual untuk mendapatkan informasi yang berguna. Misalnya, image dapat digunakan untuk mengenali objek dalam gambar, mengukur jarak dan ukuran objek, atau untuk mengidentifikasi pola-pola dalam gambar.

### 2.2 Resolusi Fitur Dalam Citra

Pengolahan citra (image Processing) merupakan proses mengolah piksel-piksel di dalam citra digital untuk tujuan tertentu. Pada awalnya pengolahan citra ini dilakukan untuk memperbaiki kualitas citra, namun dengan berkembangnya dunia komputasi yang ditandai dengan semakin meningkatnya kapasitas dan kecepatan proses komputer serta munculnya ilmu-ilmu komputasi yang memungkinkan manusia dapat mengambil informasi dari suatu citra.

Resolusi fitur (spatial resolution) dalam image atau citra mengacu pada kemampuan image untuk menampilkan detail visual dalam sebuah objek atau gambar.

Resolusi fitur dalam image atau citra dapat diukur dengan beberapa cara, di antaranya:

1. Piksel per inch (PPI): resolusi fitur diukur berdasarkan jumlah piksel yang terdapat dalam setiap inci pada gambar atau image.
2. Dots per inch (DPI): sama dengan PPI, namun biasanya digunakan dalam pencetakan atau output visual lainnya.
3. Micron per pixel ( $\mu\text{m}/\text{px}$ ): resolusi fitur diukur berdasarkan ukuran objek terkecil yang dapat ditangkap oleh piksel pada citra atau gambar.
4. Line pairs per millimeter (LP/mm): resolusi fitur diukur berdasarkan jumlah garis hitam dan putih yang terlihat pada gambar dalam satu milimeter.

Semakin tinggi nilai resolusi fitur pada suatu gambar, semakin detail dan halus gambar tersebut. Namun, semakin tinggi nilai resolusi fitur, maka semakin besar ukuran file gambar atau citra yang dihasilkan. Oleh karena itu, perlu diperhatikan penggunaan gambar pada aplikasi tertentu agar ukuran file tidak terlalu besar atau terlalu kecil.

### 2.3 Jenis Software Pengelola Citra

Ada beberapa jenis software pengolah gambar, diantaranya:

1. Adobe Photoshop: Software ini merupakan salah satu software pengolah gambar terbaik dan paling populer. Photoshop memiliki banyak fitur dan tool untuk mengedit gambar, seperti mengubah ukuran, memotong, mengganti latar belakang, menambahkan efek, dan lain sebagainya.
2. GIMP: GIMP adalah software pengolah gambar open-source yang gratis. Software ini memiliki banyak fitur yang mirip dengan Adobe Photoshop, seperti mengubah ukuran gambar, memotong, mengganti latar belakang, menambahkan efek, dan lain sebagainya.
3. CorelDRAW: CorelDRAW merupakan software pengolah gambar vektor yang biasa digunakan untuk membuat desain grafis, seperti logo, brosur, poster, dan lain sebagainya.
4. Canva: Canva adalah software pengolah gambar yang bisa diakses secara online. Software ini memiliki banyak template yang siap digunakan untuk membuat desain, seperti poster, brosur, undangan, dan lain sebagainya. Canva juga memiliki fitur untuk mengedit gambar, menambahkan teks, dan memasukkan elemen desain lainnya.
5. Paint.NET: Paint.NET merupakan software pengolah gambar yang gratis dan mudah digunakan. Software ini memiliki fitur dasar seperti mengubah ukuran gambar, memotong, dan menambahkan efek sederhana pada gambar.

### 2.4 Pentingnya Pengelolaan Citra

Manfaat pengolahan citra adalah menunjang kebutuhan kehidupan sehari-hari khususnya untuk : Memfasilitasi penyimpanan dan transmisi citra seperti menentukan metode penyimpanan citra yang efisien dalam suatu kamera digital sehingga mempercepat proses pengiriman citra dari jarak jauh misalkan dari planet Mars ke Bumi. Menyiapkan untuk ditampilkan di monitor atau di cetak. Proses yang dilakukan adalah melakukan merubah ukuran citra yang harus disesuaikan dengan ukuran media tampilan serta proses halftoning untuk proses pencetakan. Meningkatkan dan memperbaiki citra dengan menghilangkan goresan-goresan pada ataupun meningkatkan visibilitas citra. Ekstraksi informasi citra misalkan character recognizing, pengukuran pluvi air dari citra aerial.

Memperbaiki kualitas gambar, dilihat dari aspek radiometric dan aspek geometric. Aspek radiometric terdiri dari peningkatan kontras, restorasi citra, transformasi warna sedangkan aspek geometric terdiri dari rotasi, skala, translasi, transformasi geometric).

Pengelolaan image sangat penting karena gambar atau image merupakan salah satu elemen penting dalam komunikasi visual. Image dapat digunakan untuk memperjelas informasi, mempertajam pesan, dan memberikan kesan estetika yang menarik. Berikut adalah beberapa alasan mengapa pengelolaan image penting:

1. Menjaga kualitas image: Image yang tidak dikelola dengan baik dapat mengalami kerusakan, seperti hilangnya kualitas, distorsi, atau hilangnya detail. Pengelolaan image dapat membantu menjaga kualitas gambar agar tetap baik dan tidak mengalami kerusakan.

2. Meningkatkan efektivitas komunikasi: Image yang disusun dengan baik dapat meningkatkan efektivitas komunikasi. Hal ini karena image yang jelas dan teratur dapat membantu memperjelas pesan yang ingin disampaikan.
3. Menghemat waktu dan biaya: Dengan pengelolaan image yang baik, dapat menghemat waktu dan biaya dalam proses produksi. Hal ini karena image yang teratur dan mudah diakses akan memudahkan proses seleksi dan penggunaan kembali image.
4. Menjaga konsistensi brand: Pengelolaan image juga penting untuk menjaga konsistensi brand. Dengan memastikan image yang digunakan memiliki kualitas dan format yang sama, maka dapat mempertahankan citra brand yang konsisten.
5. Memudahkan akses dan penggunaan: Dengan mengelola image dengan baik, akan memudahkan akses dan penggunaan image oleh berbagai pihak. Image yang mudah dicari dan diakses akan memudahkan proses produksi dan membuat komunikasi visual menjadi lebih efisien.

Dengan demikian, pengelolaan image sangat penting untuk menjaga kualitas image, meningkatkan efektivitas komunikasi, menghemat waktu dan biaya, menjaga konsistensi brand, serta memudahkan akses dan penggunaan.