

Computer Vision

DASAR-DASAR PENGENALAN CITRA

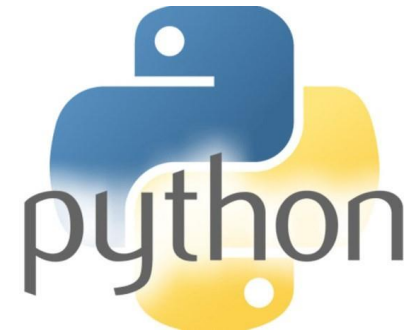
Prof. Dr. Eng. Fitri Utaminingrum, ST, MT

OUTLINE



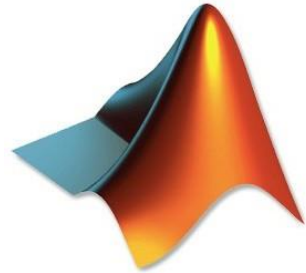
- ☐ Tool
- ☐ Matrik citra digital
- ☐ Citra biner
- ☐ Citra 4 bit (Gray dan color)
- ☐ Citra 8 bit (Gray dan Color)
- ☐ Citra berwarna 16 bit
- ☐ Citra Berwarna 24 bit

TOOLS



- Bisa menggunakan salah satu *tools* → Java, Matlab, Python dll. Namun di perkuliahan ini **contoh-contoh yang dibuat akan menggunakan MATLAB dan Python**
- Meski di beberapa tools sudah memiliki *library* atau *package* sendiri untuk pengolahan citra digital, kita harus memahami proses di dalamnya, sehingga pada beberapa bagian materi perkuliahan ini kita akan tetap belajar untuk *building from scratch*.

TOOLS



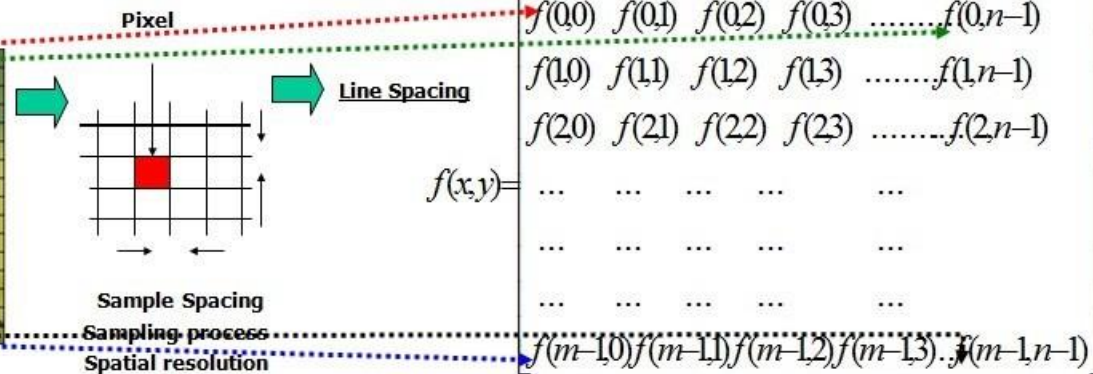
MATLAB
The Language of Technical Computing

- Minimal Matlab versi 2010 atau terbaru

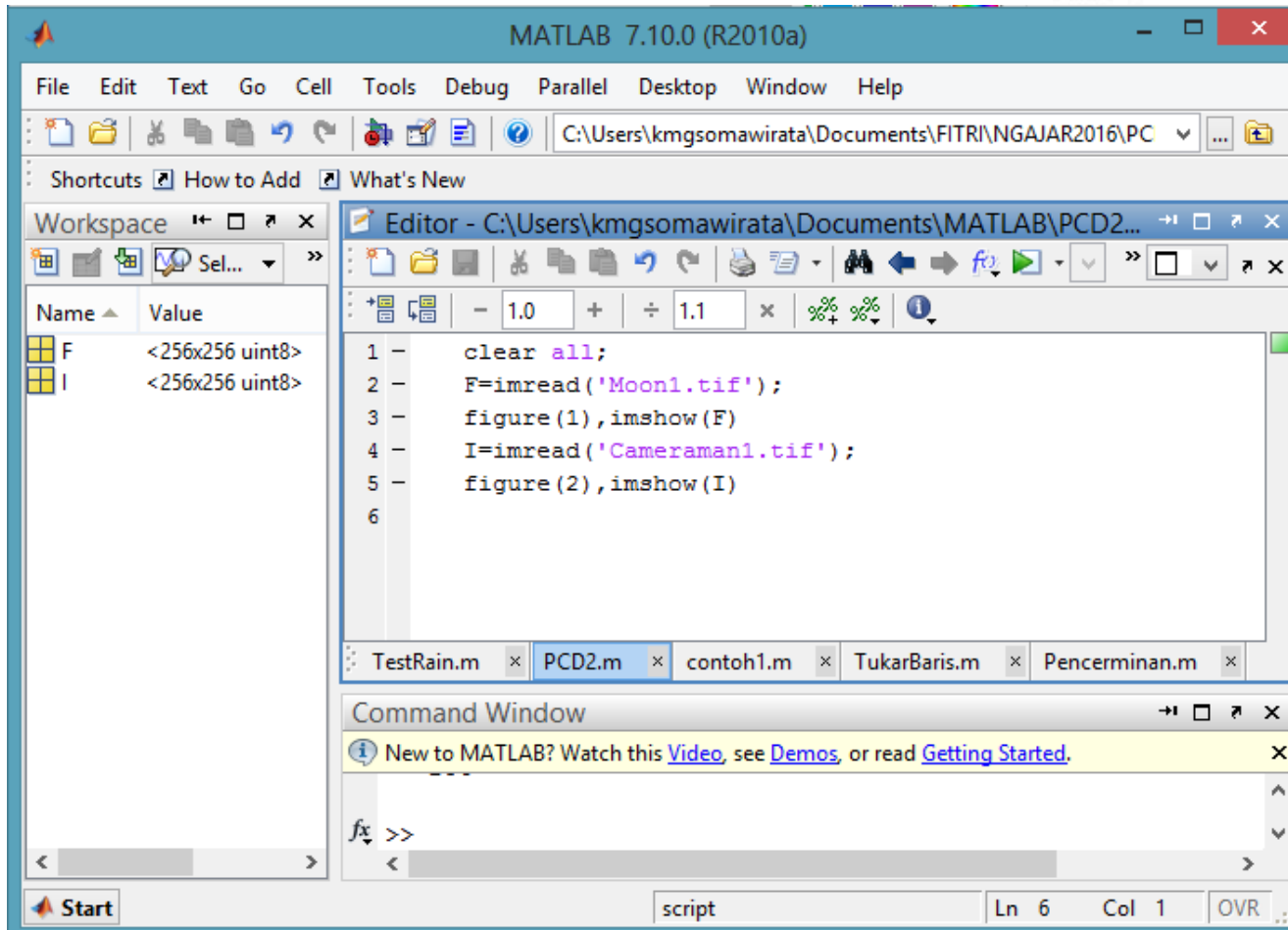


- Opencv -2.4.11
- Python -3.7.1
- numpy-1.9.1-win32-superpack-python2.7
- pycharm-community-2017.2.2
- Atau versi terbaru

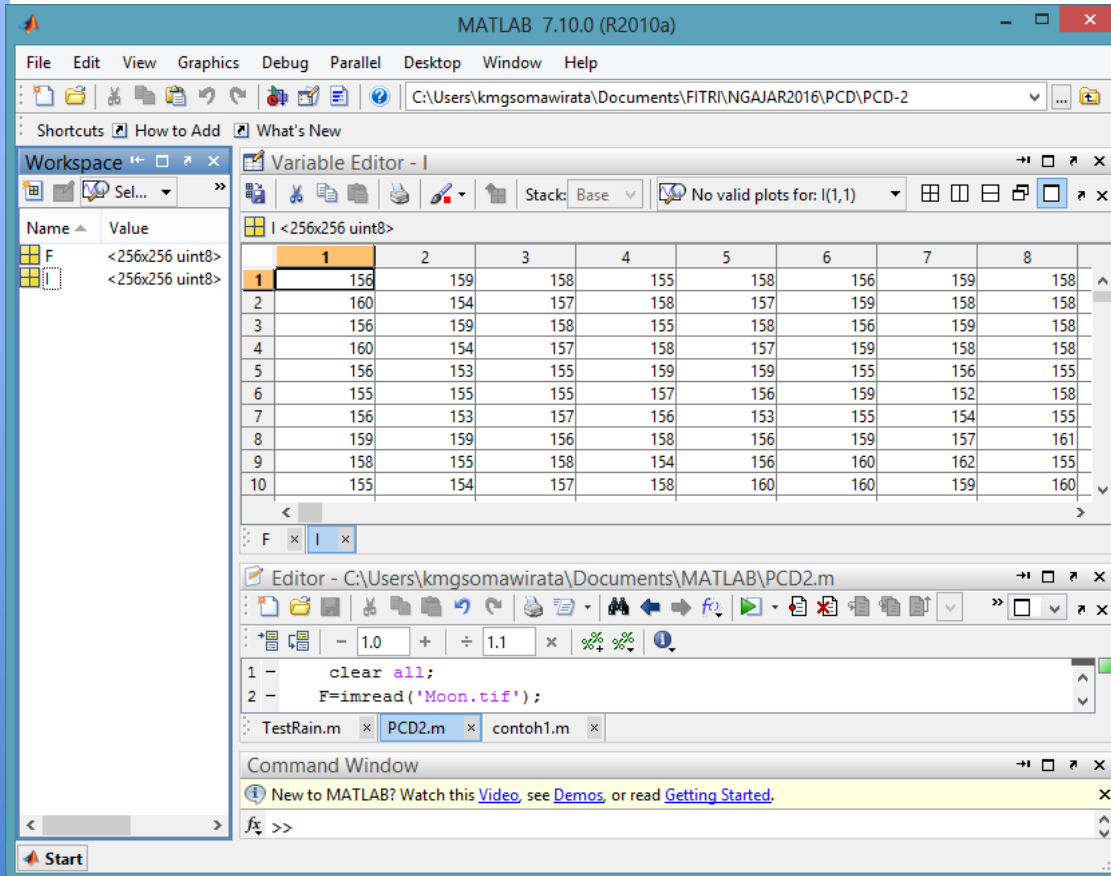
MATRIK CITRA



MATRIK CITRA



MATRIK CITRA

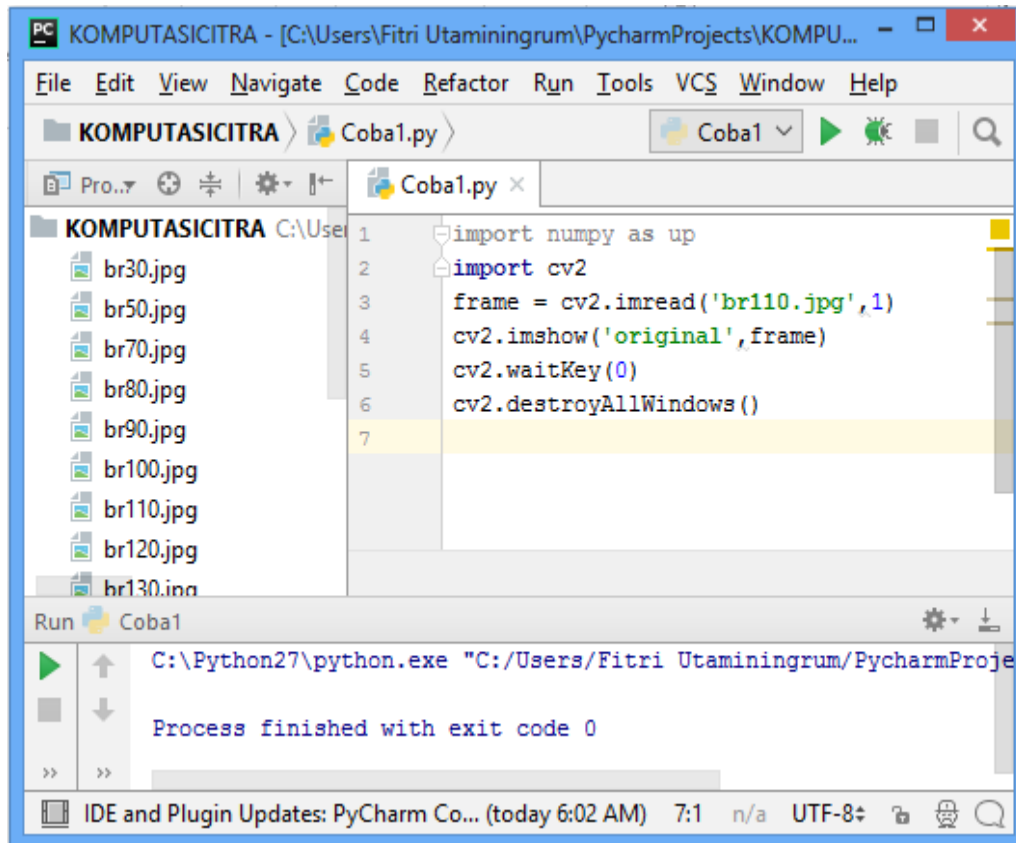


$$f(x, y) = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \dots & a_{1,M} \\ a_{2,1} & a_{2,2} & a_{2,3} & \dots & a_{2,M} \\ a_{3,1} & a_{3,2} & a_{3,3} & \dots & a_{3,M} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{N,1} & a_{N,2} & a_{N,3} & \dots & a_{N,M} \end{bmatrix}$$

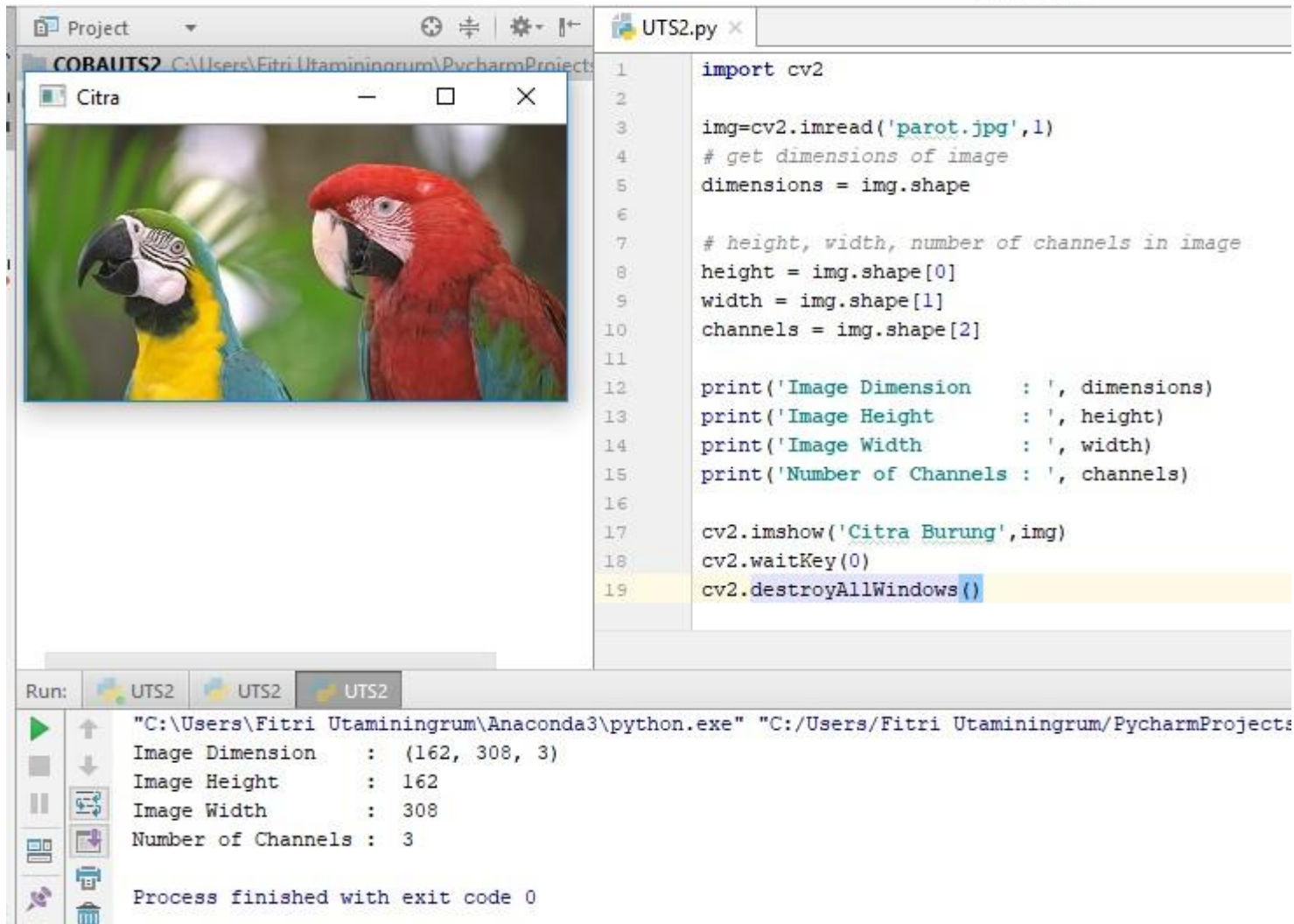
CONTOH DALAM PYTHON



Membaca file gambar dan menampilkannya



CONTOH DIMENSI CITRA



The screenshot displays a PyCharm IDE window with a project named 'COBAUTS2'. A window titled 'Citra' shows an image of two parrots. The main editor shows a Python script 'UTS2.py' that uses OpenCV to read an image and print its dimensions. The script is as follows:

```
1 import cv2
2
3 img=cv2.imread('parot.jpg',1)
4 # get dimensions of image
5 dimensions = img.shape
6
7 # height, width, number of channels in image
8 height = img.shape[0]
9 width = img.shape[1]
10 channels = img.shape[2]
11
12 print('Image Dimension : ', dimensions)
13 print('Image Height : ', height)
14 print('Image Width : ', width)
15 print('Number of Channels : ', channels)
16
17 cv2.imshow('Citra Burung',img)
18 cv2.waitKey(0)
19 cv2.destroyAllWindows()
```

The Run console at the bottom shows the output of the script:

```
Run: UTS2 UTS2 UTS2
"C:\Users\Fitri Utaminingrum\Anaconda3\python.exe" "C:/Users/Fitri Utaminingrum/PycharmProjects
Image Dimension : (162, 308, 3)
Image Height : 162
Image Width : 308
Number of Channels : 3
Process finished with exit code 0
```

KONVERSI CITRA RGB TO GRAYSCALE



Setiap pixel mempunyai nilai red (r), green (g) dan blue (b) dengan nilai masing-masing 0-255

Konversi ke Gray Scale



Setiap pixel mempunyai nilai derajat keabuan x dengan nilai 0-255

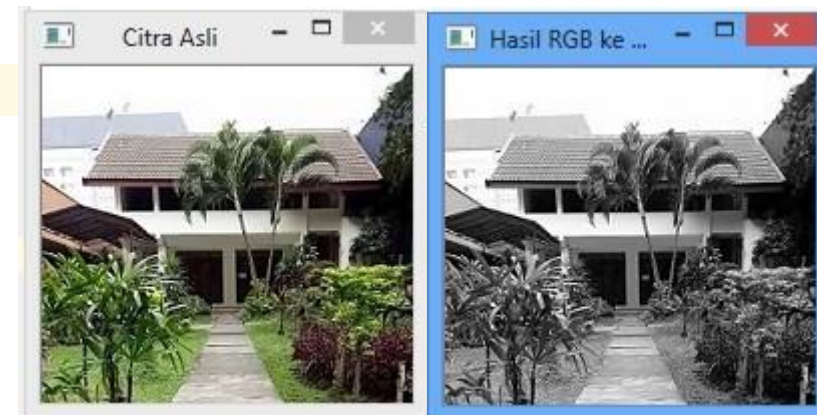
$$x = \frac{r + g + b}{3}$$

$$x = a_r \cdot r + a_g \cdot g + a_b \cdot b$$

dimana : $a_r + a_g + a_b = 1$

KONVERSI CITRA RGB TO GRAYSCALE (PHYTON)

```
1 import cv2
2 import numpy as np
3
4 img = cv2.imread('Gbr1.jpg',1)
5
6 RGBkeGray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
7
8
9 cv2.imshow('Citra Asli',img)
10 cv2.imshow('Hasil RGB ke GRAY',RGBkeGray)
11
12 cv2.waitKey(0)
13 cv2.destroyAllWindows()
14
```



KONVERSI GRAYSCALE TO BINER



Setiap pixel mempunyai nilai derajat keabuan x dengan nilai 0-255

Konversi ke Biner



Setiap pixel mempunyai nilai warna x_{bw} dengan nilai 0 dan 1

$$x_{bw} = \begin{cases} 1 & \text{jika } x \geq 128 \\ 0 & \text{jika } x < 128 \end{cases}$$

$$x_{bw} = \begin{cases} 1 & \text{jika } x \geq \bar{x} \\ 0 & \text{jika } x < \bar{x} \end{cases}$$

Merubah citra Gray ke Biner



- Untuk merubah citra gray dapat dilakukan dengan menggunakan ambang batas/Threshold
- Menggunakan fungsi `cv2.threshold()`


Merubah citra Gray ke Biner

```
1 import cv2
2 import numpy as np
3
4 img = cv2.imread('Gbr1.jpg',1)
5
6 RGBkeGray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
7 ret,thresh1 = cv2.threshold(RGBkeGray,127,255,cv2.THRESH_BINARY)
8
9
10 cv2.imshow('Citra Asli',img)
11 cv2.imshow('Hasil RGB ke GRAY',RGBkeGray)
12 cv2.imshow('Hasil GRAY ke BINER',thresh1)
13
14 cv2.waitKey(0)
15 cv2.destroyAllWindows()
```




Merubah citra Gray ke Biner


Citra Burung



Citra Burung Gray



threshold



```
COBAUTS2] - ...\\DimensiGray.py - PyCharm Community Edition 2017.2.2
CS Window Help

UTS2.py x DimensiGray.py x

1 import cv2
2
3 img=cv2.imread('parot.jpg',1)
4 cv2.imshow('Citra Burung',img)
5 # get dimensions of image
6 # height, width, number of channels in image
7 dimensionsRGB = img.shape
8 print('Image Dimension : ',dimensionsRGB)
9
10 RGBtoGray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
11 cv2.imshow('Citra Burung Gray',RGBtoGray)
12 # height, width, number of channels in image
13 dimensionsGray = RGBtoGray.shape
14 print('Image Dimension Gray : ',dimensionsGray)
15
16 # Gray to Binary
17 retval, threshold = cv2.threshold(RGBtoGray, 125, 255, cv2.THRESH_BINARY)
18 cv2.imshow('threshold',threshold)
19 dimensionsBinary = threshold.shape
20 print('Image Dimension Binary : ',dimensionsBinary)
21
22 cv2.waitKey(0)
23 cv2.destroyAllWindows()
```

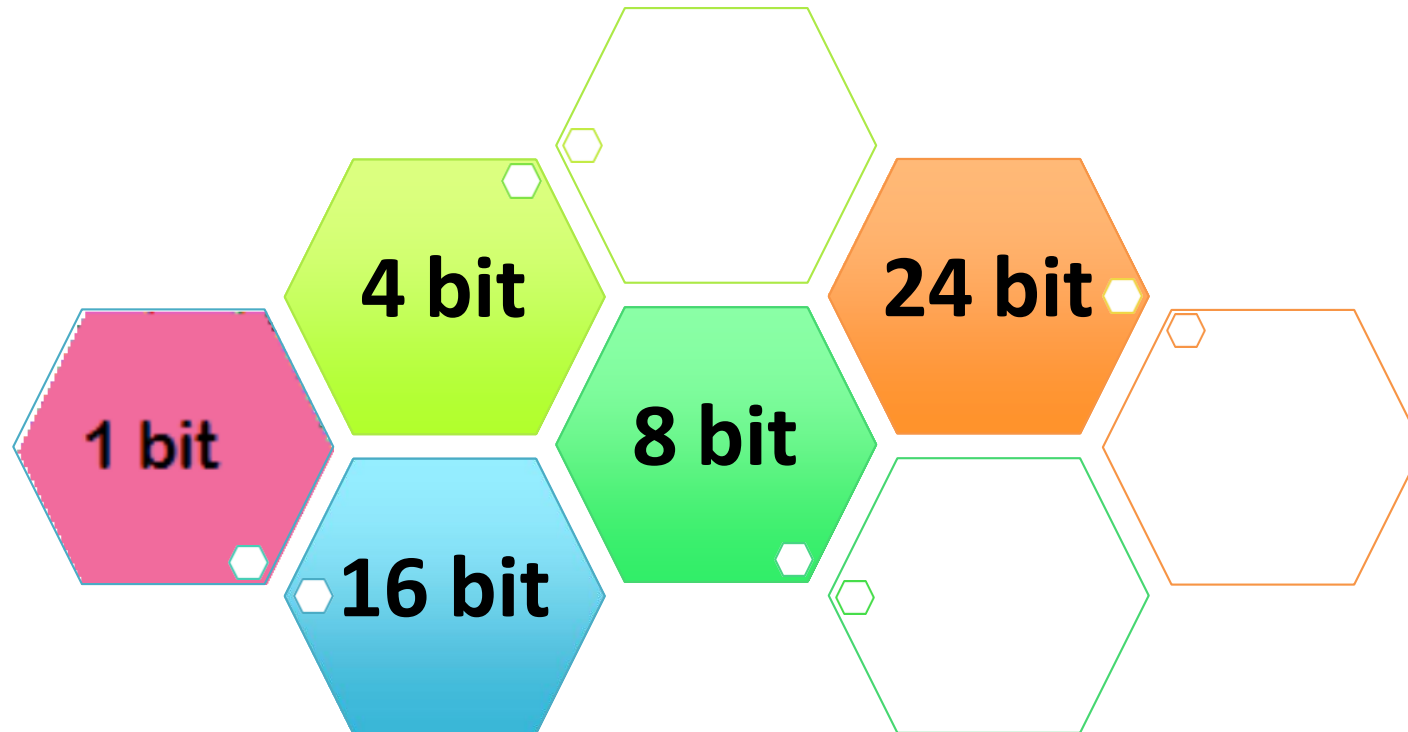
DimensiGray DimensiGray DimensiGray

"C:\\Users\\Fitri Utaminingrum\\Anaconda3\\python.exe" "C:/Users/Fitri Utaminingrum/PycharmProjects/COBAUTS2/DimensiGray.py"

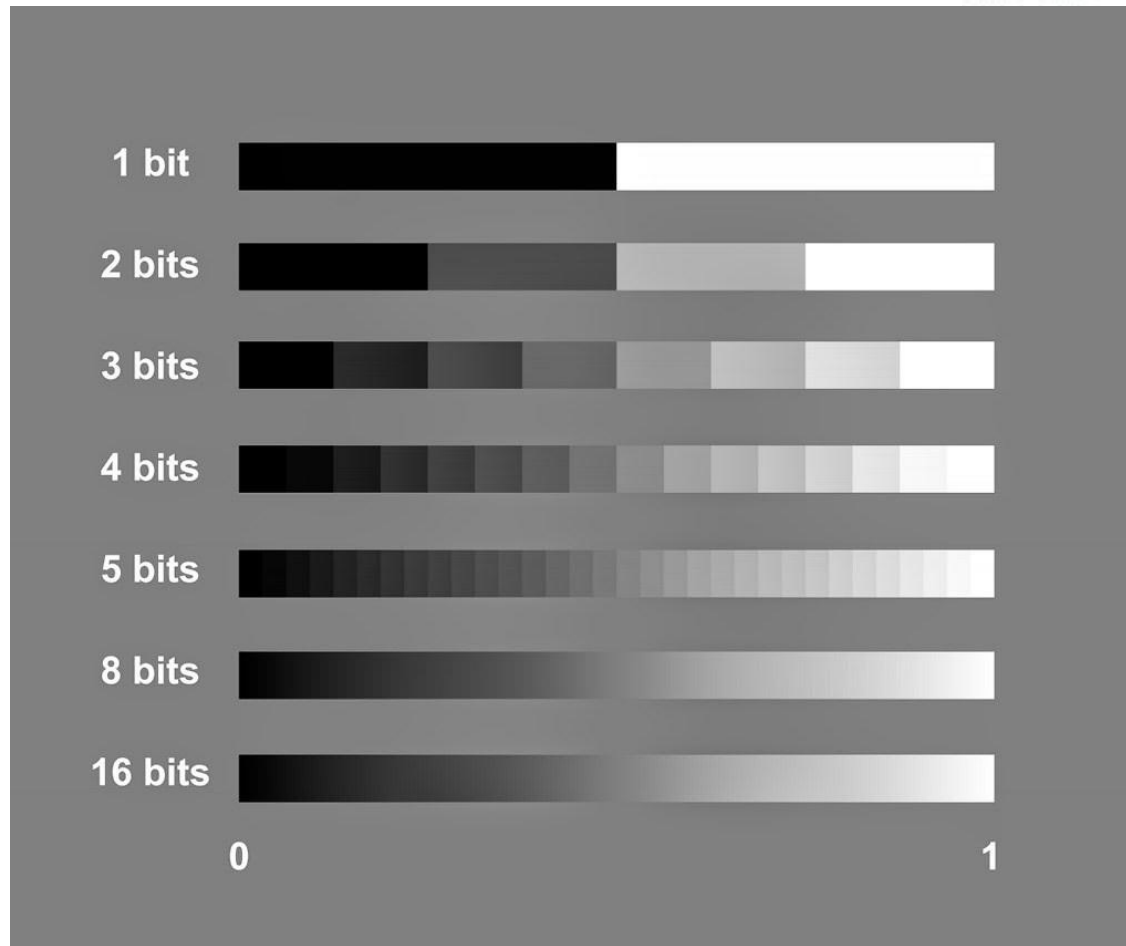
Image Dimension : (162, 308, 3)
Image Dimension Gray : (162, 308)
Image Dimension Binary : (162, 308)

Data Citra Digital

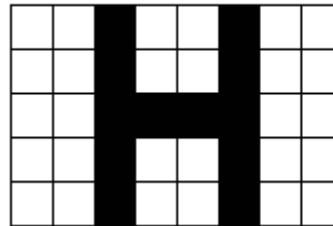
Sebuah citra digital dapat disusun oleh data citra digital mulai dari :



Citra Digital Hitam Putih



Contoh Data Citra Biner / 1 bit



Citra Biner (hitam = 0, putih = 1)

= 1 1 0 1 1 0 1 1

= 1 1 0 1 1 0 1 1

= 1 1 0 0 0 0 1 1

= 1 1 0 1 1 0 1 1

= 1 1 0 1 1 0 1 1



Citra True Color



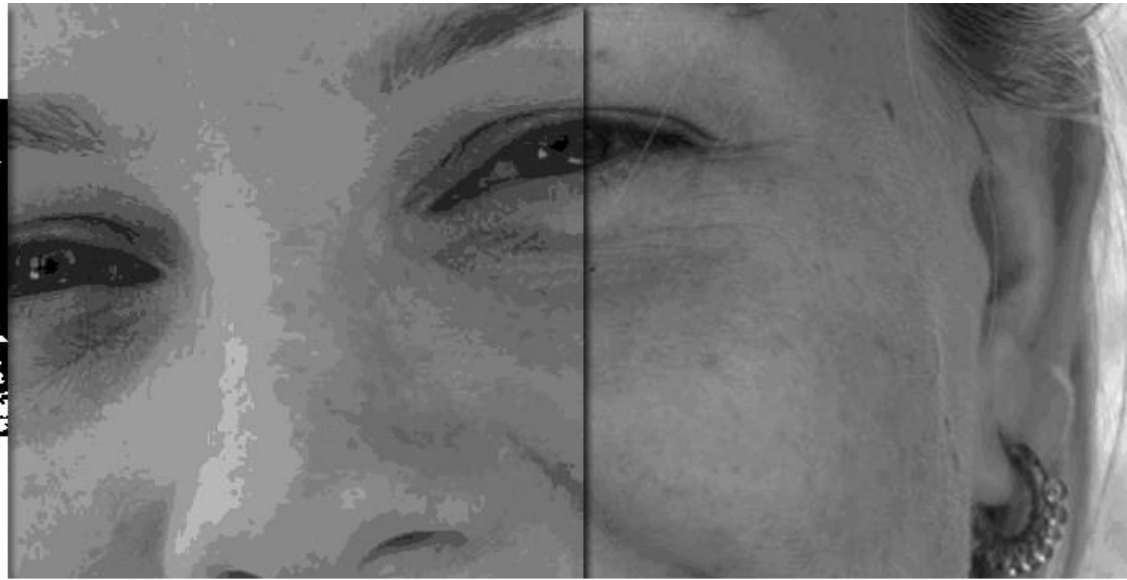
Citra Biner

Contoh Perbandingan Kualitas

Semakin lebar data citra digital maka kualitasnya semakin bagus



1-bit



4-bit

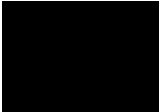















8-bit

Data Citra Berwarna

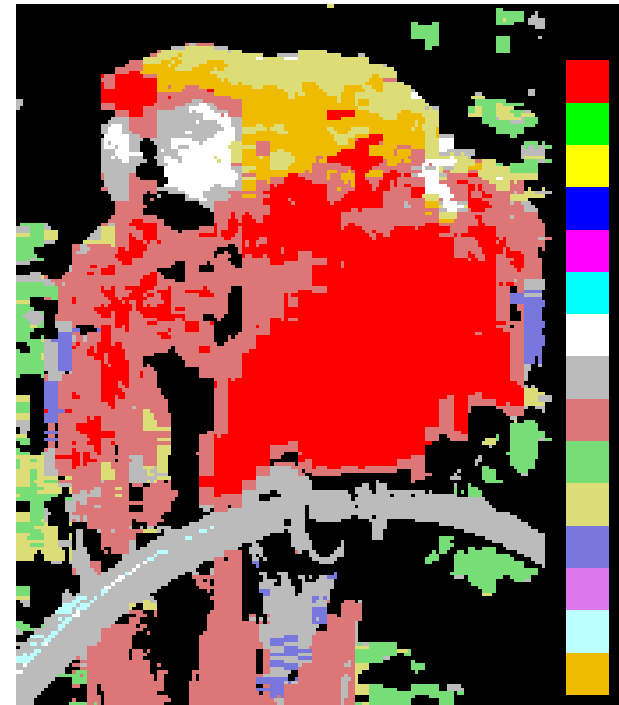


- 4 bit
- 8 bit
- 16 bit
- 24 bit

Kombinasi Berwarna 4 bit

			
Black 0, 0, 0	Dark Red 128, 0, 0	Red 255, 0, 0	Pink 255, 0, 255
			
Teal 0, 128, 128	Green 0, 128, 0	Bright Green 0, 255, 0	Turquoise 0, 255, 255
			
Dark Blue 0, 0, 128	Violet 128, 0, 128	Blue 0, 0, 255	Gray 25% 192, 192, 192
			
Gray 50% 128, 128, 128	Dark Yellow 128, 128, 0	Yellow 255, 255, 0	White 255, 255, 255

www.infotart.com



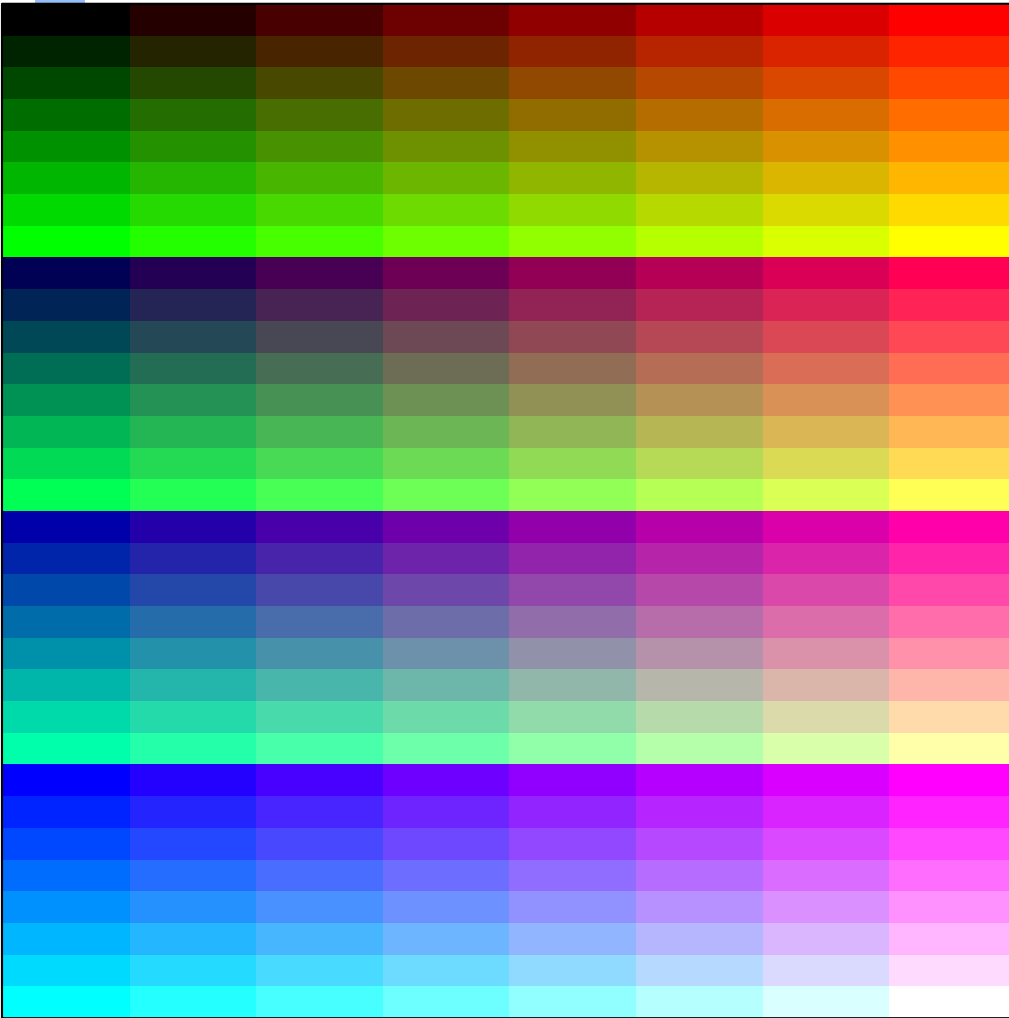
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Kombinasi Berwarna 8 bit

Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0
R	R	R	G	G	G	B	B

Kombinasi Berwarna 16 bit

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
R	R	R	R	R	G	G	G	G	G	G	B	B	B	B	B





Kombinasi Berwarna 24 bit

- Kombinasi Warna 24 bit disusun oleh 8-bit untuk R, 8-bit untuk G dan 8-bit untuk B



24-BIT COLOR
16 MILLION COLORS
1.2 MB

8-BIT COLOR
256 COLORS
420 K

8-BIT B/W
256 GRAYS
320 K

1-BIT B/W
2 COLORS
42 K

Perbandingan
kualitas citra
untuk lebar
bit data yang
berbeda

MODIFIKASI KECEMERLANGAN



- Pada dasarnya merubah nilai keabuan/warna dari gelap menuju terang adalah menggunakan formula linier :

$$\text{Grey Scale : } Po(x,y) = Pi(x,y) + C$$

Ket :

$Po(x,y)$ = Pixel Output pada koordinat x,y

$Pi(x,y)$ = Pixel Input pada koordinat x,y

C = Constanta

jika C positif maka lbh terang, C negatif lbh gelap

Untuk citra true color tentunya formula/rumus diubah dengan melibatkan RGB-nya :

$$Ro(x,y) = Ri(x,y) + C$$

$$Go(x,y) = Gi(x,y) + C$$

$$Bo(x,y) = Bi(x,y) + C$$

MODIFIKASI KECEMERLANGAN



Modifikasi Kecemerlangan $f'(x,y) = f(x,y) + C$...dengan $C = -100$

NEGASI



- Merubah citra asli menjadi negasinya itu konsepnya adalah membalikan warna dengan batas 0 dan 255, artinya jika punya warna 0 menjadi 255 dan jika punya warna 255 menjadi 0 , formula liniernya sbb :

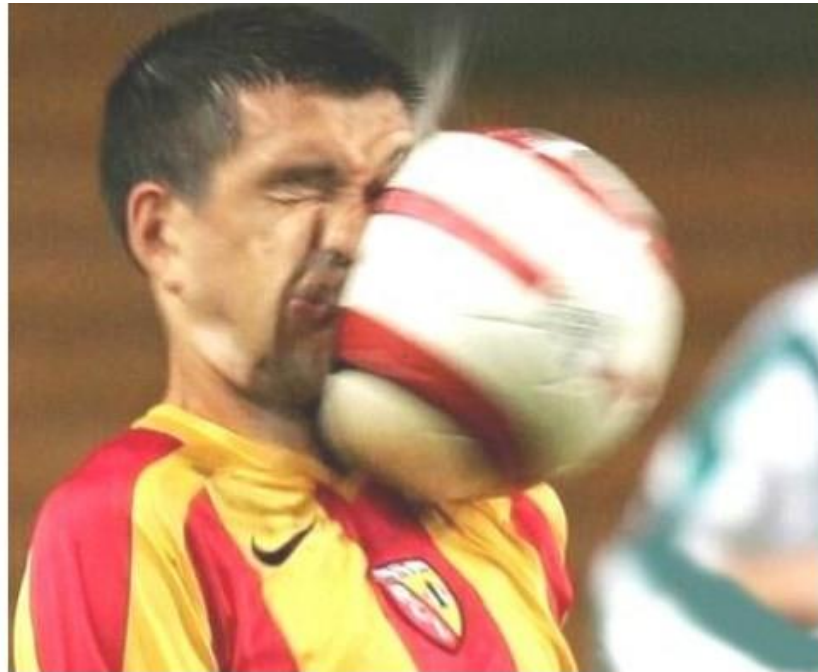
$$Po(x,y) = 255 - Pi(x,y)$$

Ket :

$Po(x,y)$ = Pixel Output pada koordinat x,y

$Pi(x,y)$ = Pixel Input pada koordinat x,y

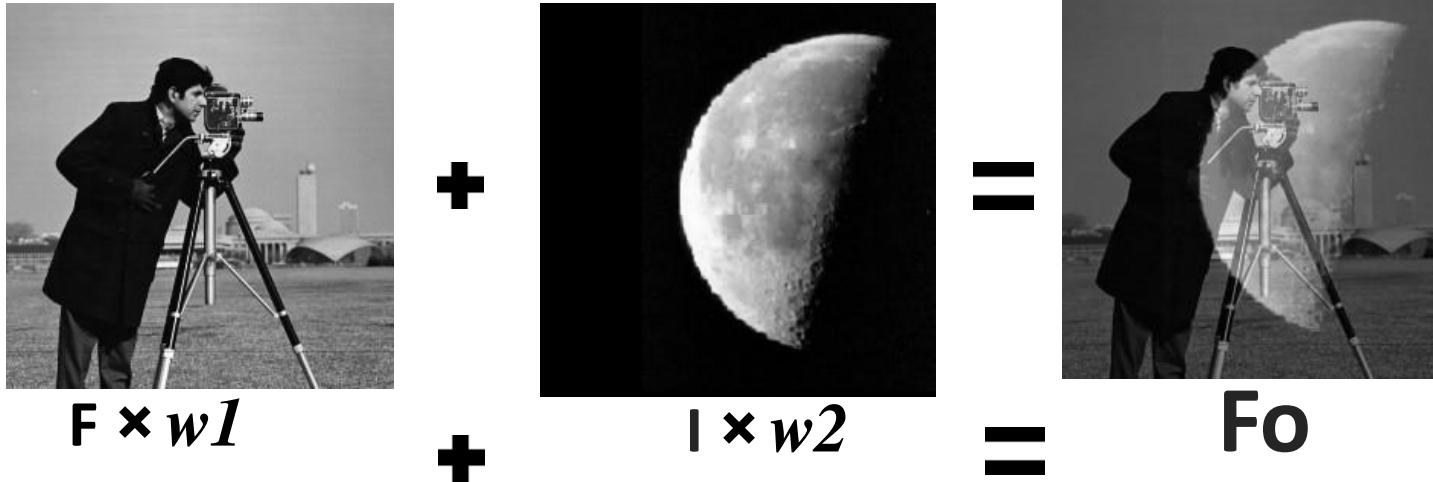
NEGASI



Operasi Negasi

OPERASI DASAR PADA Matrik Citra

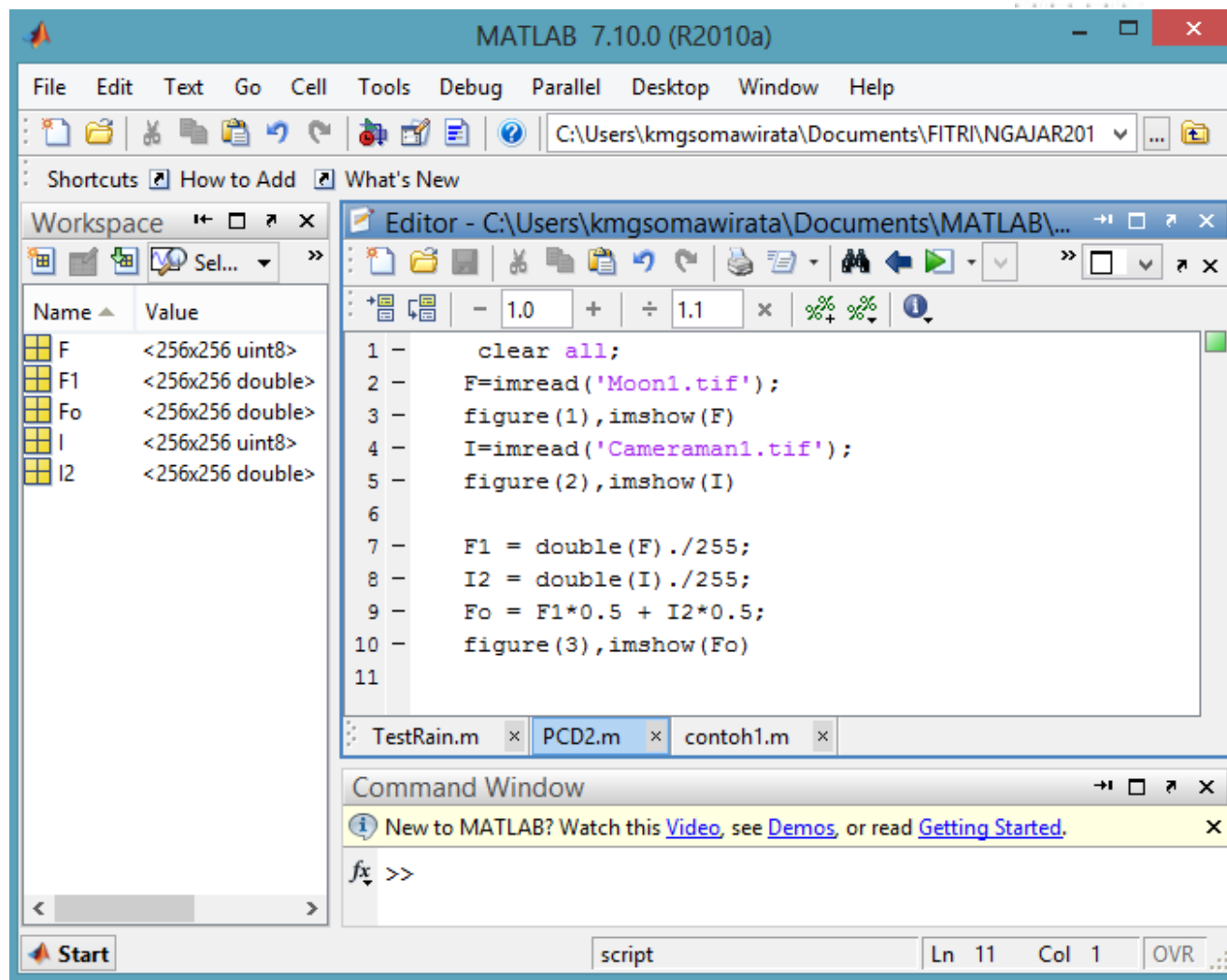
Operasi Penjumlahan


$$F \times w_1 + I \times w_2 = F_o$$

Syarat citra dapat dijumlahkan:

- Ukuran dan dimensi dari citra harus sama
- Nilai bobot (w_1 dan w_2) jika dijumlahkan sama dengan satu, Jika kurang dari satu maka citra hasil akan lebih gelap dan sebaliknya.
- Format data citra dari uint8 harus dirubah ke double

OPERASI PENJUMLAHAN DALAM MATLAB



Operasi Penukaran baris menjadi kolom/Matrik Transpose



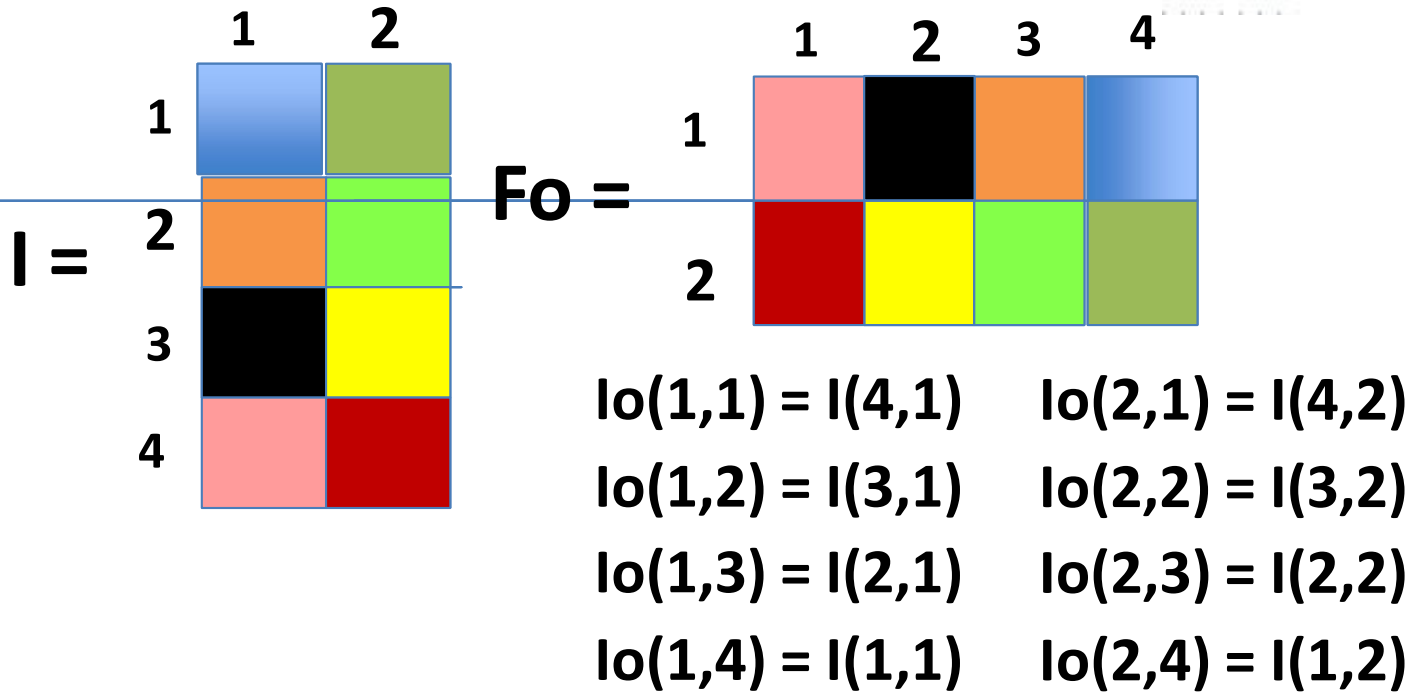
I



Fo

- Baris terakhir menjadi kolom pertama pada citra output
- Kolom pertama menjadi baris pertama pada citra output

ANALISA



□ JmlBaris Fo=JlmKolom I dan JmlKolom Fo = JmlBaris I, sehingga dapat dirumuskan

□ $Fo(\text{Baris}, \text{Kolom}) = I(\text{JmlBarisI} - (\text{Kolom} - 1), \text{Baris})$

□ $Fo(\text{Baris}, \text{Kolom}) = I(\text{JmlBarisI} - \text{Kolom} + 1, \text{Baris})$

IMPLEMENTASI DALAM MATLAB

MATLAB 7.10.0 (R2010a)

File Edit Text Go Cell Tools Debug Parallel Desktop Window Help

E:\NGAJAR-FITRI\PCD\PCD-2

Shortcuts How to Add What's New

Workspace

Name	Value
Baris	256
Fo	<256x256 uint8>
I	<256x256 uint8>
JmlBarisI	256
JmlBarisO	256
JmlKolomI	256
JmlKolomO	256
Kolom	256

Editor - E:\NGAJAR-FITRI\PCD\PCD-2\Coba.m

```
1 clear all
2 %I=imread('pout.tif');
3 I=imread('Cameraman1.tif');
4 figure(1),imshow(I)
5 [JmlBarisI JmlKolomI]=size(I);
6 JmlBarisO=JmlKolomI;
7 JmlKolomO=JmlBarisI;
8 for Baris = 1 : JmlBarisO
9     for Kolom = 1:JmlKolomO
10        %Fo (Baris,Kolom) = I (JmlKolomO-Kolom+1,Baris);
11        Fo (Baris,Kolom) = I (JmlKolomO-(Kolom-1), Baris);
12    end
13 end
14 figure(2),imshow(Fo)
```

Command Window

New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

Start script Ln 14 Col 21 OVR

IMPLEMENTASI DALAM PYTHON

```
Rotate.py x RotasiManual.py x ContohFlipping.py x
3
4 img=cv2.imread('parot.jpg')
5 cv2.imshow('Citra Input',img[:, :,0])
6 Fo = np.ones((img.shape[1],img.shape[0]), dtype="uint8")
7
8 hl,wl = img.shape[:2]
9
10 for i in range(Fo.shape[0]):
11     for j in range(Fo.shape[1]):
12         Fo[i,j]=img[hl-1-j,i,0]
13
14
15 cv2.imshow('Citra Hasil',Fo)
16 cv2.waitKey(0)
17 cv2.destroyAllWindows()
18
```



OPERASI PENCERMINAN



I



Fo

- Urutan baris untuk citra Fo = baris citra I
- Kolom pertama citra Fo = Kolom terakhir citra I

OPERASI PENCERMINAN ANALISA



□ Jika citra masukkan I dan citra keluaran Fo

- | | | |
|------------------------|------------------------|------------------------|
| ▪ $Fo(1,1) = I(1,256)$ | ▪ $Fo(2,1) = I(2,256)$ | ▪ $Fo(3,1) = I(3,256)$ |
| ▪ $Fo(1,2) = I(1,255)$ | ▪ $Fo(2,2) = I(2,255)$ | ▪ $Fo(3,2) = I(3,255)$ |
| ▪ $Fo(1,3) = I(1,254)$ | ▪ $Fo(2,3) = I(2,254)$ | ▪ $Fo(3,3) = I(3,254)$ |
| ▪ $Fo(1,5) = I(1,253)$ | ▪ $Fo(2,5) = I(2,253)$ | ▪ $Fo(3,5) = I(3,253)$ |
| ▪ $Fo(1,5) = I(1,252)$ | ▪ $Fo(2,5) = I(2,252)$ | ▪ $Fo(3,5) = I(3,252)$ |
| ▪ \vdots | ▪ \vdots | ▪ \vdots |
| ▪ \vdots | ▪ \vdots | ▪ \vdots |
| ▪ $Fo(1,255) = I(1,2)$ | ▪ $Fo(2,255) = I(2,2)$ | ▪ $Fo(3,255) = I(3,2)$ |
| ▪ $Fo(1,256) = I(1,1)$ | ▪ $Fo(2,256) = I(2,1)$ | ▪ $Fo(3,256) = I(3,1)$ |

□ Dan seterusnya

□ Ukuran citra I= 256X256 sehingga JmlBaris=256 dan JmlKolom=256,

□ JmlBaris Fo=JmlBaris I dan JmlKolom Fo = JmlKolom I, sehingga dapat dirumuskan

$$\square \quad Fo(\text{Baris}, \text{Kolom}) = I(\text{Baris}, \text{JmlKolom} + 1 - \text{Kolom})$$

IMPLEMENTASI PENCERMINAN

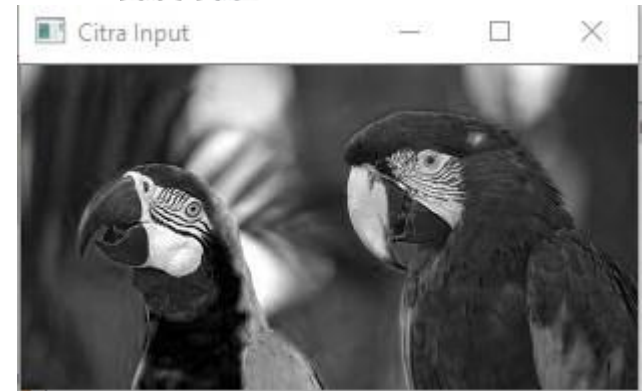


```
Editor - C:\Users\kmgsumawirata\Documents\FITRI\NGAJAR2016\...
+ [New] [Open] [Save] | [Cut] [Copy] [Paste] [Undo] [Redo] | [Print] [Find] | [Zoom In] [Zoom Out] [Zoom Reset] [Full Screen] [Help]
+ [New] [Open] [Save] | - 1.0 + ÷ 1.1 × % % | [Info]

1 - function Fo= Pencerminan(fi)
2 -     [JmlBaris JmlKolom] = size(fi);
3 -     for Baris = 1 : JmlBaris
4 -         for Kolom = 1: JmlKolom
5 -             Fo(Baris,Kolom) = fi(Baris,JmlKolom+1-Kolom);
6 -
7 -         end
8 -     end
9 -     return
10
```

IMPLEMENTASI PENCERMINAN PYTHON

```
Rotate.py x RotasiManual.py x PencerminanManual.py x Co
1 import cv2
2 import numpy as np
3
4 img=cv2.imread('parot.jpg')
5 cv2.imshow('Citra Input',img[:, :,0])
6 Fo = np.ones((img.shape[0],img.shape[1]), dtype="uint8")
7
8 h1,w1 = img.shape[:2]
9
10 for i in range(img.shape[0]):
11     for j in range(img.shape[1]):
12         Fo[i,j]=img[i,w1-1-j,0]
13
14
15 cv2.imshow('Citra Hasil',Fo)
16 cv2.waitKey(0)
17 cv2.destroyAllWindows()
18
```



CROPPING

Cropping adalah memotong sebagian citra menjadi ukuran yang lebih kecil

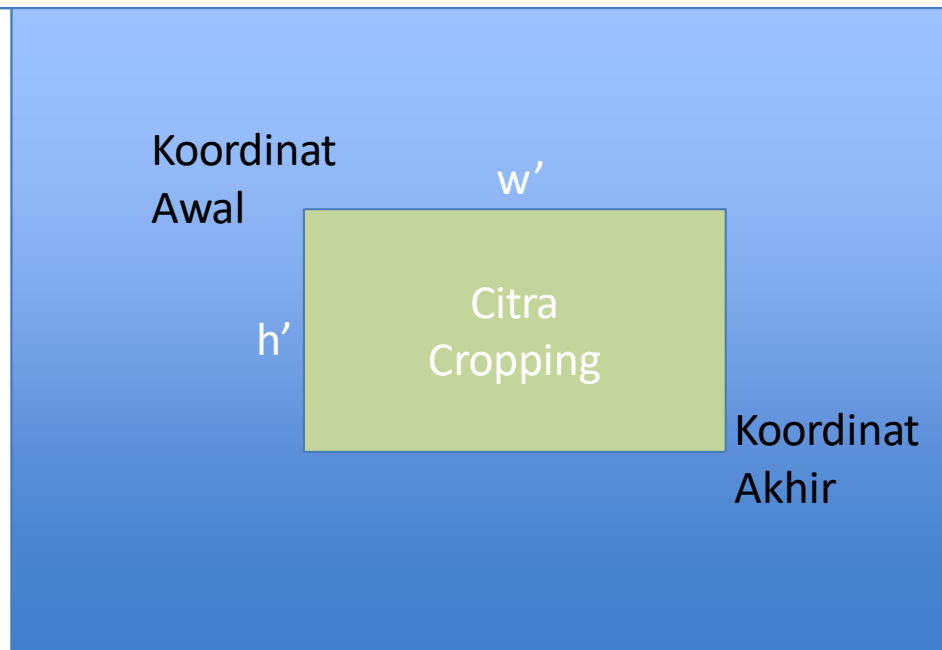
Image

KoordinatAwal(x1,y1)

KoordinatAkhir(x2,y2)

$$h' = x2 - x1$$

$$w' = y2 - y1$$



CROPPING

```
Rotate.py x RotasiManual.py x PencerminanManual.py x Cropping.py x Co
1 import cv2
2 import numpy as np
3
4 img=cv2.imread('parot.jpg')
5 cv2.imshow('Citra Input',img[:, :,0])
6
7 koAwal = [50,50]
8 koAkhir = [150,200]
9 Fo = np.ones((koAkhir[0]-koAwal[0],koAkhir[1]-koAwal[1]), dtype="uint8")
10
11 hl,wl = img.shape[:2]
12
13 for i in range(Fo.shape[0]):
14     for j in range(Fo.shape[1]):
15         Fo[i,j]=img[i+koAwal[0],j+koAwal[1],0]
16
17
18 cv2.imshow('Citra Hasil',Fo)
19 cv2.waitKey(0)
20 cv2.destroyAllWindows()
21
```

