MODUL 10 – Morfologi (**Erosi, Dilasi, Opening, Closing, Transformasi Top-hat dan Bottom-hat, Skeleton, Thickening)**

A. TUJUAN

1. Mahasiswa mampu memahami konsep Morphology
2. Mahasiswa dapat mengetahui beberapa teknik Morphology
3. Mahasiswa dapat membuat beberapa teknik morphology menggunakan Python pada Google Colab

B. ALAT DAN BAHAN

1. PC/LAPTOP
2. Github
3. *Google Colaborator*

C. ULASAN TEORI

C.1 Pengertian Operasi Morfologi

Operasi morfologi merupakan teknik pengolahan citra yang didasarkan pada bentuk segmen atau region dalam citra. Operasi morfologi biasanya diterapkan pada citra biner utamanya, ataupun warna dan citra skala keabuan. Berdasarkan nilai biner tersebut dapat dibedakan mana bagian objek dan mana bagian bukan objek atau *background*. Meskipun lebih banyak dipakai pada citra biner, operasi morfologi sering pula digunakan pada citra skala keabuan dan warna. Beberapa operasi morfologi yang bisa dilakukan adalah dilasi, Erosi, penutupan (*closing*) dan pembukaan (*opening*).

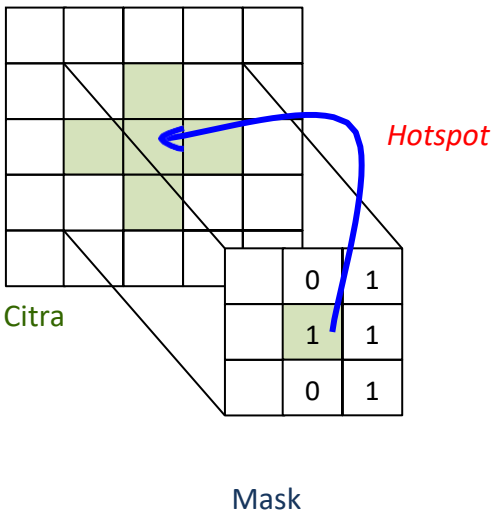
Tahapan operasi morfologi dilakukan dengan cara mempasing sebuah *Structuring Element* terhadap sebuah citra. *Structuring Element* memiliki sebuah fungsi seperti mask pada proses filtering. Dengan operasi morfologi beberapa alikasi yang dapat dilakukan adalah sebagai berikut :

- Membentuk filter spasial
- Memperoleh skeleton (rangka) objek.
- Menentukan letak objek di dalam citra.
- Memperoleh bentuk struktur objek.

Gambar 1, menjelaskan tentang pengaruh operasi morfologi dalam memisahkan beberapa objek sehingga mempermudah dalam melakukan pengamatan selanjutnya

Gambar 1. Citra sebelum dan sesudah operasi morfologi
(Sumber : Kadir, Abdul dkk, 2013)

Operasi morfologi melibatkan dua array, yaitu array yang berupa citra asli, dan juga array yang berasalah dari mask/ *structuring element* . Gambar 2 menjelaskan bagaimana penerapan operasi morfologi terhadap citra. Dengan pusat hotspot pada mask yang digunakan.



Gambar 2. Penerapan Operasi Morfologi yang Melibatkan Array Citra dan Mask
(Sumber : Kadir, Abdul dkk, 2013)

C.2 Operasi Dilasi

Operasi dilasi yang merupakan salah satu wujud dari operasi morfologi memiliki tujuan untuk memperbesar ukuran segmen objek dengan menambah lapisan di sekeliling objek. Operasi dilasi dapat dilakukan dengan 2 cara :

1. Mengubah semua titik latar yang bertetangga dengan titik batas menjadi titik objek, atau lebih mudahnya set setiap titik yang tetangganya adalah titik objek menjadi titik objek.

2. Mengubah semua titik di sekeliling titik batas menjadi titik objek, atau lebih mudahnya set semua titik tetangga sebuah titik objek menjadi titik objek.

Operasi dilasi juga digunakan untuk mendapatkan gambar yang lebih lebar dirumuskan sebagai berikut : (Gonzales & Woods, 2002):

$$A \oplus B = \left\{ z \middle| \left[(\hat{B})_z \cap A \right] \subseteq A \right\}$$

Dalam hal ini,

a) $\hat{B} = \{w | w = -b, untuk\ b \in B\}$

b) $(B)_Z = \{c | c = a + z, untuk\ a \in A\}$

c) z=(z1, z2)

Berikut ini adalah contoh operasi dilasi yang diterapkan pada matriks A dengan mask matriks B, yang digambarkan pada gambar 3.

A = { (2,2), (2,3), (2,4), (3,2), (3,3), (3,4), (4,3) }

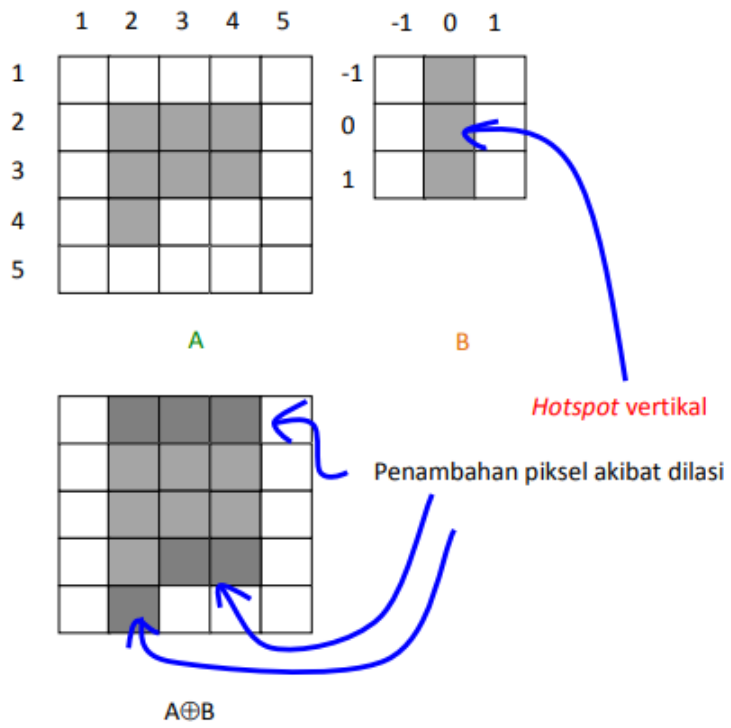
B = { (-1, 0), (0,0), (1,0) }

Dengan demikian,

A Dilasi *B* = { (2,2) + (-1, 0) , (2,2) + (0, 0) , (2,2) + (1, 0),
(2,3) + (-1, 0) , (2,3) + (0, 0) , (2,3) + (1, 0),
(2,4) + (-1, 0) , (2,4) + (0, 0) , (2,4) + (1, 0),
(3,2) + (-1, 0) , (3,2) + (0, 0) , (3,2) + (1, 0),
(3,3) + (-1, 0) , (3,3) + (0, 0) , (3,3) + (1, 0),
(3,4) + (-1, 0) , (3,4) + (0, 0) , (3,4) + (1, 0),
(4,3) + (-1, 0) , (4,3) + (0, 0) , (4,3) + (1, 0) }

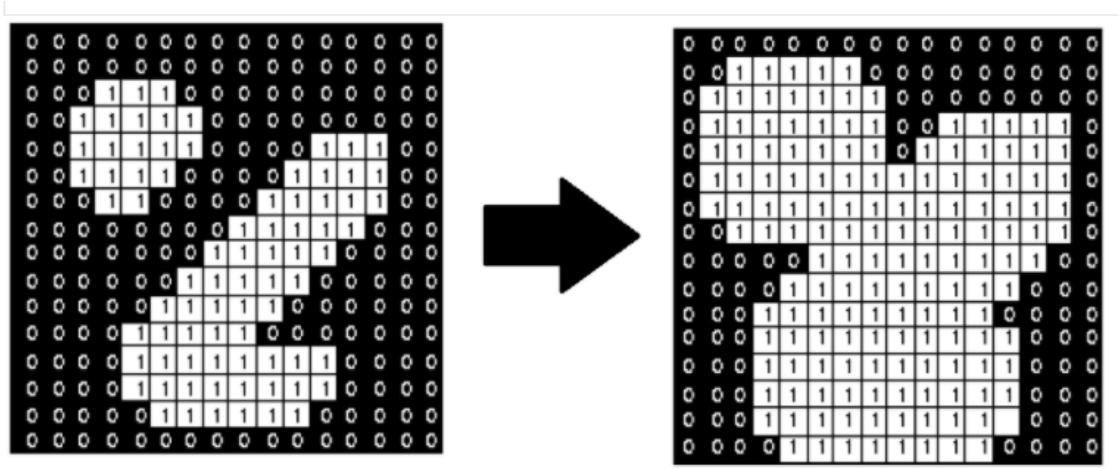
= { (1,2), (2,2), (3,2), (1,3), (2,3), (3,3),
(1,4), (2,4), (3,3), (2,2), (3,2), (4,2),
(2,3), (3,3), (4,3), (2,4), (3,4), (4,4),
(3,3), (4,3), (5,3) }

= { (1,2), (1,3), (1,4), (2,2), (2,3), (2,4),
(3,2), (3,3), (3,4), (4,2), (4,3), (4,4), (5,3) }



Gambar 3. Efek dilasi dengan hotspot vertical
(Sumber : Kadir, Abdul dkk, 2013)

Contoh operasi Dilasi yang lain dapat dilihat pada gambar 4.



Gambar 4. Contoh Efek Operasi Dilasi

(Sumber : https://devtrik.com/opencv/operasi-morfologi-pada-pengolahan-citra/)

C.3 Operasi Erosi

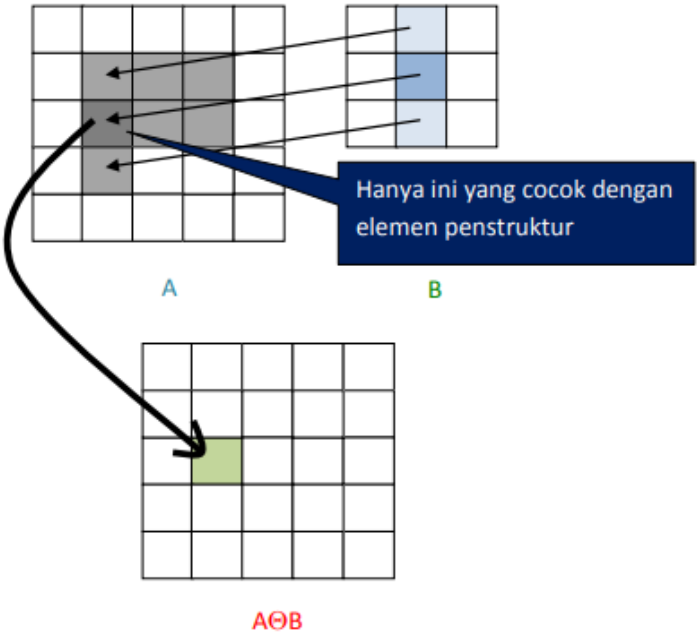
Operasi erosi merupakan salah satu wujud operasi morfologi yang berkebalikan dengan dilasi. Operasi ini malah membuat objek pada suatu citra menjadi lebih kecil atau tipis. Operasi ini dapat dilakukan dengan dua cara sebagai berikut :

1. Mengubah semua titik batas menjadi titik latar
2. Membuat semua titik di sekeliling titik latar menjadi titik latar

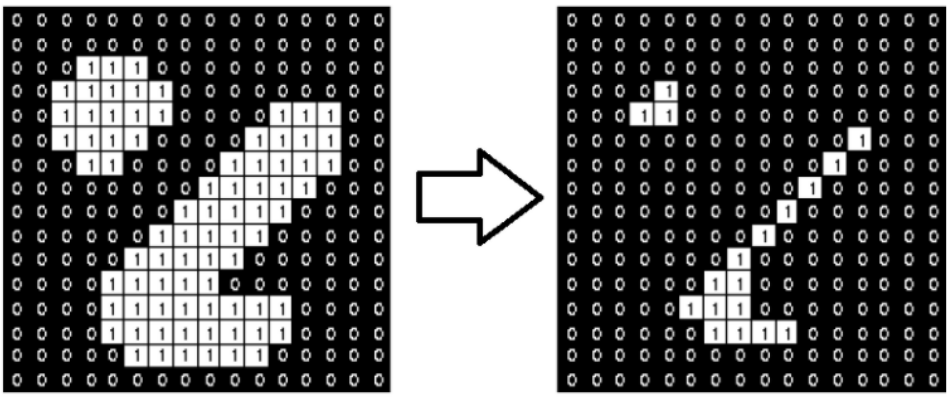
Operasi erosi mempunyai efek memperkecil struktur citra. Operasi ini dirumuskan seperti berikut (Gonzalez & Woods, 2002).

$$A \ominus B = \{z|(B)_z \subseteq A\}$$

Gambar 5 sebagai wujud operasi erosi pada suatu citra dengan mask linier vertikal. Gambar 6 merupakan contoh perubahan dari citra asli pada operasi erosi. Sedangkan gambar 7 merupakan contoh perubahan citra asli pada erosi dengan *structuring element image* berbentuk circular 11



Gambar 5 Contoh Operasi Erosi
(Sumber : Kadir, Abdul dkk, 2013)

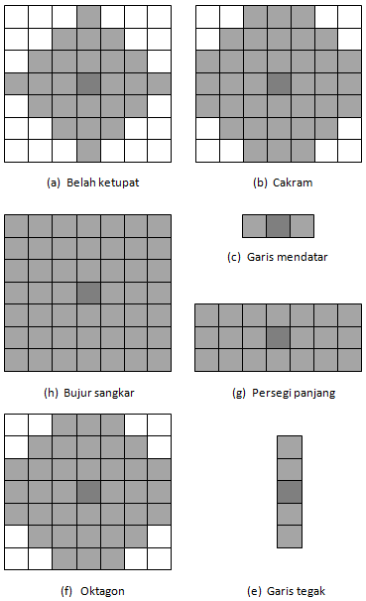


Gambar 6. Penerapan Operasi Erosi pada Citra
(Sumber : https://devtrik.com/opencv/operasi-morfologi-pada-pengolahan-citra/)

Gambar 7. Penerapan Operasi Erosi pada Citra dengan Mask Circle 11

C.4 *Structuring Element* (SE)

Dalam operasi morfologi, SE yang dapat digunakan sangat beragam, seperti yang ditunjukkan pada gambar 8. Berdasarkan berbagai bentuk SE tersebut yang paling sering digunakan adalah Cross, Rectangle, Line, dan Circle SE.



Gambar 8. Berbagai Bentuk SE
(Sumber : Kadir, Abdul dkk, 2013)

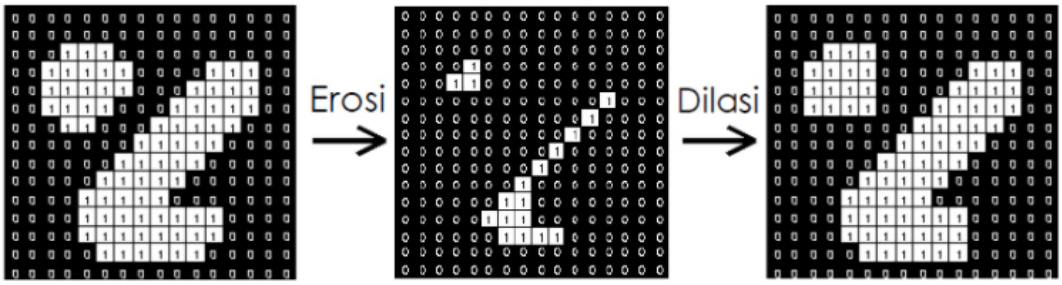
C.5 Operasi Opening

Operasi opening merupakan kombinasi antara operasi erosi dan dilasi yang dilakukan secara berurutan, tetapi citra asli dierosi terlebih dahulu baru kemudian hasilnya didilasi. Operasi ini memiliki tujuan untuk memutus bagian-bagian dari objek yang hanya terhubung dengan 1 atau 2 buah titik saja, atau menghilangkan objek-objek kecil tanpa mengubah area objek secara signifikan. Opening juga bersifat idempotent yaitu jika operasi opening diulang-ulang tidak berdampak berkelanjutan.

Operasi opening menggunakan persamaan sebagai berikut:

$$A \circ B = (A \ominus B) \oplus B$$

Operator *opening* dapat dimanfaatkan sebagai filter lolos-rendah, filter lolos-tinggi, maupun sebagai tapis lolos-bidang apabila elemen penstruktur yang digunakan berupa cakram (Shih, 2009). Gambar 9 merupakan wujud dari opening yang dimulai dari proses erosi dilanjutkan dengan proses dilasi.



Gambar 9. Operasi Opening pada Citra

Sumber : https://devtrik.com/opencv/operasi-morfologi-pada-pengolahan-citra/)

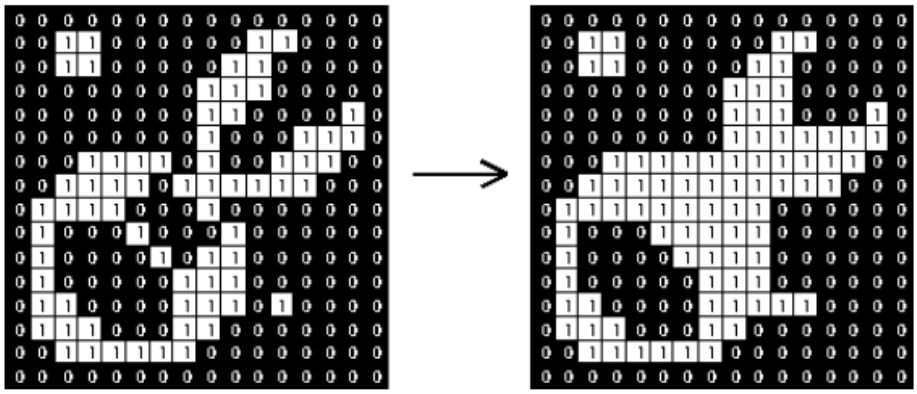
C.6 Operasi Closing

Operasi Closing adalah kombinasi antara operasi dilasi dan erosi yang dilakukan secara berurutan. Operasi closing memiliki tujuan menutup atau menghilangkan lubang-lubang kecil yang ada dalam segmen objek, serta menggabungkan objek yang berdekatan tanpa mengubah objek secara signifikan.

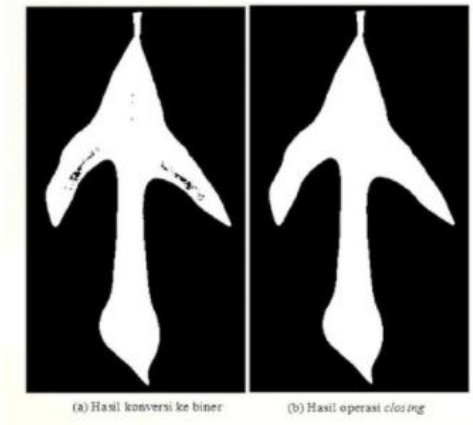
Operasi *closing* menggunakan persamaan sebagai berikut:

$$A \bullet B = (A \oplus B) \ominus B$$

Gambar 10, 11 merupakan wujud penerapan operasi closing yang diawali dengan proses dilasi terlebih dahulu, dan dilanjutkan dengan proses erosi.



Gambar 10. Operasi Closing pada Citra
Sumber : https://devtrik.com/opencv/operasi-morfologi-pada-pengolahan-citra/)



Gambar 11. Contoh lain Operasi Closing pada Citra
(Sumber : Kadir, Abdul dkk, 2013)

C7. Top Hat/White Hat Top Hat Transform, juga dikenal sebagai White Hat Transform, diperoleh dengan menghapus atau mengurangi Opening gambar dari gambar aslinya. Operator ini memberi fitur cerah pada gambar yang lebih kecil dari Structuring Element.

$$T_{hat}(f) = f - y(f)$$

C8. Black Hat/Bottom Hat Transformasi Black Hat, juga dikenal sebagai Transformasi Bottom Hat, diperoleh dengan menghapus atau mengurangi Closing gambar dari gambar aslinya. Operator ini memberi fitur gelap pada gambar yang lebih kecil dari Elemen Penataan.

$$B_{hat}(f) = B(f) - f$$

C9. Skeleton

Skeleton memberikan representasi simpel dari suatu objek dengan jumlah piksel yang kecil dan tetap mempertahankan karakteristik ukuran, posisi dan topologi dari bentuk aslinya.

Sehingga objek dari sebuah citra dapat dikenali dan diekstrak untuk proses pengolahan citra selanjutnya. Dengan kata lain, setelah sebagian besar titik pada obyek tersebut dihilangkan, maka pola dari obyek tersebut harus tetap dapat dikenali. Pola yang tertinggal ini disebut sebagai kerangka (skeleton), di mana sifat-sifatnya adalah :

a. Ketipisan: kerangka obyek berukuran setipis mungkin (1 atau 2 titik).
b. Konektivitas: kerangka dari suatu obyek terhubung satu sama lain sesuai dengan topologi pola aslinya.
c. Posisi: letak kerangka berada tepat di tengah obyek.
d. Stabilitas: setelah suatu bagian kerangka diperoleh, maka bagian tersebut tidak akan terkikis lagi oleh operasi pengikisan berikutnya.

Kebanyakan teknik untuk mengekstrak skeleton merupakan dasar dari operasi thinning. Metode skeletonisasi bertujuan untuk mendapatkan data banyaknya beserta panjang skeleton yang terbentuk. Hasil ini akan menjadi masukan dalam perhitungan intensitas rekahan (I) yaitu secara matematis ditulis sebagai berikut.

$$I = \frac{\sum_{i=1} li}{V}$$

Dengan l adalah panjang skeleton dan V adalah volume total citra.

C10. Thickening

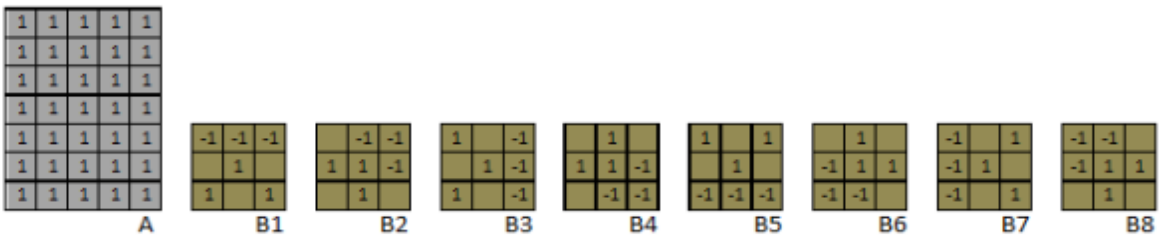
Penipisan himpunan A oleh strel B, yang dinyatakan dengan A □ B, dapat didefinisikan dalam transformasi hit-or-miss dengan bentuk:

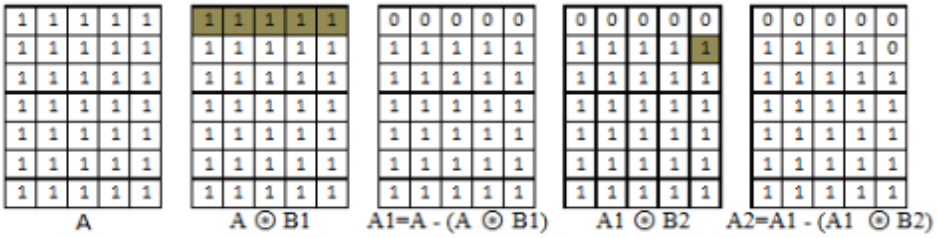
- $A \otimes B = A \cap (A \circledast B)^c$
- $A \otimes B = A - (A \circledast B)$

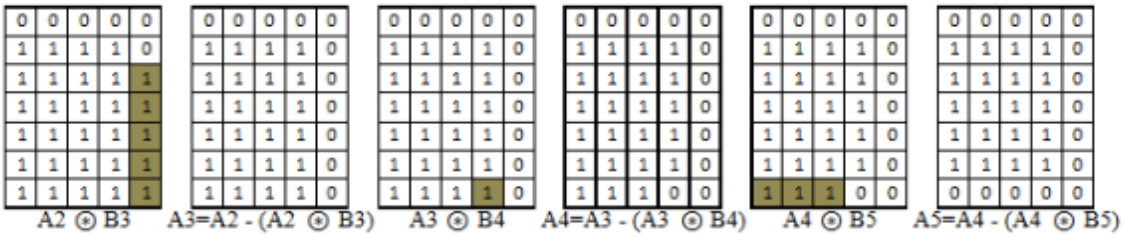
Prosesnya adalah:

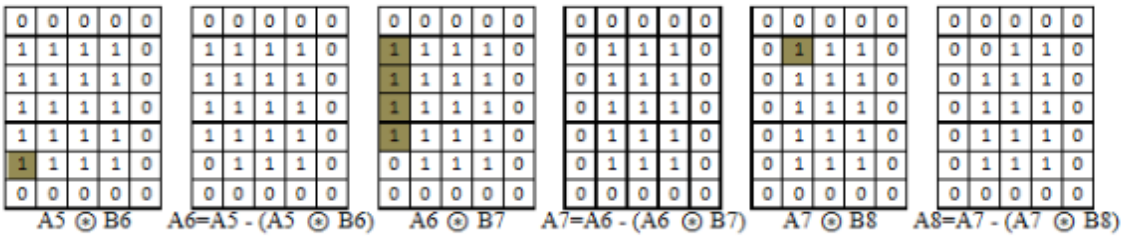
- Menipiskan A oleh satu lewatan dengan B1,
- kemudian menipiskan hasilnya dengan satu lewatan B2,
- dan seterusnya, sampai A ditipiskan dengan satu lewatan Bn.
- Semua proses ini diulang sampai tidak ada perubahan yang terjadi.

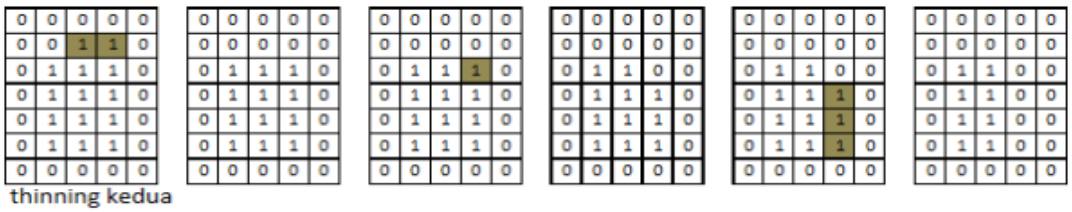
- Setiap penipisan dilewatkan dengan menggunakan persamaan A ⊗ B = A ∩ (A ⊛ B)c

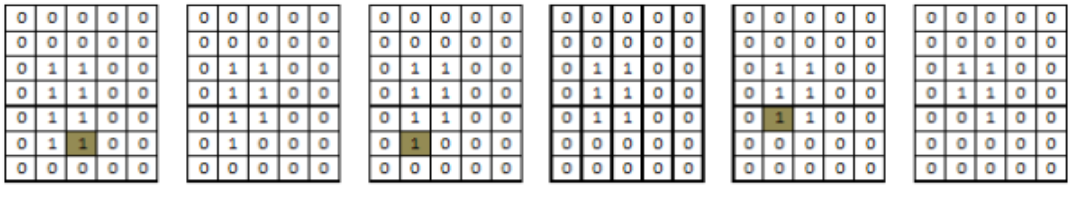


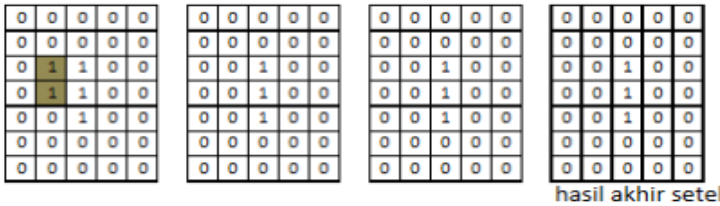












D. TUGAS PRAKTIKUM

1. Buka https://colab.research.google.com/. Setelah dipastikan bahwa google Colab terhubung dengan Github Anda, lanjutkan dengan memilih repository yang telah digunakan pada praktikum minggu lalu, rename file menjadi **"Week10_Absen.ipynb".**

   Kemudian import folder yang ada di Drive Anda dengan cara sebagai berikut.

```
from google.colab import drive
drive.mount('/content/drive')
```

2. Import beberapa library berikut yang akan digunakan selama uji coba praktikum minggu ke-6 berikut.

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

3. Buatlah fungsi dilasi serta tampilkan hasil dilasi pada citra dengan *Structuring Element* Dilasi (SED) berbentuk **cross 3 x 3 tanpa** menggunakan library *morphology* dari openCV, sehingga menghasilkan tampilan seperti di bawah ini:

```
#Dilasi tanpa library

def dilasi_citra(F,w):
  #size image
  p,q= F.shape

  imgD= np.zeros((p,q), dtype=np.uint8)

  #Generate structure element dilasi (SED)
  SED= np.ones((w,w), dtype=np.uint8)
  constant1= (w-1)//2

  #Proses Dilasi
  for i in range(constant1, p-constant1):
    for j in range(constant1,q-constant1):
      temp= F[i-constant1:i+constant1+1, j-
constant1:j+constant1+1]
      product= temp*SED
      imgD[i,j]= np.max(product)
  return imgD

img_d= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0)
ret, thresh = cv2.threshold(img_d, 125, 200, cv2.THRESH_BINARY)
w=5
```

```
imgD = dilasi_citra(thresh,w)

plt.subplot(131),plt.imshow(img_d,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgD,cmap = 'gray')
plt.title('Citra Hasil Dilasi'), plt.xticks([]), plt.yticks([])
plt.show()
```

4. Buatlah program implementasi operasi dilasi *Structuring Element* Dilasi (SED) berbentuk **cross 3 x 3 <u>dengan</u>** menggunakan library *morphology* dari openCV, sehingga menghasilkan tampilan seperti di bawah ini:

```
img = cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg')
#img = cv2.imread('morp.jpg',0)
ret, thresh = cv2.threshold(img, 127, 225, cv2.THRESH_BINARY)
kernel = np.ones((3,3),np.uint16)
dilasi = cv2.dilate(thresh,kernel,iterations = 1)
print(kernel)

plt.subplot(121),plt.imshow(img),plt.title('Citra Awal')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(dilasi),plt.title('Citra Hasil
Dilasi')
plt.xticks([]), plt.yticks([])
plt.show()
```

5. Buatlah fungsi erosi serta tampilkan hasil erosi pada citra dengan *Structuring Element* Erosi (SE) berbentuk square 5 x 5 **<u>tanpa</u>** menggunakan library *morphology* dari openCV, sehingga menghasilkan tampilan seperti di bawah ini:

```
#Erosi Tanpa Library
```

```
def erosi_citra(F,k):
  #size image
  m,n= F.shape

  #Generate structure element erosi (SE)
  SE= np.ones((k,k), dtype=np.uint8)
  constant= (k-1)//2

  imgE= np.zeros((m,n), dtype=np.uint8)

  #Proses Erosi
  for i in range(constant, m-constant):
    for j in range(constant,n-constant):
      temp= F[i-constant:i+constant+1, j-
constant:j+constant+1]
      product= temp*SE
      imgE[i,j]= np.min(product)

  return imgE

img_e= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0) #
ret1, thresh1 = cv2.threshold(img_e, 175, 225,
cv2.THRESH_BINARY)
k=5
imgE = erosi_citra(thresh1,k)

plt.subplot(131),plt.imshow(img_e,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgE,cmap = 'gray')
plt.title('Citra Hasil Erosi'), plt.xticks([]),
plt.yticks([])
plt.show()
```

6. Buatlah program implementasi operasi dilasi *Structuring Element* Erosi (SE) berbentuk square 5 x 5 **dengan** menggunakan library *morphology* dari openCV, sehingga menghasilkan tampilan seperti di bawah ini:

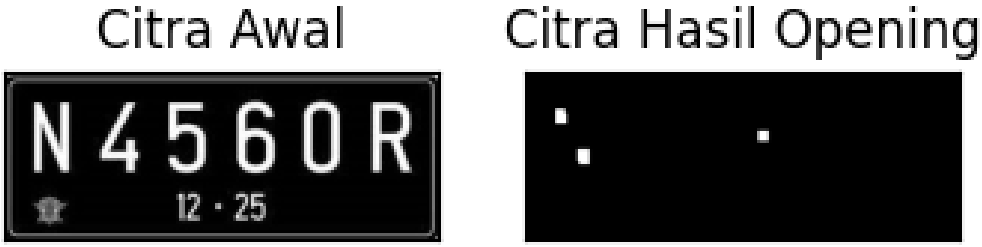
```
#EROSI LIBRARY
```

```
img = cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg')

kernel = np.ones((5,5),np.uint8)
erosion = cv2.erode(img,kernel,iterations = 1)
plt.subplot(121),plt.imshow(img),plt.title('Citra Awal')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(erosion),plt.title('Citra Hasil
Erosi')
plt.xticks([]), plt.yticks([])
plt.show()
```

7. Buatlah program implementasi operasi opening dengan *Structuring Element* berbentuk square 3 x 3 **tanpa dan <u>dengan</u>** menggunakan library *morphology* dari openCV, sehingga menghasilkan tampilan seperti di bawah ini:

Operasi Opening Menggunakan tanpa Library Morphology

```
img_o= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0) #
reto, thresho = cv2.threshold(img_o, 127, 225,
cv2.THRESH_BINARY)
k=7
imgO = dilasi_citra(erosi_citra(thresho,k),k)

plt.subplot(131),plt.imshow(img_o,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgO,cmap = 'gray')
plt.title('Citra Hasil Opening'), plt.xticks([]), plt.yticks([])
plt.show()
```



Operasi Opening Menggunakan Library Morphology

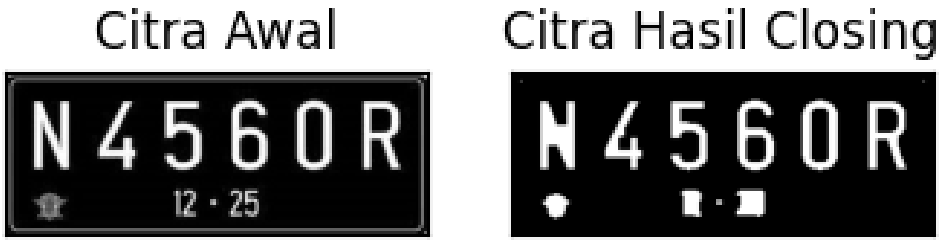
```
#Dengan Menggunakan Library
```

```
img = cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0)
ret, thresh = cv2.threshold(img, 127, 225, cv2.THRESH_BINARY)
kernel = np.ones((3,3),np.uint8)
openn = cv2.morphologyEx(thresh,cv2.MORPH_OPEN, kernel)

plt.subplot(131),plt.imshow(img,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(thresh,cmap = 'gray')
plt.title('Citra Biner'), plt.xticks([]), plt.yticks([])
plt.subplot(133),plt.imshow(openn,cmap = 'gray')
plt.title('Citra Hasil Opening'), plt.xticks([]),
plt.yticks([])

plt.show()
```

8. Buatlah program implementasi operasi closing dengan *Structuring Element* berbentuk square 3 x 3 **tanpa dan <u>dengan</u>** menggunakan library *morphology* dari openCV, sehingga menghasilkan tampilan seperti di bawah ini:

operasi closing tanpa library morphology

```
img_c= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0) #
retc, threshc = cv2.threshold(img_c, 127, 225,
cv2.THRESH_BINARY)
k=7
imgC = erosi_citra(dilasi_citra(threshc,k),k)

plt.subplot(131),plt.imshow(img_c,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgC,cmap = 'gray')
plt.title('Citra Hasil Closing'), plt.xticks([]),
plt.yticks([])
plt.show()
```



operasi closing dengan library morphology

```
img = cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0)
ret, thresh = cv2.threshold(img, 127, 225, cv2.THRESH_BINARY)
kernel = np.ones((3,3),np.uint8)
closs = cv2.morphologyEx(thresh,cv2.MORPH_CLOSE, kernel)

plt.subplot(131),plt.imshow(img,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(thresh,cmap = 'gray')
plt.title('Citra Biner'), plt.xticks([]), plt.yticks([])
plt.subplot(133),plt.imshow(closs,cmap = 'gray')
plt.title('Citra Hasil Closing'), plt.xticks([]), plt.yticks([])

plt.show()
```

9. Buatlah program implementasi operasi dilasi dengan *Structuring Element* berbentuk **Cross 3 x 3 dan 5 x 5**, sehingga menghasilkan tampilan seperti di bawah ini:

```
#Dilasi Cross SED

def dilasi_citra(F,w):
  #size image
  p,q= F.shape

  imgD= np.zeros((p,q), dtype=np.uint8)
  #Generate structure element dilasi (SED)
  # SED= np.ones((w,w), dtype=np.uint8)
  SED = np.array([[0, 1, 0],
                  [1, 1, 1],
                  [0, 1, 0]], dtype = np.uint8)
  constant1= (w-1)//2

  #Proses Dilasi
```

```
  for i in range(constant1, p-constant1):
    for j in range(constant1,q-constant1):
      temp= F[i-constant1:i+constant1+1, j-
constant1:j+constant1+1]
      product= temp*SED
      imgD[i,j]= np.max(product)
  return imgD
print('SED = Cross 3')
img_d= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0)
ret, thresh = cv2.threshold(img_d, 127, 225,
cv2.THRESH_BINARY)
w=3
imgD = dilasi_citra(thresh,w)

plt.subplot(131),plt.imshow(img_d,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgD,cmap = 'gray')
plt.title('Citra Hasil Dilasi'), plt.xticks([]),
plt.yticks([])
plt.show()


def dilasi_citra(F,w):
  #size image
  p,q= F.shape

  imgD= np.zeros((p,q), dtype=np.uint8)

  #Generate structure element dilasi (SED)
  # SED= np.ones((w,w), dtype=np.uint8)
  SED = np.array([[0, 0, 1, 0, 0],
                  [0, 0, 1, 0, 0],
                  [1, 1, 1, 1, 1],
                  [0, 0, 1, 0, 0],
                  [0, 0, 1, 0, 0]], dtype = np.uint8)
  constant1= (w-1)//2

  #Proses Dilasi
  for i in range(constant1, p-constant1):
    for j in range(constant1,q-constant1):
      temp= F[i-constant1:i+constant1+1, j-
constant1:j+constant1+1]
      product= temp*SED
      imgD[i,j]= np.max(product)
  return imgD
print('SED = Cross 5')
img_d= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0)
```

```
ret, thresh = cv2.threshold(img_d, 127, 225,
cv2.THRESH_BINARY)
w=5
imgD = dilasi_citra(thresh,w)

plt.subplot(131),plt.imshow(img_d,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgD,cmap = 'gray')
plt.title('Citra Hasil Dilasi'), plt.xticks([]),
plt.yticks([])
plt.show()
```

Cross 3

Cross 5

10. Buatlah program implementasi operasi dilasi dengan *Structuring Element* berbentuk **Circular 3 x 3 dan 5 x 5**, sehingga menghasilkan tampilan seperti di bawah ini:

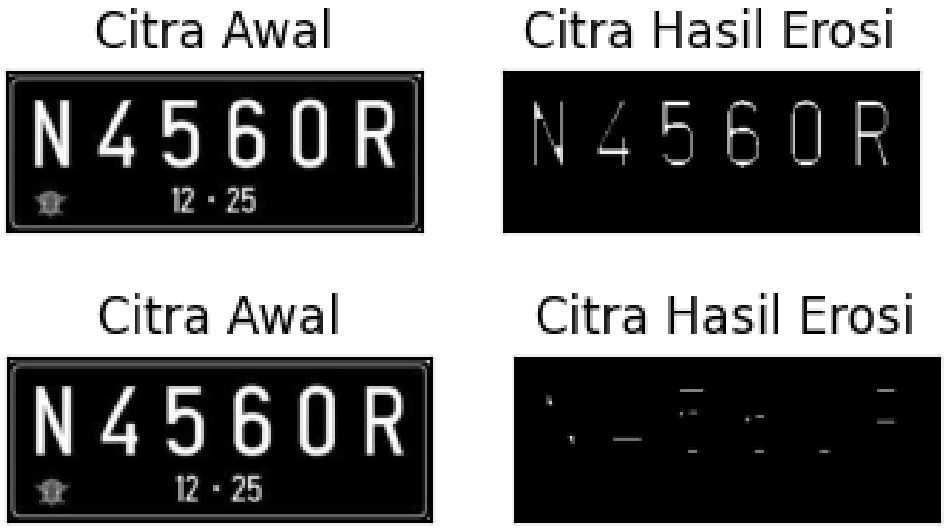
```
#Dilasi Circular SED

def dilasi_citra(F,w):
  #size image
  p,q= F.shape

  imgD= np.zeros((p,q), dtype=np.uint8)

  #Generate structure element dilasi (SED)
  SED= cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(w,w))
  constant1= (w-1)//2

  #Proses Dilasi
  for i in range(constant1, p-constant1):
    for j in range(constant1,q-constant1):
      temp= F[i-constant1:i+constant1+1, j-
constant1:j+constant1+1]
```

```
            product= temp*SED
            imgD[i,j]= np.max(product)
      return imgD
print('SED = Circle 3')
img_d= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0)
ret, thresh = cv2.threshold(img_d, 127, 225,
cv2.THRESH_BINARY)
w=3
imgD = dilasi_citra(thresh,w)

plt.subplot(131),plt.imshow(img_d,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgD,cmap = 'gray')
plt.title('Citra Hasil Dilasi'), plt.xticks([]),
plt.yticks([])
plt.show()

print('SED = Circle 5')
img_d= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0)
ret, thresh = cv2.threshold(img_d, 127, 225,
cv2.THRESH_BINARY)
w=5
imgD = dilasi_citra(thresh,w)

plt.subplot(131),plt.imshow(img_d,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgD,cmap = 'gray')
plt.title('Citra Hasil Dilasi'), plt.xticks([]),
plt.yticks([])
plt.show()
```

11. Buatlah program implementasi operasi erosi dengan *Structuring Element* berbentuk **Rectangle 3 x 5 dan 5 x 7**, sehingga menghasilkan tampilan seperti di bawah ini:

```
#Erosi Cross SE

def erosi_citra(F,p,l):
  #size image
  m,n= F.shape

  #Generate structure element erosi (SE)
  SE = cv2.getStructuringElement(cv2.MORPH_RECT,(p,l)) #SE persegi
panjang
  constant1= (l-1)//2
  constant2= (p-1)//2
  # print(SE)
  imgE= np.zeros((m,n), dtype=np.uint8)

  #Proses Erosi
  for i in range(constant1, m-constant1):
    for j in range(constant2,n-constant2):
      temp= F[i-constant1:i+constant1+1, j-constant2:j+constant2+1]
      product= temp*SE
      imgE[i,j]= np.min(product)

  return imgE
print('SE = Rectangle 3x5')
img_e= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0) #
ret1, thresh1 = cv2.threshold(img_e, 127, 225, cv2.THRESH_BINARY)
p=5
l=3
imgE = erosi_citra(thresh1,p,l)

plt.subplot(131),plt.imshow(img_e,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgE,cmap = 'gray')
plt.title('Citra Hasil Erosi'), plt.xticks([]), plt.yticks([])
plt.show()

print('SE = Rectangle 5x7')
img_e= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0) #
ret1, thresh1 = cv2.threshold(img_e, 127, 225, cv2.THRESH_BINARY)
p=7
l=5
imgE = erosi_citra(thresh1,p,l)

plt.subplot(131),plt.imshow(img_e,cmap = 'gray')
```

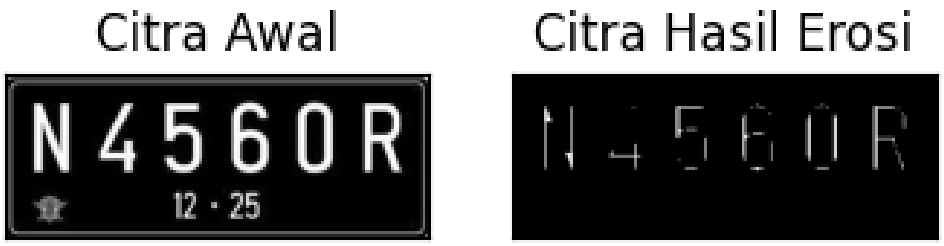
```
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgE,cmap = 'gray')
plt.title('Citra Hasil Erosi'), plt.xticks([]), plt.yticks([])
plt.show()
```



12. Buatlah program implementasi operasi erosi dengan *Structuring Element* berbentuk **Line Vertikal 3 dan 5**, sehingga menghasilkan tampilan seperti di bawah ini: (Gunakan gambar j.png)

```
#Erosi Line Vertikal SE

def erosi_citra(F,k):
  #size image
  m,n= F.shape

  #Generate structure element erosi (SE)
  SE = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(1,k)) #SE
line vertikal
  constant= (k-1)//2

  imgE= np.zeros((m,n), dtype=np.uint8)

  #Proses Erosi
  for i in range(constant, m-constant):
    for j in range(constant,n-constant):
      temp= F[i-constant:i+constant+1, j-
constant:j+constant+1]
      product= temp*SE
      imgE[i,j]= np.min(product)

  return imgE
print('SE = Line Vertikal 3')
```

```
img_e= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0) #
ret1, thresh1 = cv2.threshold(img_e, 127, 225,
cv2.THRESH_BINARY)
k=3
imgE = erosi_citra(thresh1,k)

plt.subplot(131),plt.imshow(img_e,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgE,cmap = 'gray')
plt.title('Citra Hasil Erosi'), plt.xticks([]),
plt.yticks([])
plt.show()

print('SE = Line Vertikal 5')
img_e= cv2.imread('/content/drive/MyDrive/PCVK/Images/plat
nomer.jpg',0) #
ret1, thresh1 = cv2.threshold(img_e, 127, 225,
cv2.THRESH_BINARY)
k=5
imgE = erosi_citra(thresh1,k)

plt.subplot(131),plt.imshow(img_e,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(imgE,cmap = 'gray')
plt.title('Citra Hasil Erosi'), plt.xticks([]),
plt.yticks([])
plt.show()
```



13 TopHat

```
# Getting the kernel to be used in Top-Hat
filterSize =(3, 3)
kernel = cv2.getStructuringElement(cv2.MORPH_RECT,
                   filterSize)

# Reading the image named 'input.jpg'
img = cv2.imread('/content/drive/MyDrive/PCVK/Images/plat nomer.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Applying the Top-Hat operation
tophat_img = cv2.morphologyEx(input_image,
                 cv2.MORPH_TOPHAT,
                 kernel)

#cv2.imshow("original", input_image)
#cv2.imshow("tophat", tophat_img)
#cv2.waitKey(5000)
plt.subplot(131),plt.imshow(img,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(tophat_img,cmap = 'gray')
plt.title('Citra Hasil TOP HAT'), plt.xticks([]), plt.yticks([])
plt.show()
```

14 BLACKHAT

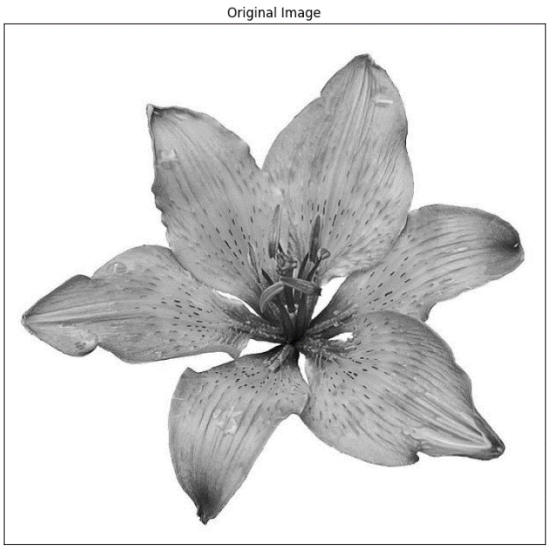
```
filterSize =(3, 3)
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, filterSize)

# Reading the image named 'input.jpg'
img = cv2.imread('/content/drive/MyDrive/PCVK/Images/kitten01.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Applying the Black-Hat operation
#tophat_img = cv2.morphologyEx(img,cv2.MORPH_BLACKHAT, kernel)
Kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (13, 5))
blackhat = cv2.morphologyEx(img, cv2.MORPH_BLACKHAT, Kernel)
```

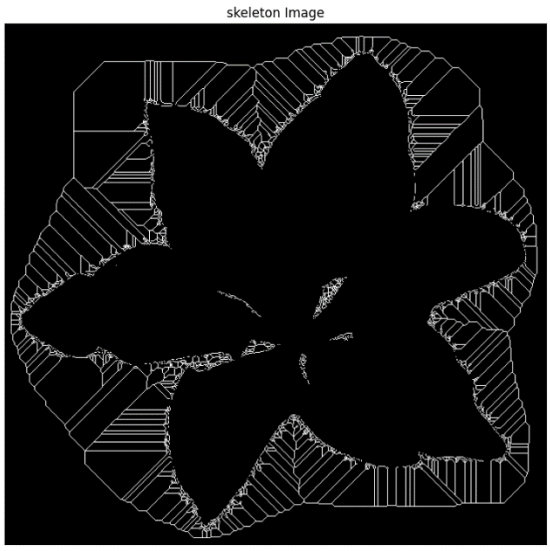
```
#cv2.imshow("original", input_image)
#cv2.imshow("tophat", tophat_img)
#cv2.waitKey(5000)
plt.subplot(131),plt.imshow(img,cmap = 'gray')
plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(blackhat,cmap = 'gray')
plt.title('Citra Hasil Black Hat'), plt.xticks([]), plt.yticks([])
plt.show()
```

15. Skeleton

```
import cv2 as cv
from matplotlib import pyplot as plt
from skimage import filters
from skimage.morphology import skeletonize

# read image
img = cv.imread('/content/drive/MyDrive/PCVK/Images/lily.jpg',0)

binary = img > filters.threshold_triangle(img)

# true false to one
binary_cp = binary.copy()
binary_cp[binary_cp == True] = 1
binary_cp[binary_cp == False] = 0

# skeletonize image
skeleton = skeletonize(binary_cp)

# print images
plt.figure(figsize=(20,20))

plt.subplot(121),plt.imshow(img,cmap = 'gray')
plt.title('Original Image'), plt.xticks([]), plt.yticks([])

plt.subplot(122),plt.imshow(skeleton,cmap = 'gray')
plt.title('skeleton Image'), plt.xticks([]), plt.yticks([])
```

```
plt.savefig("binary and bin_skeleton.png")
plt.show()
```

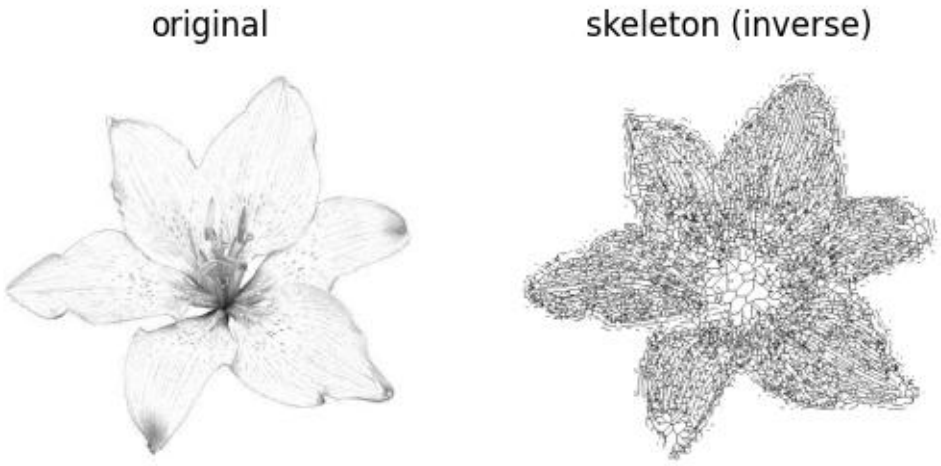




16. Skeleton Inverse

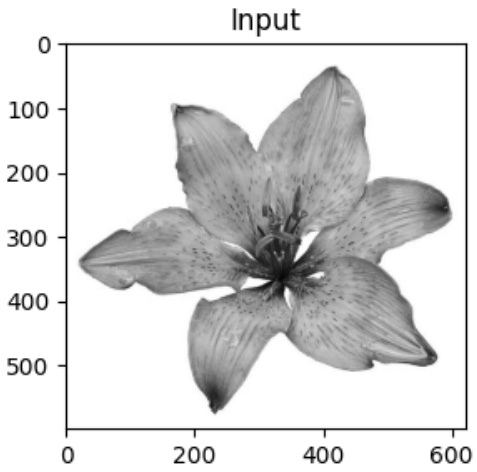
```
from skimage import io
from matplotlib import pyplot as plt
from skimage import filters
from skimage.morphology import skeletonize

# read image
img = io.imread('/content/drive/MyDrive/PCVK/Images/lily.jpg')[...,
0]

# Note: we want the black bits to be True, so use <
binary = img < filters.threshold_triangle(img)

# skeletonize image
skeleton = skeletonize(binary)

# print images
fig, ax = plt.subplots(1, 2)

ax[0].imshow(img, cmap='gray')
ax[0].set_title('original')
ax[0].set_axis_off()

# note the reversed colormap, gray_r
ax[1].imshow(skeleton, cmap='gray_r')
ax[1].set_title('skeleton (inverse)')
ax[1].set_axis_off()
```

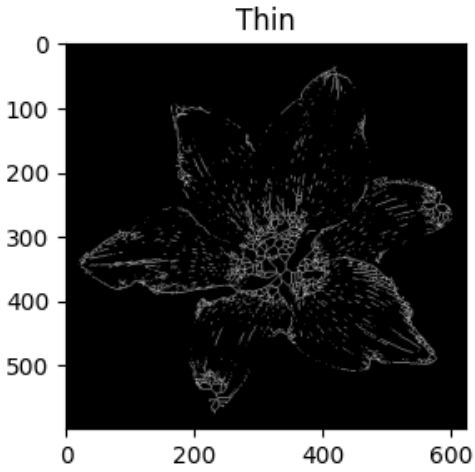
```
plt.show()
```



17. Thickening

```
from skimage import img_as_float
from skimage import io, color, morphology
import matplotlib.pyplot as plt

image =
img_as_float(color.rgb2gray(io.imread('/content/drive/MyDrive/PCVK/Images/lily.jpg')))
image_binary = image < 0.5
out_skeletonize = morphology.skeletonize(image_binary)
out_thin = morphology.thin(image_binary)


f, (ax0, ax1) = plt.subplots(1, 2, figsize=(10, 3))

ax0.imshow(image, cmap='gray')
ax0.set_title('Input')


ax1.imshow(out_thin, cmap='gray')
ax1.set_title('Thin')

plt.savefig('/tmp/char_out.png')
plt.show()
```





--- SELAMAT BELAJAR ---