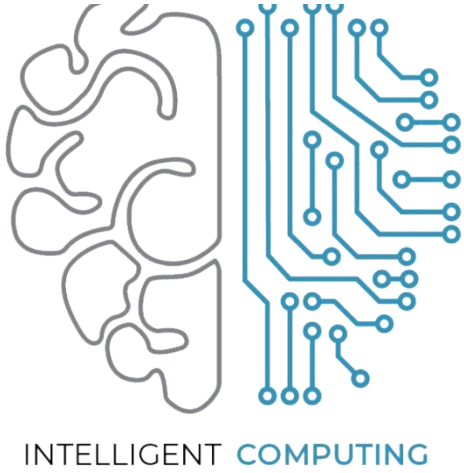
# LAPORAN PRAKTIKUM

# PENGOLAHAN CITRA DIGITAL




NAMA : Dimas Damarjati

NIM : 202231020

KELAS : B

DOSEN : Dr. Dra. Dwina Kuswardani, M.Kom

NO.PC : 13

ASISTEN : 1. Fachreza Riyanda.

2.

## INSTITUT TEKNOLOGI PLN

## TEKNIK INFORMATIKA

## 2024

- Latar belakang:

  Telah dilakukannya praktikum 3 pada mata kuliah pengolahan citra digital yang dilaksanakan pada rabu 5 juni 2024.

- Tujuan:

  Membuat laporan untuk praktikum 3 sesuai arahan yang telah diberikan.

- Alat dan bahan:

  -Jupyter Notebook

  -Library OpenCV

  -Library Skimage

- Pembahasan:

```
import cv2
import numpy as np
import matplotlib.plot as plt
%matplotlib inline
import skimage
```

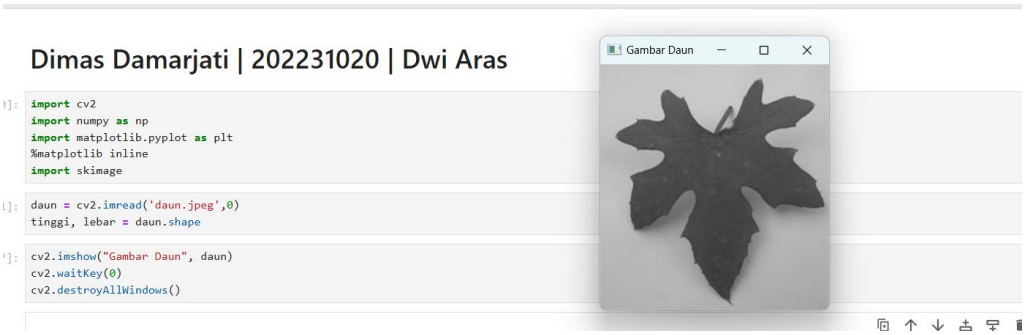
  - cv2 adalah modul OpenCV untuk pemrosesan gambar.
  - numpy (np) adalah modul untuk operasi numerik dan array.
  - matplotlib.pyplot (plt) adalah modul untuk plotting grafis.
  - %matplotlib inline agar hasil plot ditampilkan di dalam notebook.
  - skimage untuk pemrosesan gambar tambahan dari scikit-image.

```
daun = cv2.imread('daun.jpg',0)
tinggi.lebar = daun.shape
```

  - Membaca gambar daun.jpg dalam mode grayscale (0 menunjukkan mode grayscale).
  - Mendapatkan dimensi gambar (tinggi dan lebar) dari gambar yang dibaca.

```
cv2.imshow("Gambar Daun", daun)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- Menampilkan jendela gambar asli dengan judul "Gambar Daun".




- cv2.waitKey(0) menunggu penekanan tombol untuk menutup jendela.
- cv2.destroyAllWindows() menutup semua jendela.

```
nilai_ambang = 131
daun_hasil = daun.copy()
for x in range (tinggi):
    for y in range (lebar):
        if daun[x,y] < nilai_ambang:
            daun_hasil[x,y] = 0
        else:
            daun_hasil[x,y] = 255
```

- Mendefinisikan nilai ambang batas (nilai_ambang = 131).
- Membuat salinan dari gambar asli (daun_hasil = daun.copy()).
- Melakukan iterasi melalui setiap piksel pada gambar:
- Jika nilai piksel kurang dari nilai ambang, set piksel tersebut menjadi hitam (0).
- Jika nilai piksel lebih besar atau sama dengan nilai ambang, set piksel tersebut menjadi putih (255).

```
cv2.imshow("Gambar Daun", daun_hasil)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- Menampilkan jendela gambar asli dengan judul "Gambar Daun".

```
[72]: cv2.imshow("Gambar Daun", daun)
      cv2.waitKey(0)
      cv2.destroyAllWindows()
```


memperkirakan nilai amba

```
[73]: nilai_ambang = 131
      daun_hasil = daun.copy()
      for x in range (tinggi):
          for y in range (lebar):
              if daun[x,y] < nilai_ambang:
                  daun_hasil[x,y] = 0
              else:
                  daun_hasil[x,y] = 255
```

- cv2.waitKey(0) menunggu penekanan tombol untuk menutup jendela.
- cv2.destroyAllWindows() menutup semua jendela.

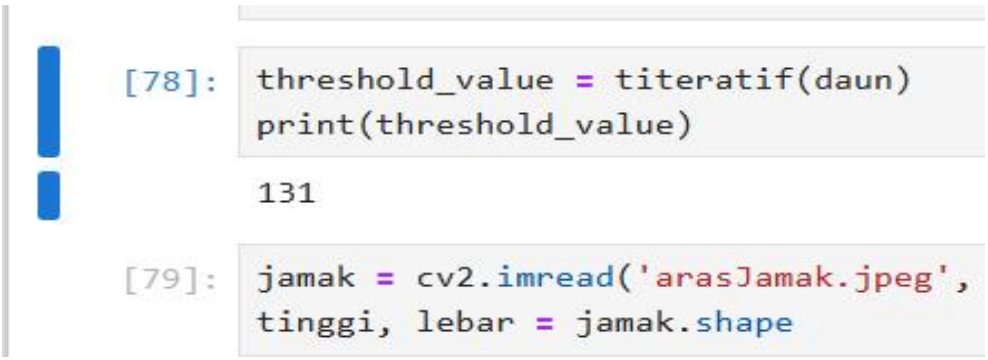
```
def titeratif(image):
    tinggi,lebar = image.shape
    t0 = 127
    while(True):
        rata_kiri=0;
        rata_kanan=0;
        jum_kiri=0;
        jum_kanan=0;
        for x in range(tinggi):
            for y in range(lebar):
                if (image[x,y] <= 127)
                    rata_kiri = rata_kiri + image[x,y]
                    jum_kiri = jum_kiri + 1
                else:
                    rata_kanan = rata_kanan + image[x,y]
```

```
            jum_kanan = jum_kanan + 1
        rata_kiri = rata_kiri/jum_kiri
        rata_kanan = rata_kanan/jum_kanan
        t1 = (rata_kiri + rata_kanan)/2
        if((to-t1)<1):
            break
    return round(t1)
```

- Mendefinisikan fungsi titeratif untuk menghitung nilai ambang threshold secara iteratif:
- Inisialisasi nilai threshold awal (t0 = 127).
- Loop hingga perbedaan antara threshold lama dan baru kurang dari 1.
- Menghitung rata-rata nilai piksel untuk di bawah dan di atas threshold.
- Menghitung threshold baru (t1) sebagai rata-rata dari dua rata-rata.
- Jika threshold konvergen, keluar dari loop dan kembalikan nilai threshold.

```
threshold_value = titeratif(daun)
print(threshold_value)
```

- Menjalankan fungsi titeratif pada gambar daun.




```
cv2.imread('arasJamak.png', 0)
tinggi, lebar = jamak.shape
```

- Membaca gambar arasJamak.png dalam mode grayscale dimana angka 0 menunjukkan mode grayscale.
- Mendapatkan dimensi gambar (tinggi dan lebar) dari gambar yang dibaca.

```
def arasjamak(image,t1,t2):
  res=image
  m,n=image.shape(m):
  for x in range(m):
    for y in range (n):
      if (image[x,y] <= t1) or (image[x,y]>= t2):
        res[x,y]= 0
      else :
        res[x,y] = 255
  return res
```

- Mendefinisikan fungsi arasjamak untuk melakukan thresholding dengan dua nilai ambang (t1 dan t2):
- Membuat salinan dari gambar asli (res = image.copy()).
- Melakukan iterasi melalui setiap piksel pada gambar:
- Jika nilai piksel kurang dari t1 atau lebih besar dari t2, set piksel menjadi hitam (0).
- Jika nilai piksel di antara t1 dan t2, set piksel menjadi putih (255).

```
shapearasJamak = arasjamak(jamak, 185, 200)
```

- Menjalankan fungsi arasjamak dengan nilai ambang 185 dan 200.

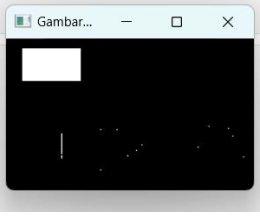
```
cv2.imshow("Gambar Shape", shapearasjamak)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- Menampilkan jendela gambar hasil thresholding dengan judul "Gambar Shape".

```
print(threshold_value)
131
```

```
[92]: jamak = cv2.imread('arasJamak.jpeg', 0)
      tinggi, lebar = jamak.shape
```

```
[93]: def arasjamak(image, t1, t2):
          res = image.copy()
          m, n = image.shape
          for x in range(m):
              for y in range(n):
                  if image[x, y] <= t1 or image[x, y] >= t2:
                      res[x, y] = 0
                  else:
```




- cv2.waitKey(0) menunggu penekanan tombol untuk menutup jendela.
- cv2.destroyAllWindows() menutup semua jendela.

```
fig,axs = plt.subplots(1,2, figsize = (10,10))
ax = axs.ravel()
ax[0].imshow(gray, cmap = "gray"
ax[0].set_title("Gambar Asli")
ax[1].imshow(edges, cmap = "gray"
ax[1].set_title("Gambar setelah diolah")
```

- Membuat subplots dengan 1 baris dan 2 kolom menggunakan plt.subplots dan mengatur ukuran gambar menjadi 10x10 inci.
- ax = axs.ravel() meratakan array axs menjadi satu dimensi.
- Menampilkan gambar asli (gray) pada subplot pertama (ax[0]) dengan colormap grayscale (cmap="gray") dan memberi judul "Gambar Asli".
- Menampilkan gambar hasil pengolahan (edges) pada subplot kedua (ax[1]) dengan colormap grayscale (cmap="gray") dan memberi judul "Gambar setelah diolah".

```
line = cv2.HoughLinesP(edges, 1,np.pi/180, 30, maxLineGap = 250)
image_line = image.copy()
for line in lines:
```

```
x1, y1, x2, y2 = line[0]

cv2.line(image_line, (x1,y1) (x2,y2),(100, 150))
```

- cv2.HoughLinesP untuk mendeteksi garis pada gambar tepi (edges).
- Membuat salinan dari gambar asli.
- Jika garis terdeteksi, iterasi melalui semua garis yang terdeteksi.
- line[0] ekstrak koordinat titik awal (x1, y1) dan titik akhir (x2, y2) dari garis.
- cv2.line(image_line, (x1, y1), (x2, y2), (255, 0, 0), 2) gambar garis pada salinan (image_line) dengan warna merah (255, 0, 0) dan ketebalan 2 piksel.

```
fig, axs = plt.subplots(1, 2, figsize=(10, 10))

ax = axs.ravel()

ax[0].imshow(gray, cmap="gray")

ax[0].set_title("Gambar Asli")

ax[1].imshow(edges, cmap="gray")

ax[1].set_title("Gambar setelah diolah")
```

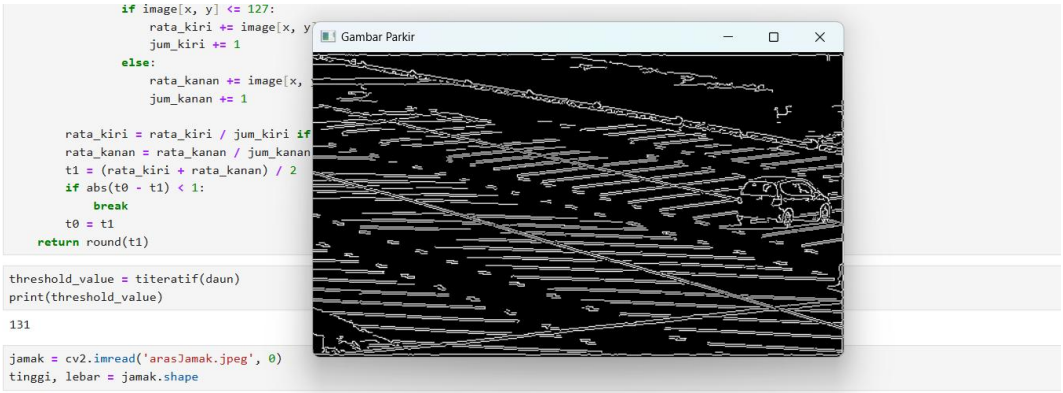
- plt.subplots(1, 2, figsize=(10, 10)) membuat figure dan array dari axs.
- axs.ravel() meratakan array menjadi array 1D.
- ax[0].imshow(gray, cmap="gray") menampilkan gambar gray pada subplot pertama (ax[0]) dengan colormap grayscale (cmap="gray").
- ax[0].set_title("Gambar Asli") memberi judul "Gambar Asli".
- ax[1].imshow(edges, cmap="gray") menampilkan gambar edges pada subplot kedua (ax[1]) dengan colormap grayscale (cmap="gray").
- ax[1].set_title("Gambar setelah diolah") memberi judul "Gambar setelah diolah" pada subplot kedua.

***cv2.imshow("Gambar Parkir", edges)***

***cv2.waitKey(0)***

***cv2.destroyAllWindows()***

- Menampilkan jendela yang menunjukkan gambar hasil thresholding dengan judul "Gambar Parkir".



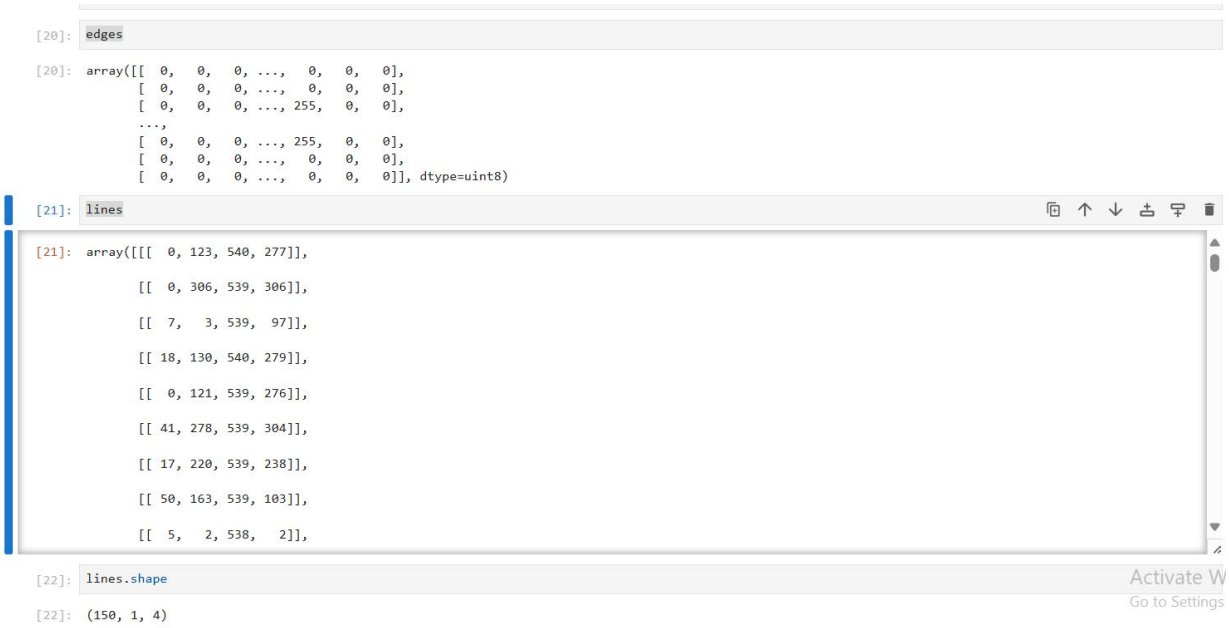

- cv2.waitKey(0) menunggu penekanan tombol untuk menutup jendela.
- cv2.destroyAllWindows() menutup semua jendela.

***edges***

***lines***

***lines.shape***

- Mengembalikan nilai edges, lines, dan lines.shape untuk melihat hasil deteksi tepi dan garis serta bentuk array garis.



- **Kesimpulan:**

Kesimpulan yang dapat diambil dalam praktikum ini yaitu praktikum kali ini membahas tentang pemrosesan gambar menggunakan library OpenCV dengan jupyer notebook dan divisualisasikan menggunakan Matplotlib. Penggambarannya melibatkan teknik thresholding, deteksi tepi, dan deteksi garis yang membantu memahami efek dari setiap langkah pemrosesan gambar, sehingga memungkinkan analisis dan perbaikan yang lebih baik untuk target gambarnya. Contohnya thresholding gambar "Gambar Shape" yang mengubah gambar grayscale menjadi gambar biner dengan threshold 185 hingga 200. Gambar menunjukkan piksel bagian porsi kiri atas putih sendiri yang dimana itu menunjukkan bagian tersebut thresholdnya diantara 185 - 200 sedangkan sisanya yang hitam berada diluar batas threshold.