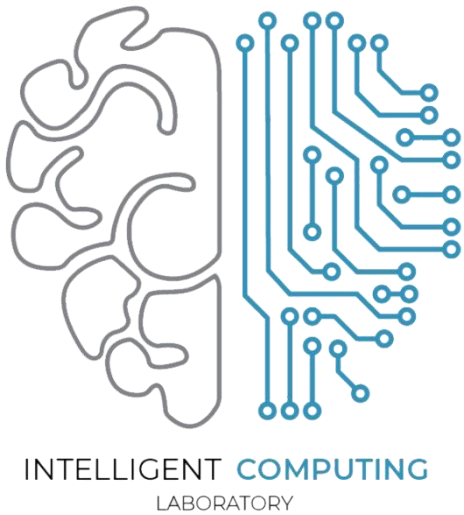
LAPORAN PRAKTIKUM

PENGOLAHAN CITRA DIGITAL



NAMA : Wahid Yaminsyah Putra

NIM : 202231040

KELAS : A

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

NO.PC : 09

ASISTEN : 1. Rafidah Shafa Ariza Ramadhan

2. Althof Zijan Putra Viandhi

3. Raffi Nandyka

INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2023

penjelasan:

[8] :

- Menjalankan fungsi library
- Menjalankan fungsi gray
- Menampilkan fungsi visualisasi data
- Menampilkan hasil langsung di dalam notebook, bukan di jendela terpisah
- Menyediakan algoritma untuk berbagai operasi pengolahan gambar

[9]:

- Membuat variabel image untuk mengimport atau membaca file

[10]:

- Menampilkan Gambar parkir dalam bentuk jendela
- Menahan gambar agar tidak tertutup
- Untuk Menutup tampilan jendela kapanpun

Wahid Yaminsyah Putra 202231040

Deteksi Garis Dan Tepi

[8]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

import skimage
```

[9]:
```
image = cv2.imread('2.jpg')
```

[10]:
```
cv2.imshow("Gambar parkir",image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

[11]:

- Mengkonversi gambar dari BGR menjadi citra gray agar menjadi satu layer warna
- Membuat variabel edges untuk menampung hasil canny(deteksi tepi) dari variabel image, aras keabuan yang di ubah antara 100-150

[12]:

- Menampilkan Gambar parkir dalam bentuk jendela gambar hasil seperti di bawah
- Menahan gambar agar tidak tertutup
- Untuk Menutup tampilan jendela kapanpun

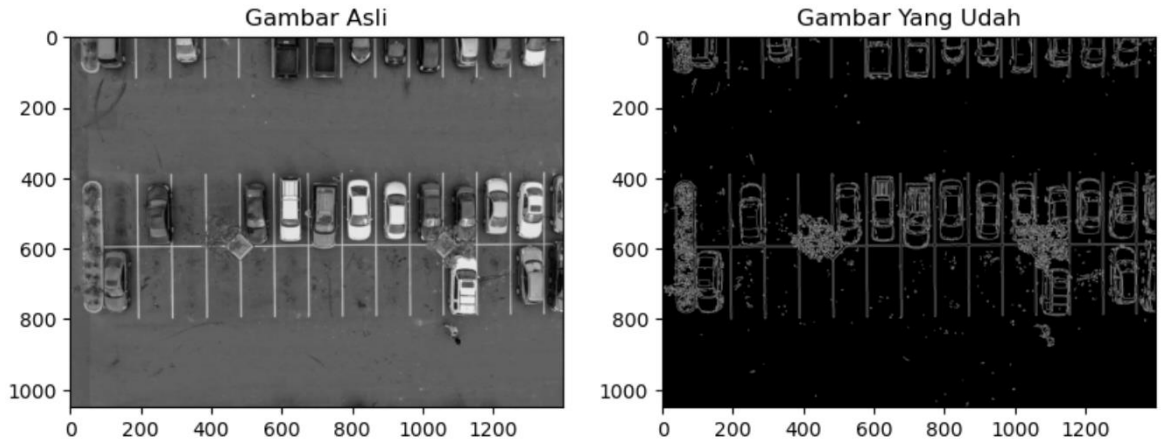
[13]:

- Tampilan menggunakan figure, axis(fungsi matplotlib)

- Sebagai fungsi numpy untuk mengkonversi array menjadi flat array
- Line 4-5 untuk Menampilkan tampilan asli
- Line 7-8 untuk menampilkan edges

Menampilkan Tepi Pada Gambar

```
[11]: gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
      edges = cv2.Canny(image, 100, 150)

[12]: cv2.imshow("Gambar parkir",edges)
      cv2.waitKey(0)
      cv2.destroyAllWindows()

[13]: 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 Yang Udah")

[13]: Text(0.5, 1.0, 'Gambar Yang Udah')
```



[14]:

- mambuat variabel lines untuk menampung hasil houglinesp(deteksi garis)
- membuat variabel image_line untuk membuat salinan variabel image (gambar asli)

[15]:

- prose perulangan antara variabel Line sampai variabel Lines
- line 2-3 membuat tampilan garis pada variabel image_Line, dengan ketentuan (x1,y1) sebagai koordinasi sudut kiri atas, ketentuan (x2,y2) sebagai sudut kanan bawah (0, 0, 225) sebagai kode BGR untuk menghasilkan warna dengan ketentuan ketebalan 1

[16]:

- Tampilan menggunakan figure, axis(fungsi matplotlib)
- Sebagai fungsi numpy untuk mengkonversi array menjadi flat array
- Line 4-5 untuk Menampilkan tampilan asli
- Line 7-8 untuk menampilkan edges

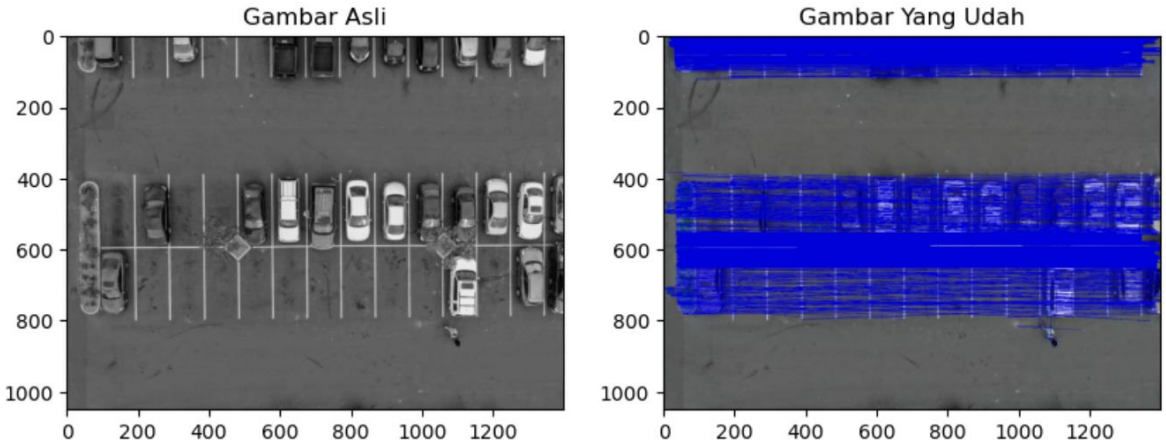
Menampilkan Garis Pada Gambar

[14]:
```
lines =  cv2.HoughLinesP(edges,1,np.pi/180,30,maxLineGap=250)
image_line = image.copy()
```

[15]:
```
for line in lines:
    x1,y1,x2,y2 = line[0]
    cv2.line(image_line,(x1,y1),(x2,y2),(0,0,225),1)
```

[16]:
```
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(image_line, cmap="gray")
ax[1].set_title("Gambar Yang Udah")
```

[16]:
```
Text(0.5, 1.0, 'Gambar Yang Udah')
```



[17]:

- Menjalankan fungsi library
- Menjalankan fungsi gray
- Menampilkan fungsi visualisasi data
- Menampilkan hasil langsung di dalam notebook, bukan di jendela terpisah
- Menyediakan algoritma untuk berbagai operasi pengolahan gambar

[18]:

- Membaca gambar daun dalam mode grayscale dengan nilai 0
- Tinggi dan lebar gambar dari shape gambar

[19]/[*]:

- Menampilkan gambar daun dalam bentuk jendela seperti gambar dibawah
- Menahan gambar agar tidak tertutup
- Untuk Menutup tampilan jendela kapanpun

```
[17]: import cv2
      import numpy as np
      import matplotlib.pyplot as plt
      %matplotlib inline

      import skimage
```

```
[18]: daun = cv2.imread("daun.jpg",0)
      tinggi,lebar = daun.shape
```

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

[20]:

- Menentukan nilai ambang batas untuk thresholding
- Membuat salinan dari gambar daun untuk menyimpan hasil thresholding
- Melakukan iterasi melalui setiap piksel dalam gambar
- Jika nilai piksel kurang dari nilai ambang, maka piksel pada daun_hasil menjadi 0 (hitam)
- Jika nilai piksel lebih besar atau sama dengan nilai ambang, maka piksel pada daun_hasil menjadi 255 (putih)

[21]/[*]:

- Menampilkan gambar daun dalam bentuk jendela
- Menahan gambar agar tidak tertutup
- Untuk Menutup tampilan jendela kapanpun

[22]:

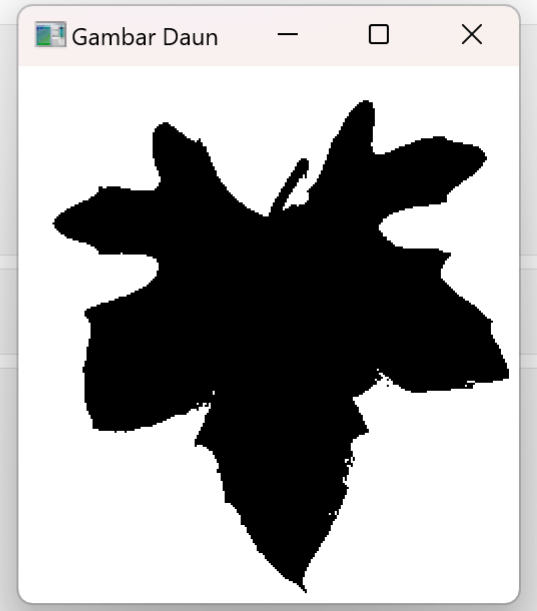
- t0 = 127 untuk Inisialisasi nilai ambang awal
- selanjutnya menggunakan loop untuk menemukan nilai ambang optimal
- line 5-8 untuk Inisialisasi variabel untuk menghitung rata-rata dan jumlah piksel
- line 11-13 Jika nilai piksel kurang dari atau sama dengan 127, maka akan di tambahkan ke kelompok kiri
- line 14-16 Jika nilai piksel lebih dari 127, tambahkan ke kelompok kanan
- 18-19 Menghitung rata-rata intensitas piksel untuk kedua kelompok
- Hitung nilai ambang baru sebagai rata-rata dari kedua rata-rata kelompok
- 21-22 Jika perubahan nilai ambang kurang dari 1, maka keluar dari loop
- Perbarui nilai ambang awal untuk iterasi berikutnya
- Mengembalikan nilai ambang yang dibulatkan

[23]:

- Memanggil fungsi titeratif dengan gambar daun dengan hasil 131

Memperkirakan Nilai Ambang

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

```
[22]: 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((t0-t1)<1 ):
            break
        t0-t1
    return round(t1)
```

```
[23]: titeratif(daun)
```

```
[23]: 131
```

[25]:

- Membaca gambar "arasjamak.jpg" dalam mode grayscale
- Tinggi dan lebar gambar dari shape gambar

[26]:

- Mendefinisikan fungsi untuk melakukan thresholding dengan dua nilai ambang
- Menyimpan referensi ke gambar asli
- Mendapatkan dimensi gambar
- Jika nilai piksel kurang dari atau sama dengan t1 atau lebih besar dari atau sama dengan t2, maka atur piksel menjadi 0 (hitam)
- Jika nilai piksel berada di antara t1 dan t2, maka atur piksel menjadi 255 (putih)
- Mengembalikan gambar hasil thresholding

[27]:

- Memanggil fungsi arasjamak dengan gambar "jamak" dan nilai ambang 185 dan 200

[28]/[*]:

- Menampilkan gambar daun dalam bentuk jendela
- Menahan gambar agar tidak tertutup
- Untuk Menutup tampilan jendela kapanpun

[25]:
```
jamak = cv2.imread("arasjamak.jpg",0)
tinggi,lebar = jamak.shape
```

[26]:
```
def arasjamak(image,t1,t2):
    res=image
    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 :
                res[x,y] = 255
    return res
```

[27]:
```
daunarasjamak = arasjamak(jamak,185,200)
```

[*]:
```
cv2.imshow("gambar daun",daunarasjamak)
cv2.waitKey(0)
cv2.destroyAllWindows()
```