LAPORAN TEORI

PENGOLAHAN CITRA DIGITAL

NAMA : Putri Azzahra

NIM : 202331080

KELAS : B

DOSEN : Ir.Darma Rusjdi, M.Kom

NO.PC : 18

ASISTEN : 1. Davina Najwa Ermawan

2. Fakhrul Fauzi Nugraha Tarigan

3. Viana Salsabila Fairuz Syahla

4. Muhammad Hanief Febriansyah

INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2025

1. Apa yang dimaksud dengan Transformasi Afine dalam transformasi geometrik?
   - Transformasi Afine adalah jenis transformasi geometrik dalam bidang komputer grafis, pengolahan citra, dan geometri yang mempertahankan hubungan garis lurus dan rasio jarak antara titik-titik yang sejajar. Artinya, garis lurus tetap lurus, dan titik-titik yang terletak pada garis lurus tetap berada pada garis lurus setelah transformasi dilakukan.

2. Bandingkan antara operasi deteksi tepi dengan operasi transformasi geometrik (resize dan rotasi). Bandingkan tujuan, cara kerja dan hasilnya!
   - Deteksi Tepi Tujuan : Untuk menemukan batas atau garis tepi dari objek dalam gambar. Ini berguna untuk analisis bentuk, segmentasi, atau pengenalan objek.
     Cara kerja nya : Menggunakan operasi matematika (seperti Sobel, Canny, Prewitt) untuk menghitung perubahan intensitas piksel yang tajam di sekitarnya. Fokusnya pada kontras dan perubahan warna/terang.
     Hasilnya : Gambar hitam putih atau abu-abu, di mana garis tepi objek terlihat jelas, sedangkan bagian dalamnya diabaikan. Gambar jadi seperti sketsa.

   - Transformasi Geometrik (Resize & Rotasi) : Untuk mengubah posisi, ukuran, atau orientasi gambar. Biasa digunakan untuk menyesuaikan tampilan, pemrosesan awal, atau manipulasi citra.
     Cara Kerja nya : Resize mengubah ukuran gambar (memperbesar atau memperkecil) dengan cara interpolasi piksel, seperti nearest neighbor, bilinear, atau bicubic. Sedangkan rotasi memutar seluruh gambar berdasarkan sudut tertentu dengan matriks rotasi. Biasanya pakai transformasi afine.
     Hasilnya : Gambar tetap utuh, hanya saja posisinya berubah sesuai arah putaran (searah jarum jam atau sebaliknya).

3. Apa perbedaan antara deteksi tepi menggunakan Canny dan HoughLinesP? Jelaskan fungsi dan hasilnya!

   - Canny adalah metode deteksi tepi yang digunakan untuk menemukan semua tepi objek dalam gambar, seperti kontur, batas, atau garis lengkung. Prosesnya dimulai dengan menghaluskan gambar (blur), lalu menghitung perubahan intensitas (gradien), dan menyaring hasilnya agar hanya tepi yang jelas yang muncul. Hasil dari Canny berupa gambar hitam-putih yang menampilkan semua tepi objek.

     HoughLinesP adalah metode lanjutan yang digunakan untuk mendeteksi garis lurus dari hasil tepi tersebut (biasanya dari output Canny). Metode ini mencari pola piksel yang membentuk garis lurus dan menggambarkannya sebagai garis nyata di atas gambar. Jadi, HoughLinesP tidak mencari semua tepi, tapi hanya garis-garis lurus saja.

4. Jelaskan apa itu metode interpolasi cubic dan jelaskan perbedaan dengan menggunakan fx dan fy dengan ukuran tinggi dan lebar!
   - Interpolasi cubic (tepatnya bicubic) adalah metode untuk memperhalus gambar saat di-resize dengan cara memperkirakan nilai piksel baru menggunakan 16–24 piksel tetangga di sekitarnya. Metode ini menghasilkan kualitas yang lebih halus dan tajam dibandingkan metode lain seperti nearest atau bilinear, sehingga cocok untuk memperbesar gambar agar tidak pecah.
   - Saat melakukan *resize*, kita bisa memilih cara menentukan ukuran baru gambar, yaitu
     Menggunakan fx dan fy, yaitu skala perbesaran. Misalnya, fx=2 berarti lebar dikali 2, fy=0.5 berarti tinggi dibagi 2.
     Atau menggunakan ukuran tinggi dan lebar langsung (misalnya width=400, height=300) untuk menentukan ukuran akhir gambar secara pasti.
   - Perbedaannya, fx-fy berdasarkan skala perbandingan, sedangkan tinggi-lebar menentukan ukuran absolut dalam piksel. Kita hanya bisa memilih salah satu metode dalam fungsi resize(), tidak bisa digunakan bersamaan.

5. Apa itu rotasi warpAffine dan bagaimana caranya agar gambar yang ingin dirotasi tidak terpotong?

   - Rotasi warpAffine adalah cara memutar gambar di OpenCV menggunakan transformasi afine. Prosesnya dilakukan dengan membuat matriks rotasi pakai cv2.getRotationMatrix2D() dan menerapkannya ke gambar menggunakan cv2.warpAffine(). Namun, kalau langsung diputar begitu saja, gambar bisa terpotong, terutama di bagian sudutnya.
   - Agar gambar tidak terpotong saat rotasi, kita harus menghitung ulang ukuran gambar baru setelah diputar. Caranya dengan menghitung lebar dan tinggi baru berdasarkan sudut rotasi, lalu menggeser pusat rotasi ke tengah frame baru agar gambar tetap seimbang dan utuh. Setelah itu, barulah rotasi dilakukan dengan ukuran kanvas yang lebih besar sehingga seluruh bagian gambar tetap terlihat tanpa terpotong.

LAPORAN PRAKTIKUM

PENGOLAHAN CITRA DIGITAL

NAMA : Putri Azzahra

NIM : 202331080

KELAS : B

DOSEN : Ir.Darma Rusjdi, M.Kom

NO.PC : 18

ASISTEN : 1. Davina Najwa Ermawan

2. Fakhrul Fauzi Nugraha Tarigan

3. Viana Salsabila Fairuz Syahla

4. Muhammad Hanief Febriansyah

INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2025

1. Deteksi tepi

```
[1]: #Putri Azzahra_202331080
     import numpy as np
     import matplotlib.pyplot as plt
     import cv2 as cv
```

Pertama, kita mengimpor library yang diperlukan. Library pertama yang diimpor adalah numpy, yang digunakan untuk memproses data dalam bentuk array atau matriks. Selanjutnya, kita mengimpor matplotlib.pyplot, yang digunakan untuk membuat berbagai visualisasi atau grafik. Terakhir, kita mengimpor cv2 dengan alias cv dari pustaka OpenCV. OpenCV digunakan untuk berbagai tugas, seperti deteksi objek, segmentasi gambar, dan transformasi citra, termasuk membaca gambar dari file, melakukan transformasi geometris, dan menampilkan hasil analisis secara visual.

```
[3]: #Putri Azzahra_202331080
     image = cv.imread("kijang.jpeg")
```

Selanjutnya, kita membuat variabel image yang fungsinya untuk membaca gambar dengan menggunakan fungsi cv.imread(). Fungsi ini akan membuka file gambar yang terletak pada lokasi yang kita tentukan, dalam hal ini adalah file gambar dengan nama "kijang.jpeg". Setelah gambar dibaca, gambar tersebut akan disimpan dalam variabel image

```
[5]: #Putri Azzahra_202331080

     cv.imshow("Gambar kijang", image)
     cv.waitKey(0)
     cv.destroyAllWindows()
```

Selanjutnya, kita menggunakan fungsi cv.imshow() untuk menampilkan gambar yang telah kita baca sebelumnya. Fungsi ini akan menampilkan gambar dalam sebuah jendela baru dengan judul "Gambar kijang". Kemudian, kita menggunakan cv.waitKey(0) yang berfungsi untuk menunggu sampai ada input dari pengguna (biasanya menekan tombol pada keyboard). Setelah itu, fungsi cv.destroyAllWindows() digunakan untuk menutup semua jendela yang terbuka setelah gambar ditampilkan.

```
[7]: #Putri Azzahra_202331080

     gray = cv.cvtColor(image, cv.COLOR_BGR2GRAY)
     edges = cv.Canny(image,75,150)

     cv.imshow("Gambar kijang", image)
     cv.waitKey(0)
     cv.destroyAllWindows()
```

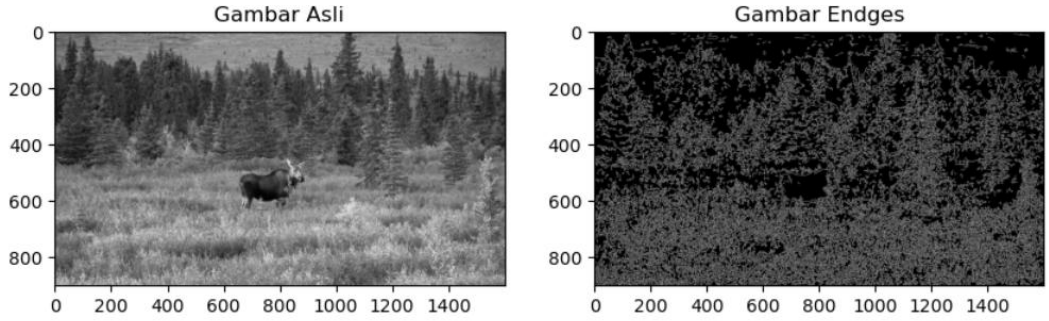
Selanjutnya, kita mengubah gambar yang sudah dibaca menjadi gambar berwarna hitam putih menggunakan fungsi cv.cvtColor(). Fungsi ini mengubah gambar dari format BGR (warna asli) ke format grayscale (abu-abu) dengan kode warna cv.COLOR_BGR2GRAY, dan hasilnya disimpan dalam variabel gray. Setelah itu, kita menggunakan fungsi cv.Canny() untuk mendeteksi tepi gambar. Fungsi ini akan mencari perbedaan kontras yang tajam dalam gambar untuk menemukan tepi-tepi objek, dengan parameter threshold 75 dan 150, dan hasilnya disimpan dalam variabel edges. Terakhir, kita menampilkan gambar asli yang sudah

diproses menggunakan cv.imshow(), dan menunggu hingga tombol ditekan dengan cv.waitKey(0), sebelum menutup jendela gambar menggunakan cv.destroyAllWindows().

```
[13]: #Putri Azzahra_202331080

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

axs[0].imshow(gray, cmap="gray")
axs[0].set_title("Gambar Asli")

axs[1].imshow(edges, cmap="gray")
axs[1].set_title("Gambar Endges")
```

Selanjutnya, kita membuat plot dengan dua gambar berdampingan menggunakan plt.subplots(). Fungsi ini membuat sebuah figure dengan dua kolom, yang masing-masing menampilkan gambar berbeda. Gambar pertama adalah gambar grayscale yang sudah kita buat sebelumnya, yang ditampilkan di axs[0] dengan judul "Gambar Asli". Gambar kedua adalah hasil deteksi tepi (edges) yang ditampilkan di axs[1] dengan judul "Gambar Endges". Kedua gambar tersebut diatur dengan colormap gray agar tampil dalam skala abu-abu.
Outputnya:

```
13]: Text(0.5, 1.0, 'Gambar Endges')
```



Outpunya di sebelah kiri, ada gambar asli yang telah diubah menjadi grayscale, diberi judul "Gambar Asli". Di sebelah kanan, terdapat gambar hasil deteksi tepi (edges) yang menampilkan garis-garis tepi objek dalam gambar, diberi judul "Gambar Endges". Gambar di sebelah kanan menunjukkan detail struktur gambar berdasarkan perbedaan kontras yang tajam.

```
14]: #Putri Azzahra_202331080

lines = cv.HoughLinesP(edges,1,np.pi/180,30,maxLineGap=250)
image_line = image.copy()
```

Selanjutnya, kita menggunakan fungsi cv.HoughLinesP() untuk mendeteksi garis-garis pada gambar berdasarkan hasil deteksi tepi sebelumnya (edges). Hasil deteksi garis disimpan dalam variabel lines. Kemudian, kita membuat salinan gambar asli dengan image.copy() dan menyimpannya di variabel image_line, untuk memastikan gambar asli tetap utuh saat memodifikasi atau menggambar garis pada gambar.

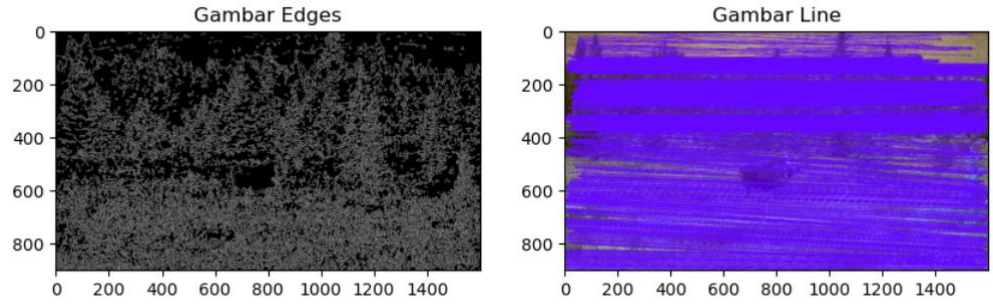
```
[15]: #Putri Azzahra_202331080

for line in lines:
    x1,y1,x2,y2 = line [0]
    cv.line(image_line, (x1,y1),(x2,y2),(100,8,255),1)
```

Selanjutnya, kita menggunakan perulangan for untuk menggambar garis-garis yang terdeteksi pada gambar. Setiap garis yang terdeteksi memiliki koordinat (x1, y1) dan (x2, y2), yang kemudian digunakan untuk menggambar garis pada gambar salinan image_line menggunakan fungsi cv.line(). Garis digambar dengan warna hijau (RGB: 100, 8, 255) dan ketebalan garis 1.

```
[16]: #Putri Azzahra_202331080

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

axs[0].imshow(edges, cmap="gray")
axs[0].set_title("Gambar Edges")

axs[1].imshow(image_line, cmap="gray")
axs[1].set_title("Gambar Line")
```

Selanjutnya, kita membuat plot dengan dua gambar berdampingan. Gambar pertama menampilkan hasil deteksi tepi (edges) dengan judul "Gambar Edges". Gambar kedua menampilkan gambar yang sudah digambar garisnya dengan judul "Gambar Line". Kedua gambar ini ditampilkan dengan colormap gray untuk skala abu-abu.
Outputnya:

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



Di sebelah kiri, merupakan gambar hasil deteksi tepi dengan judul "Gambar Edges" yang menunjukkan garis-garis tajam pada gambar. Di sebelah kanan, ada gambar yang sama dengan garis-garis yang digambar pada gambar menggunakan deteksi Hough, diberi judul "Gambar Line". Garis-garis ini digambarkan dengan warna ungu pada gambar.

2. Geometrik

```
[58]: #Putri Azzahra_202331080

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

Pertama kita impor librabry dulu, library pertama yaitu numpy digunakan untuk operasi numerik pada array atau matriks. Kedua, matplotlib.pyplot digunakan untuk visualisasi data, seperti menampilkan gambar atau grafik. Terakhir, cv2 diimpor dari pustaka OpenCV untuk pengolahan citra dan video, seperti membaca gambar, deteksi tepi, dan transformasi citra.

```
[59]: #Putri Azzahra_202331080

      img_jet = cv2.imread('jalan.jpg')
      rows,cols,_ = img_jet.shape
      print('IMG SHAPE: ', img_jet.shape)

      IMG SHAPE:  (450, 678, 3)
```

Lalu, kita membaca gambar dengan nama "jalan.jpg" menggunakan fungsi cv2.imread() dan menyimpannya dalam variabel img_jet. Selanjutnya, kita mengambil dimensi gambar menggunakan img_jet.shape, yang memberikan jumlah baris, kolom, dan saluran warna gambar. Hasilnya kemudian dicetak dengan print, yang menunjukkan ukuran gambar, yaitu (450, 678, 3), yang berarti gambar memiliki 450 baris, 678 kolom, dan 3 saluran warna (RGB).

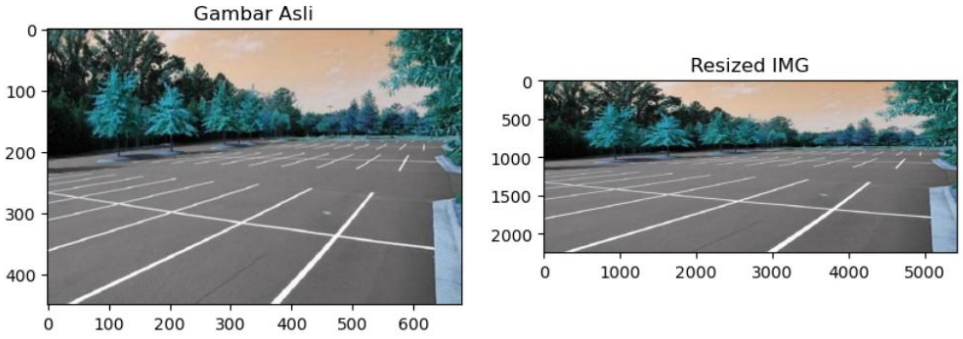
```
[60]: #Putri Azzahra_202331080
      res = cv2.resize(img_jet, None, fx=8, fy=5, interpolation=cv2.INTER_CUBIC)
```

Selanjutnya, kita meresize gambar img_jet menggunakan fungsi cv2.resize(). Gambar diperbesar dengan faktor skala 8 kali untuk lebar (fx=8) dan 5 kali untuk tinggi (fy=5). Proses resizing menggunakan metode interpolasi cv2.INTER_CUBIC untuk menghasilkan gambar yang lebih halus. Hasil gambar yang telah diresize disimpan dalam variabel res.

```
[61]: #Putri Azzahra_202331080

      fig, axs = plt.subplots(1,2,figsize = (10,5))
      axs = axs.ravel()

      axs[0].imshow(img_jet)
      axs[0].set_title("Gambar Asli")

      axs[1].imshow(res)
      axs[1].set_title("Resized IMG")
```

Selanjutnya kita membuat plot dengan dua gambar berdampingan. Gambar pertama adalah gambar asli img_jet yang diberi judul "Gambar Asli". Gambar kedua adalah gambar yang telah diresize res, yang diberi judul "Resized IMG". Kedua gambar ditampilkan dalam satu figure dengan ukuran (10,5).
Outputnya:

```
61]: Text(0.5, 1.0, 'Resized IMG')
```



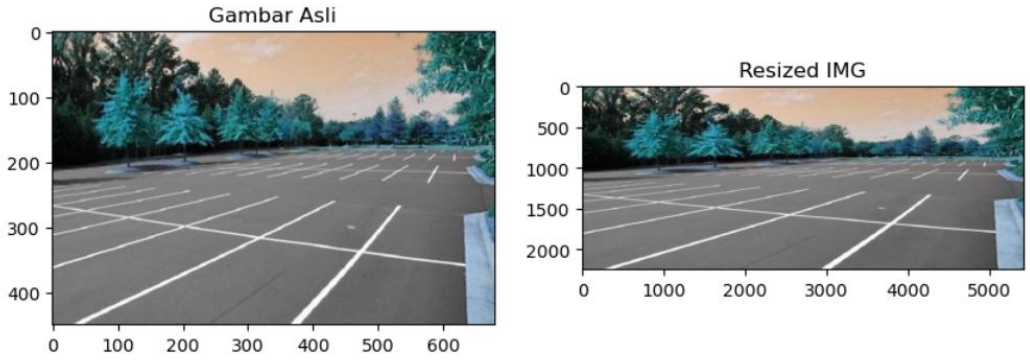
Output ini menampilkan dua gambar berdampingan. Di kiri adalah gambar asli "Gambar Asli", dan di kanan adalah gambar yang telah diresize menjadi lebih besar, dengan judul

"Resized IMG". Gambar yang diresize memiliki dimensi yang lebih besar sesuai dengan faktor skala yang diterapkan.

```
[62]: #Putri Azzahra_202331080

tinggi, lebar = img_jet.shape[:2]
res2 = cv2.resize(img_jet, (8*tinggi, 5*lebar), interpolation=cv2.INTER_CUBIC)

fig, axs = plt.subplots(1,2,figsize = (10,5))
axs = axs.ravel()

axs[0].imshow(img_jet)
axs[0].set_title("Gambar Asli")

axs[1].imshow(res)
axs[1].set_title("Resized IMG")
```

Selanjutnya kita mengambil dimensi gambar img_jet menggunakan img_jet.shape[:2] untuk mendapatkan tinggi dan lebar gambar. Kemudian, gambar diresize dengan faktor skala 8 untuk tinggi dan 5 untuk lebar menggunakan cv2.resize(), hasilnya disimpan di variabel res2. Setelah itu, dua gambar (gambar asli dan gambar yang diresize) ditampilkan berdampingan menggunakan plt.subplots() dengan ukuran (10,5) dan diberikan judul "Gambar Asli" dan "Resized IMG".
Outputnya:

```
[62]: Text(0.5, 1.0, 'Resized IMG')
```



Hasilnya disebelah kiri adalah gambar asli yang diberi judul "Gambar Asli". Di kanan, gambar yang telah diresize dengan faktor skala berbeda ditampilkan dengan judul "Resized IMG". Gambar yang diresize memiliki dimensi yang lebih besar dibandingkan gambar asli.

```
[63]: #Putri Azzahra_202331080

img_jalan = cv2.imread('jalan.jpg', 0)
rows, cols = img_jalan.shape
print("IMG Shape: ", img_jalan.shape)

IMG Shape:  (450, 678)
```

Selanjutnya kita membaca gambar "jalan.jpg" dengan menggunakan cv2.imread() dan disimpan dalam variabel img_jalan. Gambar dibaca dalam format grayscale dengan menambahkan parameter 0. Kemudian, dimensi gambar diambil menggunakan

img_jalan.shape, yang memberikan ukuran gambar (450, 678), artinya gambar memiliki 450 baris dan 678 kolom. Hasilnya kemudian dicetak menggunakan print().

[64]:
```
#Putri Azzahra_202331080

M = cv2.getRotationMatrix2D(((cols - 1) / 2.0, (rows - 1) / 2.0), -250, 1)
img_putar = cv2.warpAffine(img_jalan, M, (cols, rows))

fig, axs = plt.subplots(1,2,figsize = (10,5))
axs = axs.ravel()

axs[0].imshow(img_jalan, cmap='gray')
axs[0].set_title("Gambar Asli")

axs[1].imshow(img_putar, cmap='gray')
axs[1].set_title("Gambar Putar")

for a in axs:
    a.axis('off')

plt.tight_layout()
plt.show()
```

Selanjutnya kita membuat matriks transformasi rotasi menggunakan cv2.getRotationMatrix2D() untuk memutar gambar dengan titik pusat di tengah gambar, dan rotasi sebesar -250 derajat. Kemudian, gambar diputar menggunakan cv2.warpAffine() dan disimpan dalam variabel img_putar. Dua gambar, yaitu gambar asli dan gambar yang telah diputar, ditampilkan berdampingan menggunakan plt.subplots(). Setiap gambar diberi judul "Gambar Asli" dan "Gambar Putar". Terakhir, tampilan plot disesuaikan dengan plt.tight_layout() dan ditampilkan dengan plt.show().
Ouputnya:

Outputnya baagian kiri adalah gambar asli dan gambar kanan adalah hasil rotasi

[9]:
```
#Putri Azzahra_202331080
from skimage import io, transform
img_jalan2 = io.imread('jalan.jpg')

rotated = transform.rotate(img_jalan2, 220, resize=False)
rotated2 = transform.rotate(img_jalan2, 220, resize=True)
```

```
fig, axs = plt.subplots(1, 3, figsize=(15, 5))
axs = axs.ravel()

axs[0].imshow(img_jalan2)
axs[0].set_title("Gambar Asli")

axs[1].imshow(rotated)
axs[1].set_title("Putar 220° (Tanpa Resize)")

axs[2].imshow(rotated2)
axs[2].set_title("Putar 220° (Dengan Resize)")

for a in axs:
    a.axis('off')

plt.tight_layout()
plt.show()
```

Selanjutnya kita membaca gambar "jalan.jpg" menggunakan io.imread() dan menyimpannya dalam variabel img_jalan2. Gambar tersebut kemudian diputar 220 derajat dengan dua cara berbeda. Gambar pertama diputar tanpa resize menggunakan transform.rotate() dengan parameter resize=False, dan hasilnya disimpan di variabel rotated. Gambar kedua diputar dengan resize menggunakan parameter resize=True, disimpan di variabel rotated2. Kedua gambar tersebut, bersama gambar asli, ditampilkan berdampingan menggunakan plt.subplots(). Setiap gambar diberi judul sesuai dengan metode rotasi yang digunakan.
Outputnya:

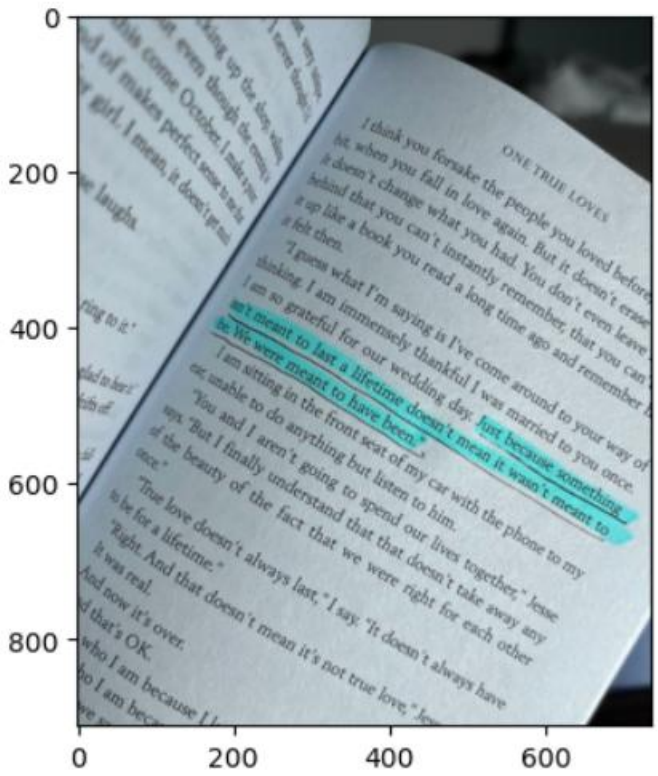
3. Pengaplikasian

```
[74]: #Putri Azzahra_202331080
      t = cv2.imread('buku.jpg')
      plt.imshow(t)
```

Pertama, kita baca gambar dengan nama "buku.jpg" menggunakan cv2.imread() dan disimpan dalam variabel t. Kemudian, gambar tersebut ditampilkan menggunakan plt.imshow() untuk dilihat.

Outputnya:

```
[74]: <matplotlib.image.AxesImage at 0x2313adfdca0>
```



Dan output yang dihasilkan dapat dilihat diatas

```
[75]: #Putri Azzahra_202331080

src = np.array([
    [1,1],
    [731,2],
    [735,903],
    [423,910],
    [1,908]
])

crp = np.array([
    [516,507],
    [728,642],
    [671,710],
    [150,400],
    [202,356]
])

tform = transform.ProjectiveTransform()
tform.estimate(src, crp)

warped = transform.warp(t, tform, output_shape=(50,300))
warped2 = transform.warp (t, tform2, output_shape=(50,300))
```

Selanjutnya, kita mendefinisikan dua set koordinat: src yang mewakili titik-titik pada gambar asli dan crp yang mewakili titik-titik pada gambar tujuan yang diinginkan setelah transformasi. Kemudian, menggunakan pustaka transform.ProjectiveTransform(), kita menghitung matriks transformasi berdasarkan titik kontrol yang telah didefinisikan, dengan memanggil tform.estimate(src, crp). Setelah matriks transformasi dihitung, kita menggunakan fungsi transform.warp() untuk menerapkan transformasi pada gambar dan menghasilkan dua

hasil transformasi yang berbeda, warped dan warped2, dengan ukuran output yang telah ditentukan (50, 300).

```
fig, axs = plt.subplots(1,3,figsize = (10,5))
axs = axs.ravel()

axs[0].imshow(warped)
axs[0].set_title("Warped IMG 1")

axs[1].imshow(t)
axs[1].plot(crp[:,0], crp[:,1], '.r')
axs[1].set_title("Gambar Ori")    #Putri Azzahra_202331080

axs[2].imshow(warped2)
axs[2].set_title("Warped IMG 2")

for a in axs:
    a.axis('off')

plt.tight_layout()
plt.show()
```

Lalu setelah itu, gambar yang telah diubah ini, bersama dengan gambar asli yang ditampilkan dengan titik kontrol, kemudian diposisikan dan diberi judul pada subplot untuk visualisasi yang jelas, menggunakan plt.subplots(). Setiap gambar diberi judul sesuai dengan proses transformasinya, dan sumbu gambar dimatikan dengan a.axis('off') agar tampil lebih rapi. Outputnya: