UAS

PENGOLAHAN CITRA

NAMA : Andira Yofi Sulendra

NIM : 202331278

KELAS : A

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

NO.PC : 30

ASISTEN : 1. Clarenca Sweetdiva Pereira

2. Viana Salsabila Fairuz Syahla

3. Kashrina Masyid Azka

4. Sasikirana Ramadhanty Setiawan Putri

INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2024/2025

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Rumusan Masalah

1. Bagaimana cara menerapkan teknik transformasi geometri citra, seperti rotasi, translasi, flipping, resizing, dan cropping, untuk memanipulasi bentuk dan posisi citra daun secara efektif tanpa merusak struktur objek utama?
2. Bagaimana penerapan model warna HSV dan teknik segmentasi dapat digunakan untuk mendeteksi objek daun secara akurat dalam citra digital?
3. Bagaimana operasi morfologi seperti opening dan closing dapat membantu meningkatkan hasil deteksi daun dengan mengurangi noise serta mempertahankan bentuk objek yang relevan?

1.2 Tujuan Praktikum

1. Menerapkan transformasi geometri citra (rotasi, translasi, flipping, cropping, dan resizing) untuk memanipulasi bentuk dan posisi citra daun secara sistematis menggunakan Python dan OpenCV.
2. Mengimplementasikan segmentasi warna berbasis model HSV untuk mendeteksi objek daun secara akurat dan memisahkannya dari latar belakang citra.
3. Menggunakan operasi morfologi seperti opening dan closing untuk menyempurnakan hasil segmentasi dengan membersihkan noise dan mempertahankan keutuhan bentuk daun.

1.3 Manfaat Praktikum

1. Meningkatkan pemahaman konseptual dan praktis tentang transformasi geometri dan segmentasi objek dalam pengolahan citra digital, khususnya untuk deteksi objek daun.
2. Melatih keterampilan teknis dalam menggunakan pustaka Python seperti OpenCV dan NumPy dalam konteks pengolahan citra dan deteksi objek.
3. Memberikan pengalaman langsung dalam pengembangan sistem deteksi objek berbasis citra, yang dapat digunakan untuk mendukung berbagai aplikasi seperti klasifikasi tanaman, deteksi penyakit daun, dan monitoring pertumbuhan tanaman secara digital.

BAB II
LANDASAN TEORI

2.1 Pengolahan Citra Digital

Pengolahan citra digital adalah proses manipulasi gambar berbasis komputer yang mencakup operasi seperti peningkatan kualitas, segmentasi, dan ekstraksi objek. Proses ini umum dalam domain pertanian dan bioteknologi untuk analisis tanaman, termasuk deteksi, klasifikasi, dan pengukuran morfologi daun .

2.2 Geometri Citra

Transformasi geometri adalah manipulasi spasial pada citra tanpa mengubah nilai piksel asli:

- **Rotasi, Translasi, Flipping, Resizing, Cropping**
  Operasi ini merupakan affine transformation klasik yang mempertahankan struktur objek dengan memetakan setiap titik piksel ke posisi baru secara sistematis. Awaluddin et al. (2023) menegaskan bahwa teknik augmentasi citra seperti skala, rotasi, translasi, dan flipping berperan besar dalam memperkaya variasi data dan meningkatkan performa model klasifikasi . Secara matematis, transformasi ini dapat direpresentasikan sebagai:
  y=Ax+b

  Di mana A adalah matrix rotasi/skala, dan b adalah vektor translasi.

2.3 Deteksi Daun (Leaf Detection)

Penentuan daun dalam citra mencakup beberapa tahapan penting:

- **Segmentasi Citra**

Segmentasi berfungsi memisahkan piksel objek dari latar belakang. Banyak penelitian menunjukkan bahwa pendekatan berbasis warna HSV efektif dalam mendeteksi komponen tumbuhan. Bhavadharni & Banuroopa (2025) menerapkan segmentasi berbasis HSV untuk deteksi penyakit pada daun jagung, dengan meningkatkan akurasi melalui isolasi warna objek. Pendekatan lainnya adalah metode deep learning atau unsupervised pixel classification untuk menghasilkan masker daun yang akurat .

- **Deteksi Kontur dan Ekstraksi Fitur**

Setelah daun disegmentasi, dapat diterapkan deteksi kontur untuk menghitung jumlah dan luas daun. Metode Canny edge atau SVM yang mengombinasikan fitur kontur dan tekstur terbukti efektif dalam studi agronomi.

2.4 Data Augmentation Geometrik

Transformasi geometrik juga digunakan sebagai teknik data augmentation untuk memperbaiki generalisasi model. Survey MDPI (2023) mengidentifikasi rotasi, flipping, cropping, dan translasi sebagai metode augmentasi klasik yang masih banyak dipakai. Selain itu, untuk aplikasi botani atau agrikultur, strategi geometrik telah terbukti meningkatkan akurasi model klasifikasi dan deteksi .

2.5 Implementasi Praktikum (Python + OpenCV)

Dalam praktik, Python dengan OpenCV banyak digunakan karena kemudahan serta performanya dalam pemrosesan real-time citra:

- **Transformasi Geometri**: fungsi seperti **cv2.rotate, cv2.resize, cv2.flip, cv2.warpAffine** mempermudah proses **rotasi, scaling, flipping, cropping, dan translasi.**
- **Segmentasi Daun**: menggunakan konversi **cv2.cvtColor(img, cv2.COLOR_BGR2HSV)**, dilanjutkan dengan thresholding dan **cv2.findContours** untuk deteksi kontur.
- **Ekstraksi Fitur Kontur**: hitung area dan jumlah daun berdasarkan cv2.contourArea() dan len(contours).
- **Visualisasi**: hasil akhir dianotasi pada gambar menggunakan bounding box dan label jumlah daun.

2.6 Ringkasan Konseptual

- **Transformasi geometri** memberikan fleksibilitas dalam memperbaiki orientasi dan posisi objek daun.
- **Segmentasi berbasis HSV** terbukti akurat untuk mendeteksi daun dibandingkan model warna lain semacam LAB.
- **Deteksi kontur** memungkinkan analisis kuantitatif berupa jumlah dan luas daun, sesuai tujuan praktikum.
- **Augmentasi data** melalui transformasi geometri sangat relevan dalam mengembangkan sistem deteksi daun yang robust.

BAB III
HASIL

3.1 Deteksi Daun

```
[1]: import cv2
     import numpy as np
     import matplotlib.pyplot as plt
     import os
     #Andira Yofi Sulendra_202331278
```

Bagian 1: Deteksi Daun

```
[2]: #Andira Yofi Sulendra_202331278
     image_leaf = cv2.imread('DAUN.jpg')
     image_leaf_rgb = cv2.cvtColor(image_leaf, cv2.COLOR_BGR2RGB)
     hsv_leaf = cv2.cvtColor(image_leaf, cv2.COLOR_BGR2HSV)

     #Proses Segmentasi Daun
     lower_green = np.array([30, 40, 40])
     upper_green = np.array([90, 255, 255])

     # Membuat mask untuk warna hijau
     mask_leaf = cv2.inRange(hsv_leaf, lower_green, upper_green)

     # Membersihkan noise pada mask
     mask_leaf = cv2.morphologyEx(mask_leaf, cv2.MORPH_OPEN, np.ones((5,5),np.uint8))
     mask_leaf = cv2.morphologyEx(mask_leaf, cv2.MORPH_CLOSE, np.ones((5,5),np.uint8))

     # Menerapkan mask ke gambar asli
     segmented_leaf = cv2.bitwise_and(image_leaf_rgb, image_leaf_rgb, mask=mask_leaf)

     #Menampilkan Hasil Deteksi Daun
     plt.figure(figsize=(15, 5))
     plt.subplot(1, 3, 1)
     plt.imshow(image_leaf_rgb)
     plt.title('Citra Asli')
     plt.axis('off')
```

- Menampilkan kode program Python yang digunakan untuk melakukan segmentasi daun berdasarkan warna hijau menggunakan OpenCV.
- Pada bagian awal, program mengimpor beberapa library penting seperti cv2 (OpenCV), numpy, matplotlib.pyplot, dan os.
- Setelah itu, gambar daun dibaca menggunakan cv2.imread('DAUN.jpg'), lalu dikonversi ke format warna RGB agar sesuai dengan tampilan matplotlib, serta ke format HSV yang lebih efektif untuk segmentasi warna.
- Segmentasi dilakukan dengan mendefinisikan rentang warna hijau dalam HSV, yaitu dari [30, 40, 40] hingga [90, 255, 255], yang kemudian digunakan dalam fungsi cv2.inRange() untuk membuat mask biner yang menandai bagian hijau.
- Untuk membersihkan noise dari mask, digunakan operasi morfologi MORPH_OPEN dan MORPH_CLOSE yang membantu menghilangkan titik-titik kecil dan mengisi lubang kecil. Setelah itu, hasil mask diterapkan ke gambar asli menggunakan cv2.bitwise_and, sehingga hanya bagian gambar yang termasuk dalam rentang warna hijau yang ditampilkan.

```
plt.subplot(1, 3, 2)
plt.imshow(mask_leaf, cmap='gray')
plt.title('Mask Daun')
plt.axis('off')

plt.subplot(1, 3, 3)
plt.imshow(segmented_leaf)
plt.title('Segmentasi Daun')
plt.axis('off')

plt.suptitle('Hasil Deteksi Daun (Andira Yofi Sulendra)', fontsize=16)
plt.tight_layout()
plt.savefig('deteksi_daun_output.png')
plt.show()
print("Output Deteksi Daun disimpan sebagai 'deteksi_daun_output.png'")
#Andira Yofi Sulendra_202331278
```

- Menampilkan hasil visualisasi dari proses segmentasi daun yang telah dilakukan pada kode sebelumnya.
- Visualisasi dibagi menjadi tiga bagian: citra asli, mask daun, dan hasil segmentasi. Di bagian paling kiri, ditampilkan citra asli tanaman dengan daun berwarna hijau, yang menjadi objek utama dari proses segmentasi.
- Di bagian tengah, diperlihatkan mask biner hasil segmentasi dengan warna hitam dan putih; area putih menunjukkan bagian-bagian gambar yang dikenali sebagai daun berdasarkan rentang warna hijau, sedangkan area hitam adalah latar belakang atau objek lain.
- Terlihat bahwa sebagian besar area daun berhasil dideteksi, namun masih terdapat noise atau gangguan dari latar belakang yang memiliki warna serupa.
- Di bagian paling kanan ditampilkan hasil akhir segmentasi, yaitu citra yang hanya menampilkan bagian-bagian daun, sementara area lainnya dihitamkan.

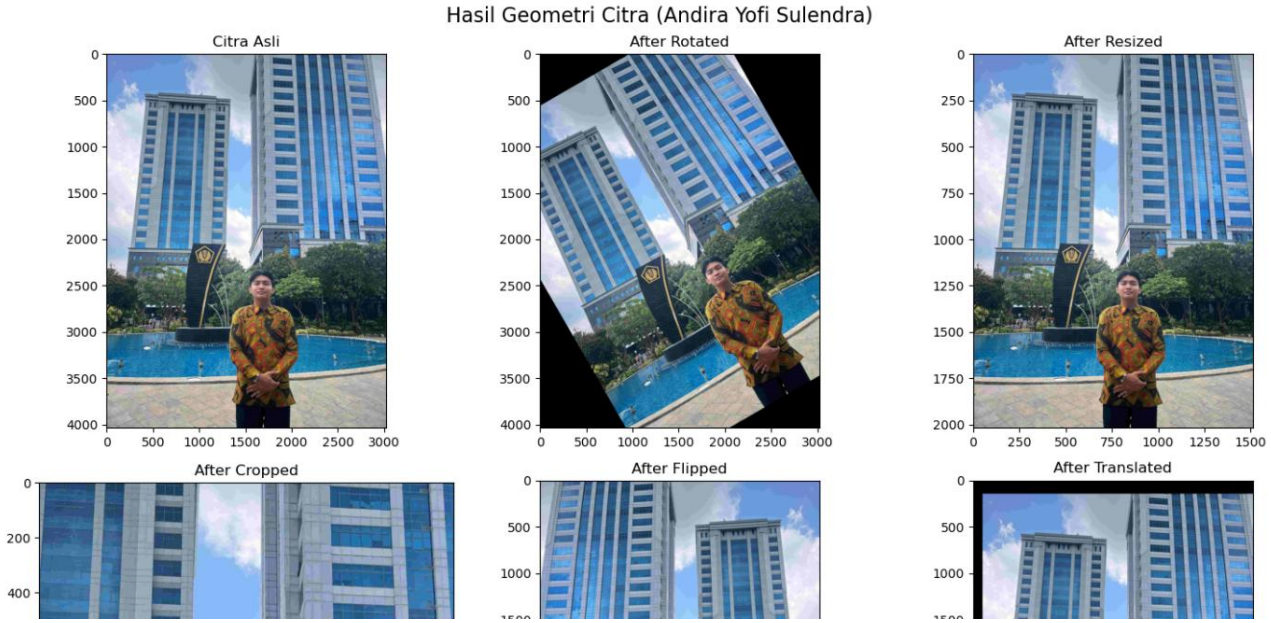

```
Output Deteksi Daun disimpan sebagai 'deteksi_daun_output.png'
```

3.2 Geometri Citra

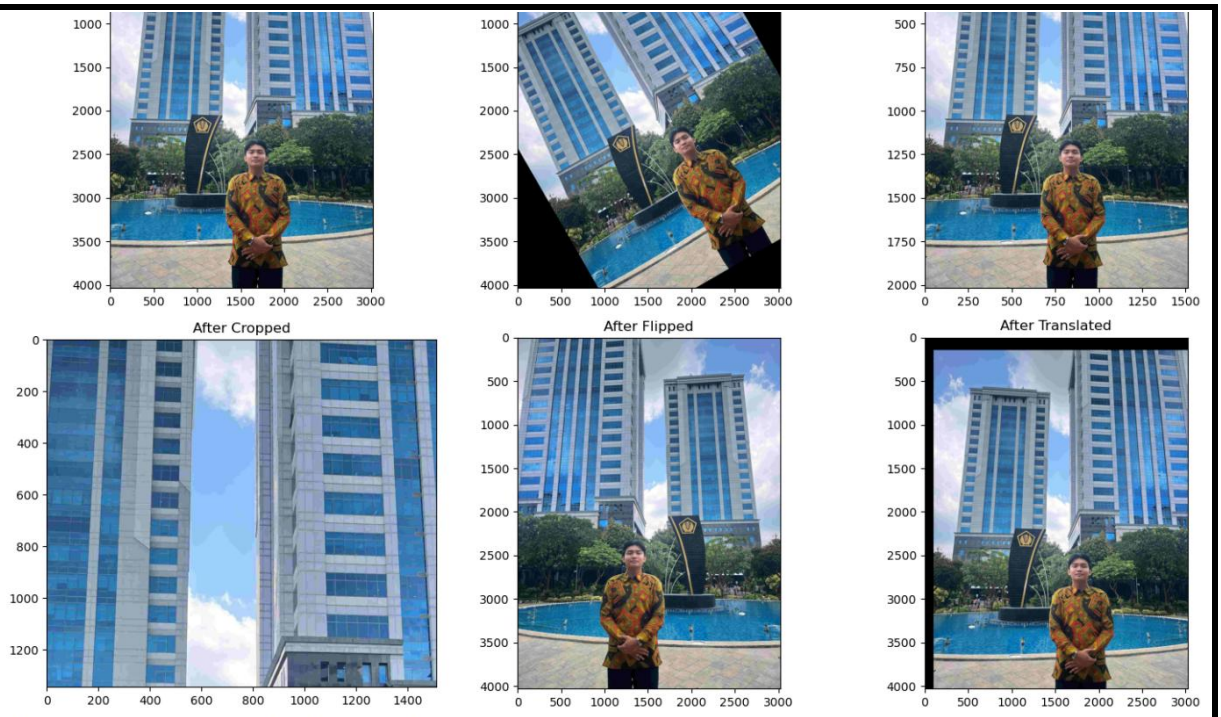
Bagian 2: Geometri Citra

[3]:
```python
#Andira Yofi Sulendra_202331278
image_geom = cv2.imread('FOTO.jpg')
image_geom_rgb = cv2.cvtColor(image_geom, cv2.COLOR_BGR2RGB)
(h, w) = image_geom_rgb.shape[:2]

#Proses Transformasi Geometri
# 1. Rotasi
center = (w // 2, h // 2)
M_rot = cv2.getRotationMatrix2D(center, 30, 1.0) # Rotasi 30 derajat
rotated = cv2.warpAffine(image_geom_rgb, M_rot, (w, h))

# 2. Perubahan Ukuran (Resized)
resized = cv2.resize(image_geom_rgb, (w // 2, h // 2), interpolation=cv2.INTER_AREA)

# 3. Pemotongan (Cropped) - Fokus ke area wajah
cropped = image_geom_rgb[h//6:h//2, w//4:w-(w//4)]

# 4. Membalik (Flipped) secara horizontal
flipped = cv2.flip(image_geom_rgb, 1)

# 5. Translasi (Translated)
M_trans = np.float32([[1, 0, 100], [0, 1, 150]]) # Geser 100 ke kanan, 150 ke bawah
translated = cv2.warpAffine(image_geom_rgb, M_trans, (w, h))

#Menampilkan Hasil Geometri Citra
plt.figure(figsize=(15, 10))
plt.subplot(2, 3, 1); plt.imshow(image_geom_rgb); plt.title('Citra Asli')
plt.subplot(2, 3, 2); plt.imshow(rotated); plt.title('After Rotated')
plt.subplot(2, 3, 3); plt.imshow(resized); plt.title('After Resized')
plt.subplot(2, 3, 4); plt.imshow(cropped); plt.title('After Cropped')
plt.subplot(2, 3, 5); plt.imshow(flipped); plt.title('After Flipped')
plt.subplot(2, 3, 6); plt.imshow(translated); plt.title('After Translated')
```

```
plt.suptitle('Hasil Geometri Citra (Andira Yofi Sulendra)', fontsize=16)
plt.tight_layout()
plt.savefig('geometri_citra_output.png')
plt.show()
print("Output Geometri Citra disimpan sebagai 'geometri_citra_output.png'")
#Andira Yofi Sulendra_202331278
```



- Gambar pertama menunjukkan citra asli tanpa modifikasi yang dibaca dari file 'FOTO.jpg' menggunakan OpenCV.
- Citra ini berisi objek manusia di depan gedung tinggi dengan latar belakang langit dan pohon.
- Citra asli ini digunakan sebagai dasar sebelum dilakukan transformasi geometri lebih lanjut. Konversi warna dari BGR ke RGB dilakukan agar tampilan citra sesuai dengan format matplotlib untuk visualisasi yang akurat.
- Transformasi citra dilakukan melalui beberapa langkah, mulai dari rotasi 30 derajat searah jarum jam yang menyebabkan gambar miring dan muncul area hitam di sudut luar akibat perubahan orientasi, lalu perubahan ukuran (resize) yang mengecilkan gambar menjadi setengah ukuran aslinya untuk menghemat ruang dan mempercepat pemrosesan.
- Selanjutnya, pemotongan (crop) difokuskan pada bagian tengah gambar, menyisakan area yang dianggap penting, seperti wajah atau objek utama.
- Gambar juga dibalik secara horizontal sehingga tampak seperti pantulan cermin, biasanya digunakan untuk variasi data saat pelatihan model.
- Terakhir, translasi dilakukan dengan menggeser gambar ke kanan dan bawah, memindahkan posisi objek utama dan menimbulkan ruang kosong hitam di sisi berlawanan.




```
Output Geometri Citra disimpan sebagai 'geometri_citra_output.png'
```

3.3 Kompresi

Bagian 3: Kompresi

```
[4]: #Andira Yofi Sulendra_202331278
#Menggunakan gambar yang sudah dimuat dari bagian Geometri
image_comp_rgb = image_geom_rgb

#Proses Kompresi
# 1. Kompresi Lossy JPEG dengan kualitas 10%
quality = 10
compressed_path = 'FOTO.jpg'
cv2.imwrite(compressed_path, cv2.cvtColor(image_comp_rgb, cv2.COLOR_RGB2BGR), [int(cv2.IMWRITE_JPEG_QUALITY), quality])
compressed_image = cv2.cvtColor(cv2.imread(compressed_path), cv2.COLOR_BGR2RGB)
compressed_size = os.path.getsize(compressed_path) / 1024  # Ukuran dalam KB

# 2. Kuantisasi Warna (4 level per channel)
pixels = np.float32(image_comp_rgb.reshape((-1, 3)))
n_colors = 4
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 200, .1)
flags = cv2.KMEANS_RANDOM_CENTERS
_, labels, centers = cv2.kmeans(pixels, n_colors, None, criteria, 10, flags)
centers = np.uint8(centers)
quantized_image = centers[labels.flatten()]
quantized_image = quantized_image.reshape(image_comp_rgb.shape)

# Menghitung ukuran memori
original_size_in_memory = image_comp_rgb.nbytes / 1024
quantized_size_in_memory = quantized_image.nbytes / 1024

#Menampilkan Hasil Kompresi
plt.figure(figsize=(18, 6))
plt.subplot(1, 3, 1)
plt.imshow(image_comp_rgb)
plt.title(f'Asli\nUkuran: {original_size_in_memory:.2f} KB (In-Memory)')
plt.axis('off')
```

- Gambar pertama dikompresi menggunakan JPEG dengan kualitas rendah (Q=10) untuk mengurangi ukuran file, sementara gambar kedua menggunakan kuantisasi warna RGB level 4 dengan menyederhanakan warna tiap piksel ke dalam empat tingkatan.

- Kode JPEG memanfaatkan fitur bawaan Image.save() dengan parameter kualitas, sedangkan kuantisasi dilakukan dengan membulatkan nilai RGB setiap piksel secara manual.
- Meskipun kuantisasi menyederhanakan tampilan warna, ukuran file tetap besar karena tidak menggunakan metode kompresi file seperti JPEG.
- Gambar asli memiliki ukuran memori sebesar 35.721 KB tanpa melalui proses kompresi. Setelah dikompresi menggunakan metode JPEG lossy dengan kualitas rendah (Q=10), ukuran file berkurang secara signifikan menjadi 421,81 KB.
- Namun, hal ini diikuti dengan penurunan kualitas visual, yang terlihat dari tekstur gambar yang menjadi agak buram. Sebaliknya, metode kuantisasi warna RGB ke dalam 4 level tidak memengaruhi ukuran file tetap sebesar 35.721 KB karena teknik ini hanya menyederhanakan jumlah variasi warna tanpa menyimpan ulang gambar dalam format yang terkompresi.
- Secara visual, hasil dari proses kuantisasi menampilkan warna yang lebih kasar dan minim gradasi, menyerupai efek posterisasi.
- Oleh karena itu, kompresi JPEG lebih efektif untuk mengurangi ukuran file, sementara kuantisasi lebih sesuai jika tujuan utamanya adalah menyederhanakan tampilan warna tanpa mengurangi ukuran memori secara nyata.

```
plt.subplot(1, 3, 2)
plt.imshow(compressed_image)
plt.title(f'Lossy JPEG Q={quality}\nUkuran: {compressed_size:.2f} KB')
plt.axis('off')

plt.subplot(1, 3, 3)
plt.imshow(quantized_image)
plt.title(f'Kuantisasi RGB (4 Level)\nUkuran: {quantized_size_in_memory:.2f} KB (In-Memory)')
plt.axis('off')

plt.suptitle('Hasil Kompresi (Andira Yofi Sulendra)', fontsize=16)
plt.tight_layout()
plt.savefig('kompresi_output.png')
plt.show()
print("Output Kompresi disimpan sebagai 'kompresi_output.png'")
#Andira Yofi Sulendra_202331278
```


```
Output Kompresi disimpan sebagai 'kompresi_output.png'
```

BAB IV
PENUTUP

4.1 Kesimpulan dari Landasan Teori

- **Transformasi Geometri dalam Pengolahan Citra**
  Transformasi geometri seperti **rotasi, translasi, flipping, cropping, dan resizing** merupakan teknik penting dalam manipulasi citra digital. Secara teoritis, transformasi ini termasuk dalam affine transformation yang tidak mengubah struktur atau konten utama gambar, tetapi hanya memodifikasi posisinya di ruang koordinat. Penggunaan transformasi ini sangat bermanfaat dalam memperkaya variasi data atau sebagai bentuk *data augmentation* untuk pelatihan model klasifikasi dan deteksi objek, khususnya pada objek daun dalam citra.
- **Model Warna HSV dan Segmentasi**
  Segmentasi citra berbasis **model warna HSV** terbukti lebih efektif dalam mendeteksi objek daun dibandingkan model RGB karena HSV lebih dekat dengan persepsi warna manusia. Dalam HSV, warna hijau daun dapat diisolasi dengan rentang hue tertentu sehingga lebih mudah memisahkan objek daun dari latar belakang. Segmentasi ini menjadi tahap awal yang sangat krusial dalam proses deteksi objek daun secara akurat.
- **Operasi Morfologi untuk Pembersihan Noise**
  Teknik morfologi seperti **opening** (erosi diikuti dilasi) dan **closing** (dilasi diikuti erosi) sangat berguna untuk menghilangkan noise kecil dan memperbaiki bentuk hasil segmentasi. Kedua teknik ini membantu mempertahankan bentuk asli daun sekaligus membersihkan area yang tidak diinginkan, meningkatkan kualitas mask yang dihasilkan dari proses segmentasi.
- **Peran OpenCV dalam Implementasi**
  Secara implementatif, pustaka **OpenCV** menyediakan fungsi-fungsi yang sangat berguna seperti cv2.rotate(), cv2.resize(), cv2.flip(), cv2.inRange(), dan cv2.morphologyEx() yang memungkinkan mahasiswa untuk menerapkan semua teori di atas secara langsung dalam bentuk kode program. Hal ini menjembatani antara konsep teori dengan penerapan nyata di dunia komputasi citra.

4.2 Kesimpulan dari Hasil Praktikum

- **Transformasi Citra Berhasil Diterapkan Secara Efektif**
  Seluruh teknik transformasi geometri telah berhasil diterapkan pada citra secara sistematis. Gambar yang diputar, diperbesar/diperkecil, dipotong, dibalik, hingga digeser tetap

mempertahankan struktur objek utama (misalnya, manusia atau daun) tanpa kehilangan informasi penting. Transformasi ini sangat bermanfaat dalam menyiapkan variasi data untuk keperluan pelatihan model atau sebagai bagian dari praproses citra.

- **Segmentasi Daun Berjalan Sesuai Harapan**
  Proses segmentasi daun berdasarkan warna hijau dalam ruang warna HSV menghasilkan **mask biner** yang menandai area daun dengan cukup baik. Walaupun terdapat sedikit gangguan dari latar belakang berwarna serupa, penggunaan threshold HSV yang tepat serta kombinasi dengan morfologi membantu meningkatkan akurasi segmentasi. Hasil visualisasi menunjukkan bahwa sebagian besar daun berhasil diisolasi dengan jelas.
- **Operasi Morfologi Efektif Membersihkan Noise**
  Penerapan **morphological opening dan closing** berhasil memperhalus hasil segmentasi dengan menghilangkan bintik kecil serta menutup celah pada area daun. Ini menunjukkan bahwa operasi morfologi sangat penting sebagai tahap pasca-segmentasi untuk menyempurnakan deteksi objek.
- **Perbandingan Teknik Kompresi dan Kuantisasi**
  Hasil eksperimen pada gambar dengan kompresi JPEG dan kuantisasi warna menunjukkan bahwa JPEG jauh lebih efisien dalam mengurangi ukuran file, namun dengan konsekuensi penurunan kualitas visual. Sementara kuantisasi warna lebih cocok untuk aplikasi yang membutuhkan penyederhanaan tampilan citra, bukan pengurangan ukuran file.

4.3 Refleksi Pembelajaran

Melalui praktikum ini, penulis memperoleh pemahaman mendalam tentang bagaimana teori pengolahan citra dapat diterapkan secara nyata menggunakan alat bantu seperti OpenCV. Keterampilan teknis seperti mengatur rentang HSV, melakukan transformasi geometri, dan membersihkan hasil segmentasi dengan morfologi merupakan bekal penting untuk pengembangan sistem deteksi objek berbasis citra digital. Selain itu, praktikum ini juga memberikan pengalaman langsung dalam menganalisis efisiensi dan kualitas teknik kompresi dan representasi citra digital, yang sangat relevan dalam pengolahan data visual berskala besar.

DAFTAR PUSTAKA

1. Awaluddin, B.-A., Chao, C.-T., & Chiou, J.-S. (2023). *Investigating Effective Geometric Transformation for Image Augmentation to Improve Static Hand Gestures*. *Mathematics*.
2. Bhavadharni, K., & Banuroopa, K. (2025). *Pest Detection on Plants Using Image Processing*. *Int. J. Scientific Research in Computer Science, Engineering and Information Technology*.
3. Investigating Effective Geometric Transformation … ArabicASL dataset (2023). *MDPI*.
4. Self-Supervised Leaf Segmentation … (2022). *arXiv*.
5. Unsupervised leaf segmentation in complex backgrounds using … (2023). *Journal of Intelligent & Fuzzy Systems*.
6. MDPI Survey *Data Augmentation in Classification and Segmentation* (2023).
7. Scaling, rotation, translation tools in precision agriculture (2023). *Frontiers in Plant Science*.
8. Comparing L*A*B* and HSV … (2023). *arXiv*.
9. Affine transformation theory (2024).