UTS

PENGOLAHAN CITRA DIGITAL

NAMA : Ahmad Miftahul Arza Firisky

NIM : 202331067

KELAS : A

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

NO.PC : 12

ASISTEN : 1. Clarenca Sweetdiva Pereira

2. Viana Salsabila Fairuzz 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 mengidentifikasi dan mendeteksi warna-warna spesifik (merah, hijau, biru) dalam sebuah citra digital?
2. Bagaimana pengaruh penggunaan ambang batas (threshold) dengan nilai yang berbeda terhadap hasil segmentasi warna pada citra digital?
3. Bagaimana metode yang tepat untuk memperbaiki kualitas citra dengan kondisi backlight (pencahayaan belakang) yang buruk?
4. Bagaimana pengaruh proses peningkatan kecerahan (brightness) dan kontras (contrast) terhadap citra grayscale dengan kondisi pencahayaan yang kurang optimal?

1.2 Tujuan Masalah

1. Memahami dan mengimplementasikan teknik deteksi warna pada citra digital menggunakan pemisahan channel warna RGB (Red, Green, Blue) dan analisis histogramnya.
2. Menerapkan dan menganalisis pengaruh berbagai nilai ambang batas (threshold) untuk segmentasi warna pada citra, mulai dari ambang batas terkecil hingga terbesar.
3. Mengidentifikasi perbedaan hasil pengenalan citra dengan menggunakan kombinasi warna yang berbeda (None, Blue, Red-Blue, Red-Green-Blue).
4. Mempraktikkan metode perbaikan kualitas citra dengan kondisi backlight melalui konversi ke grayscale dan penerapan teknik peningkatan brightness dan contrast.

1.3 Manfaat Masalah

1. Meningkatkan pemahaman praktis tentang konsep pengolahan citra digital, khususnya dalam deteksi dan segmentasi warna.
2. Memberikan pengetahuan dan keterampilan dalam menganalisis citra digital melalui histogram untuk memahami distribusi intensitas warna.
3. Mengembangkan kemampuan dalam menerapkan berbagai teknik perbaikan kualitas citra untuk mengatasi permasalahan umum seperti pencahayaan backlight yang buruk.
4. Memperoleh pengalaman langsung dalam penggunaan ruang warna HSV untuk segmentasi warna yang lebih efektif dibandingkan dengan ruang warna RGB.

BAB II
LANDASAN TEORI

2.1 Pengolahan Citra Digital

Pengolahan citra digital merupakan disiplin ilmu yang mempelajari teknik-teknik manipulasi citra secara digital. Pengolahan citra digital melibatkan serangkaian operasi matematis yang diterapkan pada citra untuk meningkatkan kualitas atau mengekstrak informasi tertentu. Proses ini menjadi fundamental dalam berbagai aplikasi seperti pengenalan pola, computer vision, dan analisis medis.

Citra digital sendiri merupakan representasi dua dimensi dari citra sebagai sekumpulan nilai digital yang disebut piksel (picture element). Setiap piksel memiliki nilai yang menunjukkan intensitas atau warna pada posisi tertentu dalam citra. Dalam konteks komputasi, citra digital disimpan sebagai matriks dengan dimensi M×N, di mana M merepresentasikan jumlah baris dan N jumlah kolom.

2.2 Model Warna RGB

Model warna RGB (Red, Green, Blue) adalah salah satu model warna yang paling umum digunakan dalam pengolahan citra digital. Model ini didasarkan pada teori bahwa mata manusia sensitif terhadap tiga warna primer: merah, hijau, dan biru. Dalam model warna RGB, setiap piksel digambarkan oleh tiga nilai yang masing-masing merepresentasikan intensitas dari komponen merah, hijau, dan biru.

Dalam representasi digital 8-bit, setiap komponen warna memiliki rentang nilai antara 0 (tidak ada intensitas) hingga 255 (intensitas penuh). Dengan demikian, model warna RGB dapat merepresentasikan sekitar 16,7 juta warna yang berbeda (2^24). Kombinasi dari tiga komponen warna dengan intensitas berbeda akan menghasilkan berbagai macam warna.

2.3 Model Warna HSV

HSV (Hue, Saturation, Value) merupakan model warna alternatif yang sering digunakan dalam pengolahan citra, terutama untuk segmentasi warna. Berbeda dengan RGB yang merupakan model warna aditif, HSV lebih sesuai dengan bagaimana manusia mempersepsikan warna.

Komponen Hue merepresentasikan jenis warna (seperti merah, hijau, atau biru) dan dinyatakan dalam derajat dari 0° hingga 360°. Saturation menunjukkan kemurnian atau intensitas dari warna (0-100%), sementara Value mengindikasikan kecerahan warna (0-100%). Model warna HSV lebih efektif untuk segmentasi warna karena memisahkan informasi intensitas (Value) dari informasi warna (Hue), sehingga lebih tahan terhadap variasi

2.4 Histogram

Histogram citra adalah representasi grafis yang menunjukkan distribusi frekuensi dari intensitas piksel dalam sebuah citra. Dalam citra grayscale, histogram menampilkan jumlah piksel untuk setiap nilai intensitas dari 0 (hitam) hingga 255 (putih). Untuk citra berwarna, histogram dapat dibuat untuk setiap channel warna (R, G, B) secara terpisah.

Analisis histogram memberikan informasi penting tentang kontras, kecerahan, dan distribusi intensitas citra. Histogram yang memiliki distribusi merata di seluruh rentang intensitas umumnya menghasilkan citra dengan kontras yang baik, sementara histogram yang terkonsentrasi pada rentang sempit menandakan citra dengan kontras rendah.

2.5 Segmentasi Citra Berbasis Warna

Segmentasi citra berbasis warna merupakan teknik untuk memisahkan objek dalam citra berdasarkan warnanya. Teknik ini sangat penting dalam berbagai aplikasi computer vision seperti pengenalan objek dan pelacakan gerakan. Proses segmentasi berbasis warna umumnya melibatkan penentuan rentang nilai untuk masing-masing komponen warna yang akan dideteksi.

Thresholding adalah salah satu metode segmentasi paling sederhana yang bekerja dengan cara memisahkan piksel berdasarkan nilai ambang batas tertentu. Dalam segmentasi warna, thresholding diterapkan pada setiap channel warna atau pada komponen dalam model warna tertentu seperti HSV.

2.6 Perbaikan Kualitas Citra

Perbaikan kualitas citra (image enhancement) bertujuan untuk meningkatkan kualitas visual citra agar lebih mudah diinterpretasi oleh manusia atau diproses lebih lanjut oleh komputer. Teknik perbaikan kualitas citra dapat dibagi menjadi dua kategori utama: domain spasial (yang bekerja langsung pada piksel) dan domain frekuensi (yang bekerja pada transformasi citra).

2.7 Masalah Backlight dalam Citra Digital

Backlight adalah kondisi ketika sumber cahaya utama berada di belakang objek yang difoto, sehingga objek tampak gelap atau siluet. Citra dengan kondisi backlight seringkali memiliki kontras yang tinggi antara area terang dan gelap, namun detail pada objek utama cenderung hilang karena pencahayaan yang tidak memadai.

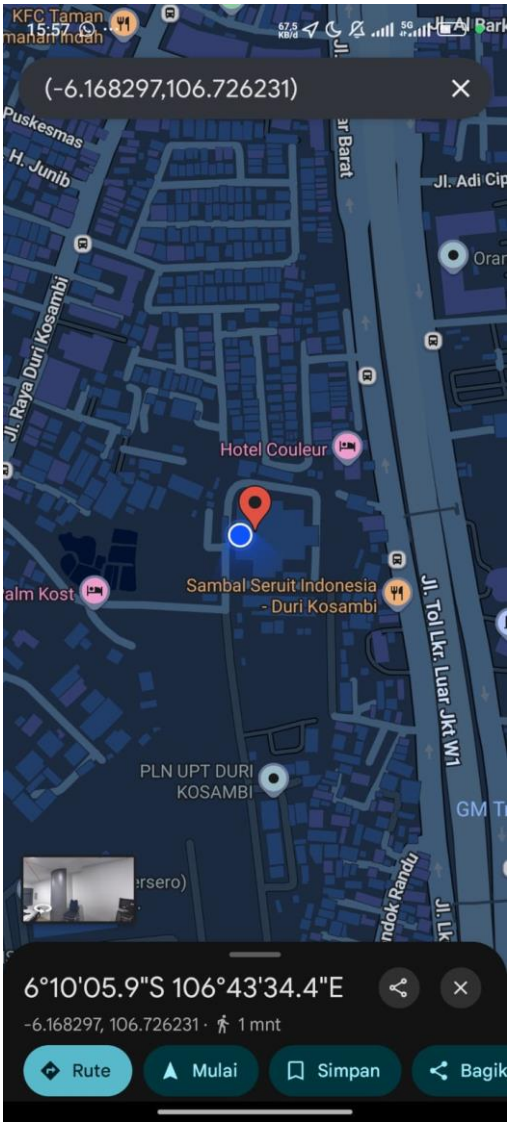
Untuk mengatasi masalah backlight, berbagai teknik perbaikan citra dapat diterapkan, seperti penyesuaian gamma, ekualisasi histogram adaptif, dan filter lokal. Kombinasi antara peningkatan kecerahan dan kontras seringkali efektif untuk memperbaiki citra dengan kondisi backlight.

BAB III
HASIL

- Bukti Gambar

Lokasi :


15:58
Rincian
Waktu ambil: 6 Mei 2025
Selasa 15:56
Info berkas: IMG_20250506_155654.jpg
5,07MB 4624x3472px
Data EXIF: POCO F5, Xiaomi
f/1.8 1/4846s ISO50
4.68mm(Panjang fokus ekuivalen 24 mm)
Tanpa flash
Jalur lokal: Penyimpanan internal/DCIM/Camera/
IMG_20250506_155654.jpg
Lokasi: Jl. Duri Cengkareng Raya No.RT.10, RT.1/
RW.7, Duri Kosambi, Kecamatan Cengkareng,
Kota Jakarta Barat, Daerah Khusus Ibukota
Jakarta 11750, Indonesia


(-6.168297,106.726231)
Hotel Couleur
Sambal Seruit Indonesia
- Duri Kosambi
PLN UPT DURI
KOSAMBI
Jl. Raya Duri Kosambi
Jl. Tol Lkr. Luar Jkt W1
6°10'05.9"S 106°43'34.4"E
-6.168297, 106.726231 · 1 mnt
Rute
Mulai
Simpan

3.1 Deteksi Warna Pada Citra

- Potret Citra yang bertuliskan nama lengkap masing-masing.
  Gambar Citra :

- Program

Import Library

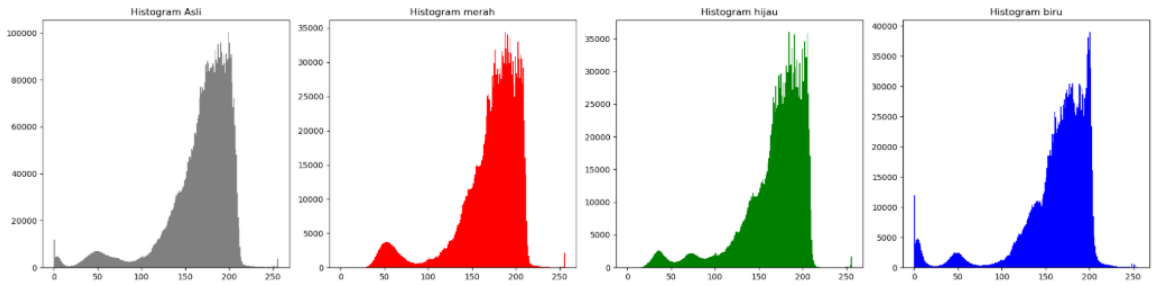
[3]:
```
import cv2
import matplotlib.pyplot as plt
import numpy as np
## 202331067_Ahmad Miftahul Arza Firisky
```

[4]:
```
image = cv2.imread("foto_nama1.jpg")
```

[5]:
```
# Memisahkan channel BGR
B, G, R = cv2.split(image)

# Konversi BGR ke RGB
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Membuat subplot untuk menampilkan gambar
fig, axs = plt.subplots(2, 2, figsize=(15, 8))

# Citra RGB
axs[0, 0].imshow(image_rgb)
axs[0, 0].set_title("CITRA KONTRAS")
axs[0, 0].axis('on')
axs[0, 0].grid(False)

# Biru
axs[0, 1].imshow(B, cmap='gray')
axs[0, 1].set_title("BIRU")
axs[0, 1].axis('on')
axs[0, 1].grid(False)

# Merah
axs[1, 0].imshow(R, cmap='gray')
axs[1, 0].set_title("MERAH")
axs[1, 0].axis('on')
axs[1, 0].grid(False)

# CHijau
axs[1, 1].imshow(G, cmap='gray')
axs[1, 1].set_title("HIJAU")
axs[1, 1].axis('on')
axs[1, 1].grid(False)

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

```
[6]: # Menampilkan histogram dari RGB dan masing-masing channel
plt.figure(figsize=(20, 5))

plt.subplot(141)
plt.title('Histogram Asli')
plt.hist(image_rgb.ravel(), bins=256, range=[0, 256], color='gray')

plt.subplot(142)
plt.title('Histogram merah')
plt.hist(R.ravel(), bins=256, range=[0, 256], color='r')

plt.subplot(143)
plt.title('Histogram hijau')
plt.hist(G.ravel(), bins=256, range=[0, 256], color='g')

plt.subplot(144)
plt.title('Histogram biru')
plt.hist(B.ravel(), bins=256, range=[0, 256], color='b')

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

- Hasil Run



- Ekstraksi Warna
  Pada gambar pertama, terlihat bahwa telah memisahkan citra asli (kontras) menjadi tiga channel warna yaitu : Merah, Hijau, Biru.
  - Histogram Asli :
    - Tampak dua puncak utama — satu di intensitas rendah (sekitar 30–60) dan satu lagi di intensitas tinggi (sekitar 200–255). Puncak tinggi di kanan menunjukkan area latar belakang gambar (kertas putih), sementara puncak di kiri mewakili teks berwarna yang lebih gelap.
  - Channel (Biru) :
    - Tulisan berwarna biru ("HUL ARZA") justru tidak terlihat jelas di channel ini, ini karena objek berwarna biru memiliki nilai tinggi di channel biru, sehingga tampak terang (nyaris putih) saat ditampilkan dalam skala abu-abu.
    - Warna biru "hilang" atau "kabur" karena kontras rendah terhadap latar belakang putih.

  - Channel Hijau (G) :
    - Tulisan hijau ("AHMAD MIFTA") juga tampak kurang jelas, hampir menyatu dengan latar belakang.
    - Hal ini lagi-lagi karena warna hijau punya intensitas tinggi di channel G, sehingga tampak cerah (abu-abu muda sampai putih), tidak kontras.

  - Channel Merah (R) :
    - Tulisan merah ("FIRISKY") terlihat lebih gelap dan jelas, artinya channel ini mendeteksi merah dengan baik — karena dalam channel R, warna merah tampil dengan nilai rendah ke sedang (lebih gelap) pada bagian teks dan latar yang terang.

- Analisis Histogram
  - Histogram Channel Merah (Red) :
    - Distribusi mulai naik pada rentang 100 hingga 250, dengan lonjakan signifikan di atas 200. Ini menunjukkan dominasi warna terang (latar belakang). Namun, terdapat lekukan kecil di rentang intensitas menengah (sekitar 150–180), yang mewakili teks "FIRISKY" berwarna merah. Teks ini tampak lebih jelas di channel merah karena memiliki nilai intensitas yang lebih gelap dibanding latar.

  - Histogram Channel Hijau (Green)
    - Distribusi mirip dengan channel merah, tetapi lebih banyak puncak di kisaran menengah–tinggi (sekitar 170–230). Ini menunjukkan bahwa teks hijau "AHMAD MIFTA" memiliki nilai hijau tinggi dan menyatu dengan latar belakang, sehingga tampak kabur pada channel hijau.

- Histogram Channel Biru (Blue)
    - Terdapat dua puncak besar di rentang tinggi (sekitar 180–255), mirip dengan channel hijau. Ini mengindikasikan area teks biru “HUL ARZA” yang tidak terlihat jelas di channel biru karena memiliki nilai intensitas tinggi (cerah), sehingga tampak terang dan kurang kontras.

Histogram membantu mengidentifikasi bagian mana dari gambar yang memiliki nilai intensitas rendah (gelap/teks) dan tinggi (terang/latar). Channel merah lebih berhasil menampilkan teks berwarna merah karena memberikan kontras tinggi terhadap latar. Channel hijau dan biru tidak menampilkan teks hijau/biru dengan baik karena nilai intensitas warnanya terlalu tinggi, sehingga tampak kurang kontras dan menyatu dengan latar.

3.2 Ambang Batas Terkecil Sampai Dengan Terbesar

- Program

[7]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Load an image first
color_image = cv2.imread('foto_nama1.jpg')  # Replace with your image path

# Now proceed with your color processing
hsv = cv2.cvtColor(color_image, cv2.COLOR_BGR2HSV)

lower_blue = np.array([100, 50, 50])
upper_blue = np.array([130, 255, 255])

mask = cv2.inRange(hsv, lower_blue, upper_blue)

hasil = cv2.bitwise_and(color_image, color_image, mask=mask)

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

axs[0].imshow(cv2.cvtColor(color_image, cv2.COLOR_BGR2RGB))
axs[0].set_title('Citra Asli')

axs[1].imshow(mask, cmap='gray')
axs[1].set_title('Blue')

plt.show()
```

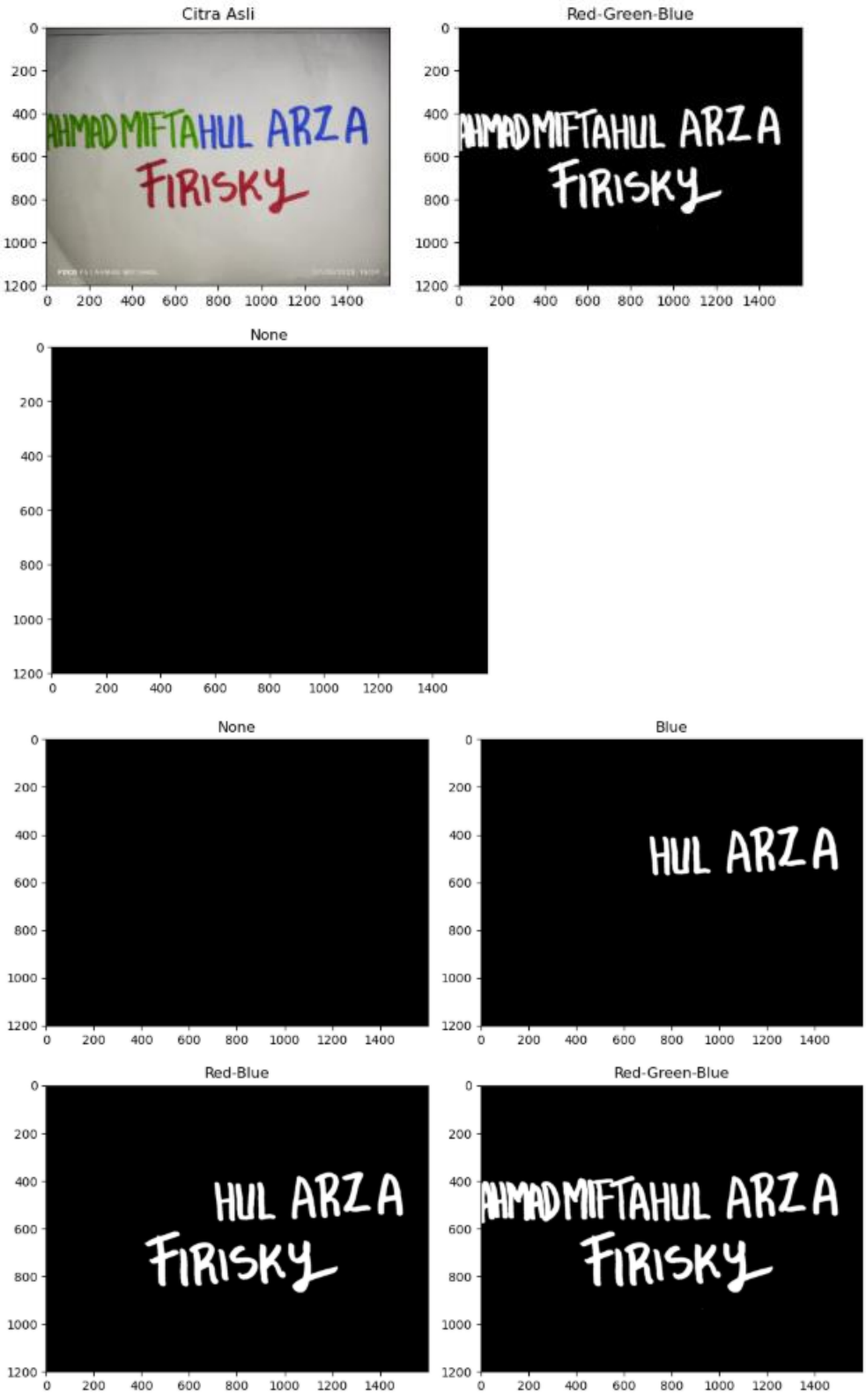
[8]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Load the image
img = cv2.imread('foto_nama1.jpg')
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# Convert to HSV for color detection
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

# Define blue and red ranges in HSV
blue_mask = cv2.inRange(hsv, np.array([100, 50, 50]), np.array([130, 255, 255]))

# Red needs two ranges (wraps around in HSV)
red_mask1 = cv2.inRange(hsv, np.array([0, 50, 50]), np.array([10, 255, 255]))
red_mask2 = cv2.inRange(hsv, np.array([170, 50, 50]), np.array([180, 255, 255]))
red_mask = cv2.bitwise_or(red_mask1, red_mask2)

# Combine red and blue masks
red_blue_mask = cv2.bitwise_or(red_mask, blue_mask)

# Display original and mask
fig, ax = plt.subplots(1, 2, figsize=(10, 4))
ax[0].imshow(img_rgb)
ax[0].set_title('Citra Asli')
ax[0].axis('off')

ax[1].imshow(red_blue_mask, cmap='gray')
ax[1].set_title('Red-Blue')
ax[1].axis('off')

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

[9]:
```
hsv = cv2.cvtColor(color_image, cv2.COLOR_BGR2HSV)

lower_blue = np.array([100, 50, 50])
upper_blue = np.array([130, 255, 255])

lower_red1 = np.array([0, 50, 50])
upper_red1 = np.array([10, 255, 255])
lower_red2 = np.array([170, 50, 50])
upper_red2 = np.array([180, 255, 255])

lower_green = np.array([35, 50, 50])
upper_green = np.array([85, 255, 255])

mask_blue = cv2.inRange(hsv, lower_blue, upper_blue)
mask_red1 = cv2.inRange(hsv, lower_red1, upper_red1)
mask_red2 = cv2.inRange(hsv, lower_red2, upper_red2)
mask_green = cv2.inRange(hsv, lower_green, upper_green)

mask_combined = cv2.bitwise_or(mask_blue, mask_red1)
mask_combined = cv2.bitwise_or(mask_combined, mask_red2)
mask_combined = cv2.bitwise_or(mask_combined, mask_green)

hasil = cv2.bitwise_and(color_image, color_image, mask=mask_combined)

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

axs[0].imshow(cv2.cvtColor(color_image, cv2.COLOR_BGR2RGB))
axs[0].set_title('Citra Asli')

axs[1].imshow(mask_combined, cmap='gray')
axs[1].set_title('Red-Green-Blue')

plt.show()
```

[10]:
```
def display_image(image, title="Image"):
    plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
    plt.title(title)
    plt.axis('on')
    plt.show()

hsv_image = cv2.imread('foto_nama1.jpg', cv2.IMREAD_COLOR)
hsv_image = cv2.cvtColor(hsv_image, cv2.COLOR_BGR2HSV)

lower_blue = np.array([100, 50, 50])
upper_blue = np.array([130, 255, 255])

lower_red1 = np.array([0, 50, 50])
upper_red1 = np.array([10, 255, 255])

lower_red2 = np.array([170, 50, 50])
upper_red2 = np.array([180, 255, 255])

lower_green = np.array([50, 50, 50])
upper_green = np.array([80, 255, 255])

blue_mask = cv2.inRange(hsv_image, lower_blue, upper_blue)
red_mask1 = cv2.inRange(hsv_image, lower_red1, upper_red1)
red_mask2 = cv2.inRange(hsv_image, lower_red2, upper_red2)
green_mask = cv2.inRange(hsv_image, lower_green, upper_green)

red_blue_mask = cv2.bitwise_or(red_mask1, red_mask2)
combined_mask = cv2.bitwise_or(blue_mask, cv2.bitwise_or(red_blue_mask, green_mask))

black_image = np.zeros_like(hsv_image, dtype=np.uint8)

black_detected_image = cv2.bitwise_and(black_image, black_image, mask=combined_mask)

display_image(black_detected_image, "None")
```

[11]:
```
# Convert to HSV for color detection
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

# Define color ranges in HSV
blue_mask = cv2.inRange(hsv, np.array([100, 50, 50]), np.array([130, 255, 255]))

# Red needs two ranges (wraps around in HSV)
red_mask1 = cv2.inRange(hsv, np.array([0, 50, 50]), np.array([10, 255, 255]))
red_mask2 = cv2.inRange(hsv, np.array([170, 50, 50]), np.array([180, 255, 255]))
red_mask = cv2.bitwise_or(red_mask1, red_mask2)

# Green range
green_mask = cv2.inRange(hsv, np.array([35, 50, 50]), np.array([85, 255, 255]))

# Combine masks
red_blue_mask = cv2.bitwise_or(red_mask, blue_mask)
red_blue_green_mask = cv2.bitwise_or(red_blue_mask, green_mask)

# Create black image for "None" panel
black_image = np.zeros_like(img_rgb)

# Display 4 panels (2x2)
fig, axs = plt.subplots(2, 2, figsize=(10, 8))

# None panel (black image)
axs[0, 0].imshow(black_image)
axs[0, 0].set_title('None')

# Blue mask
axs[0, 1].imshow(blue_mask, cmap='gray')
axs[0, 1].set_title('Blue')

# Red-Blue mask
axs[1, 0].imshow(red_blue_mask, cmap='gray')
axs[1, 0].set_title('Red-Blue')

# Red-Green-Blue mask
axs[1, 1].imshow(red_blue_green_mask, cmap='gray')
axs[1, 1].set_title('Red-Green-Blue')

# Turn on axes for all subplots
for ax in axs.flat:
    ax.axis('on')

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

- Hasil Run


Citra Asli
Red-Green-Blue
None
None
Blue
Red-Blue
Red-Green-Blue

- Urutan Ambang Batas dari yang Terkecil ke Terbesar
  1. None
     Tidak ada warna yang ditampilkan karena semua threshold tidak digunakan (mask = 0). Tidak ada threshold yang digunakan, semua mask = 0. Menampilkan citra hitam polos. Ambang batas efektif: 0 warna dikenali terendah.
  2. Blue
     Hanya bagian teks berwarna biru ("HUL ARZA") yang muncul. Threshold HSV untuk warna biru diterapkan. Hanya teks "HUL ARZA" yang muncul (teks berwarna biru). Ambang batas: hanya mencakup satu jenis warna masih rendah.
  3. Red-Blue
     Menampilkan kombinasi teks merah dan biru ("HUL ARZA" + "FIRISKY"). Kombinasi ambang batas merah dan biru. Teks "HUL ARZA" dan "FIRISKY" berhasil ditampilkan. Ambang batas mencakup dua spektrum warna sedang.
  4. Red-Green-Blue
     Menampilkan keseluruhan teks ("AHMAD MIFTAHUL ARZA FIRISKY") karena semua warna (merah, hijau, biru) sudah dicakup. Menggunakan semua ambang batas warna utama (merah, hijau, biru). Semua teks "AHMAD MIFTAHUL ARZA FIRISKY" berhasil dikenali. Ambang batas mencakup seluruh warna teks terbesar.

- Nilai Ambang Batas Warna dan Penjelasannya
  1. Biru (Blue)
     - Hue : 100 – 130
     - Saturation : 50 – 255
     - Value : 50 – 255
     - Alasan: Nilai ini mencakup spektrum warna biru dalam ruang HSV. Diterapkan untuk menampilkan bagian teks yang ditulis dengan warna biru, seperti "HUL ARZA". Rentang Saturation dan Value diatur tidak terlalu rendah agar menghindari deteksi noise dari area gelap.
  2. Merah (Red)
     - Hue : 0 – 10 dan 170 – 180
     - Saturation : 50 – 255
     - Value : 50 – 255
     - Alasan : Warna merah di HSV memerlukan dua rentang karena hue untuk merah berada di dua sisi spektrum (wrap-around). Nilai ini digunakan untuk menangkap teks berwarna merah seperti "FIRISKY". Nilai Saturation dan Value yang cukup tinggi menjaga agar hanya warna merah pekat yang terdeteksi.
  3. Hijau (Green)
     - Hue : 35 – 85
     - Saturation : 50 – 255
     - Value : 50 – 255
     - Alasan : Rentang ini mencakup hijau dari muda hingga tua. Diperlukan untuk mendeteksi huruf berwarna hijau seperti pada bagian "AHMAD

MIFTAHUL". Penggunaan nilai HSV ini cukup umum dalam segmentasi warna hijau karena mencakup sebagian besar gradasi hijau alami.

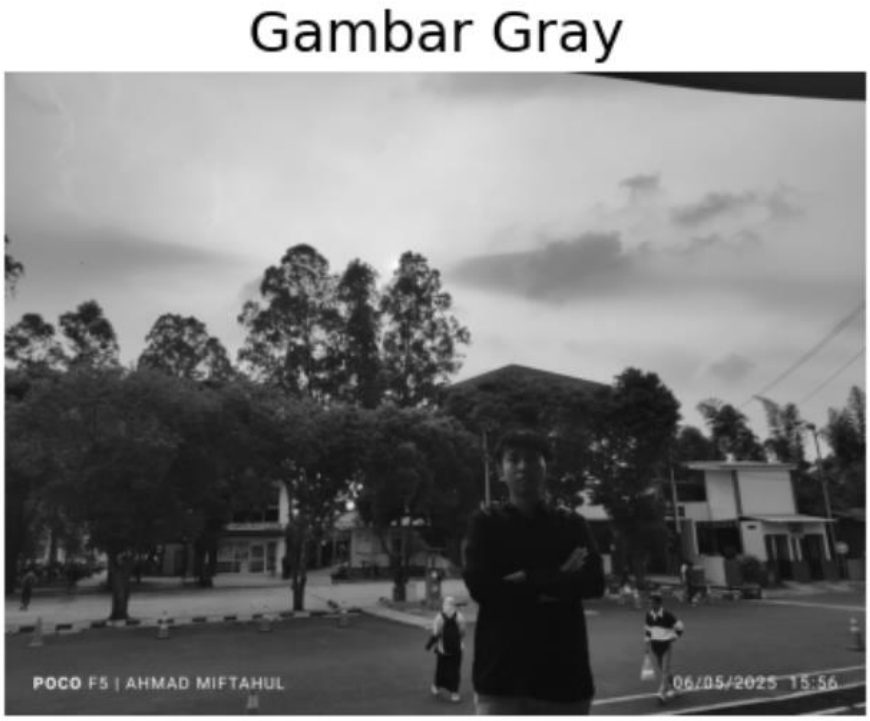
3.3 Memperbaiki Gambar Backlight

- Program

[12]:
```python
# Baca gambar asli dari file
path = "foto.jpg"
img = cv2.imread(path)

# Konversi BGR ke RGB untuk ditampilkan dengan matplotlib
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# Konversi ke grayscale
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Pencerahan gambar gray (ditambah nilai brightness)
bright_gray = cv2.convertScaleAbs(img_gray, alpha=1, beta=50)  # beta menambah kecerahan

# Penambahan kontras pada gambar gray
contrast_gray = cv2.convertScaleAbs(img_gray, alpha=1.5, beta=0)  # alpha meningkatkan kontras

# Kombinasi: Dipercerah + Diperkontras
bright_contrast_gray = cv2.convertScaleAbs(img_gray, alpha=1.5, beta=50)

plt.figure(figsize=(20, 30))  # Lebar diperbesar

plt.subplot(6, 1, 1)
plt.imshow(img_rgb)
plt.title("Gambar Asli", fontsize=18)
plt.axis("off")

plt.subplot(6, 1, 2)
plt.imshow(img_gray, cmap='gray')
plt.title("Gambar Gray", fontsize=18)
plt.axis("off")

plt.subplot(6, 1, 3)
plt.imshow(bright_gray, cmap='gray')
plt.title("Gambar Gray yang Dipercerah", fontsize=18)
plt.axis("off")

plt.subplot(6, 1, 4)
plt.imshow(contrast_gray, cmap='gray')
plt.title("Gambar Gray yang Diperkontras", fontsize=18)
plt.axis("off")

plt.subplot(6, 1, 5)
plt.imshow(bright_contrast_gray, cmap='gray')
plt.title("Gambar Gray Dipercerah & Diperkontras", fontsize=18)
plt.axis("off")

plt.subplots_adjust(hspace=0.4)  # Atur jarak antar gambar
plt.show()
```

- Gambar Asli

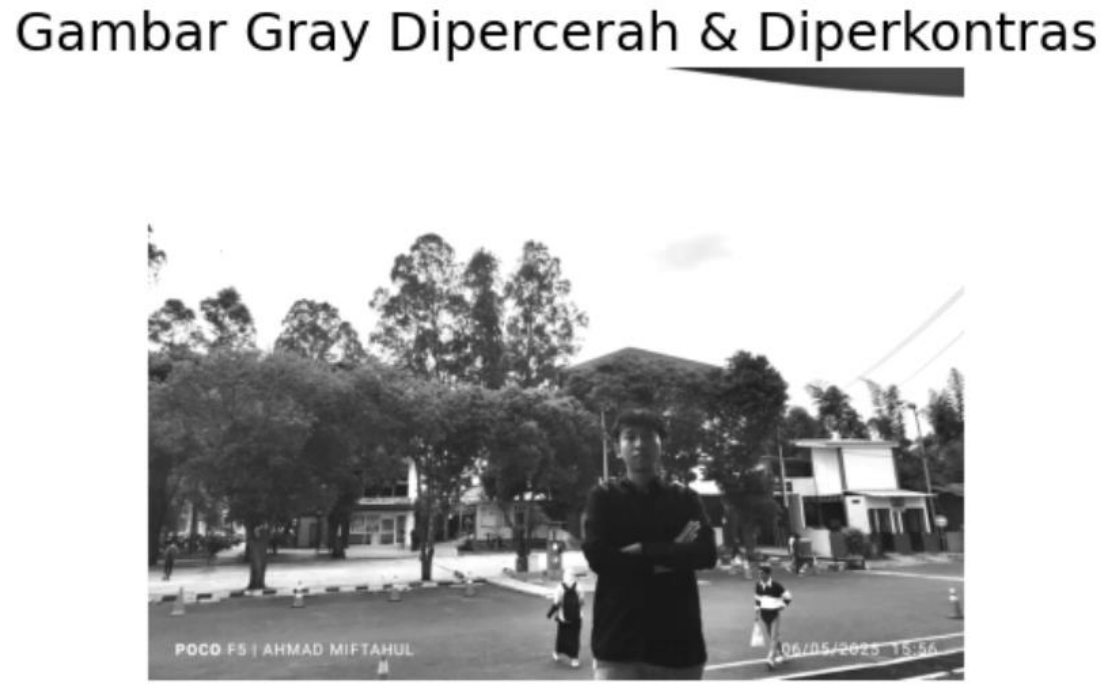
- Gambar Grey



- Gambar Gray yang dipercerah

- Gambar gray yang diperkontras

- Gambar gray yang dipercerah & diperkontras



- Penjelasan
  - Konversi ke Grayscale :
    - img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    - Gambar dikonversi ke grayscale (abu-abu), yaitu gambar tanpa warna, hanya tingkat kecerahan (0-255).
  - Pencerahan :
    - bright_gray = cv2.convertScaleAbs(img_gray, alpha=1, beta=50)
    - alpha = 1: tidak mengubah kontras
    - beta = 50: menambahkan nilai kecerahan (brightness) ke semua pixel Gambar menjadi lebih terang
  - Peningkatan Kontras :
    - contrast_gray = cv2.convertScaleAbs(img_gray, alpha=1.5, beta=0)
    - alpha = 1.5: meningkatkan kontras
    - beta = 0: tidak mengubah brightness
    - Gambar menjadi lebih tajam/kontras tinggi
  - Gabungan Pencerahan + Kontras
    - bright_contrast_gray = cv2.convertScaleAbs(img_gray, alpha=1.5, beta=50)
    - Kombinasi gambar yang lebih cerah dan lebih kontras

- Penjelasan Gambar
  1. Gambar Asli
     Merupakan gambar berwarna asli yang diperoleh langsung dari kamera. Warna-warna pada gambar ini masih utuh dan berfungsi sebagai pembanding sebelum dilakukan proses pengolahan citra lebih lanjut. Gambar ini masih mengandung tiga kanal warna (RGB), yaitu merah, hijau, dan biru.
  2. Gambar Gray
     Hasil konversi gambar berwarna ke dalam format grayscale (abu-abu). Pada tahap ini, informasi warna dihilangkan dan setiap piksel hanya menyimpan nilai intensitas cahaya antara 0 (hitam) hingga 255 (putih). Gambar grayscale sering digunakan dalam pengolahan citra karena lebih sederhana dan fokus pada struktur atau kontur objek.
  3. Gambar Gray yang Dipercerah
     Merupakan gambar grayscale yang telah ditambahkan nilai brightness (kecerahan). Setiap piksel dibuat lebih terang dengan menambahkan nilai konstan pada intensitasnya. Hasilnya, gambar tampak lebih cerah, terutama pada area yang semula gelap. Teknik ini biasa digunakan untuk memperjelas objek yang kurang terlihat akibat pencahayaan rendah.
  4. Gambar gray yang Diperkontras
     Pada gambar ini, tingkat kontras ditingkatkan sehingga perbedaan antara area terang dan gelap menjadi lebih mencolok. Piksel terang menjadi lebih terang, dan piksel gelap menjadi lebih gelap. Efek ini membuat garis batas dan detail pada gambar tampak lebih tegas, cocok untuk menonjolkan fitur visual atau kontur objek.
  5. Gambar gary Dipercerah & Diperkontras
     Ini adalah gabungan dari dua proses sebelumnya: pencerahan dan peningkatan kontras. Gambar menjadi tidak hanya lebih terang, tetapi juga memiliki kontras yang kuat. Hasilnya adalah gambar grayscale yang lebih informatif dan jelas, di mana detail dalam area gelap maupun terang dapat terlihat lebih nyata dan mencolok.

BAB IV

PENUTUP

Deteksi warna pada citra digital dapat dilakukan dengan memisahkan citra menjadi tiga channel warna dasar RGB (Red, Green, Blue). Melalui praktikum ini, telah berhasil diidentifikasi teks dengan warna berbeda ("AHMAD MIFTAHUL ARZA FIRISKY") dimana setiap bagian teks memiliki warna yang berbeda. Analisis histogram menunjukkan bahwa distribusi intensitas pada masing-masing channel memberikan informasi penting tentang dominasi warna dalam citra.

Penggunaan ambang batas (threshold) dalam segmentasi warna sangat memengaruhi hasil deteksi objek. Dari percobaan yang dilakukan, terlihat jelas perbedaan hasil segmentasi ketika menggunakan ambang batas tunggal (Blue), kombinasi (Red-Blue), hingga keseluruhan warna (Red-Green-Blue). Ambang batas yang tepat mampu mengidentifikasi warna target dengan akurat sambil meminimalkan noise dan false detection.

Model warna HSV terbukti lebih efektif untuk segmentasi warna dibandingkan model RGB, terutama ketika berhadapan dengan variasi pencahayaan. Dalam praktikum, penggunaan rentang nilai HSV yang spesifik untuk warna merah (Hue: 0-10 dan 170-180), hijau (Hue: 35-85), dan biru (Hue: 100-130) berhasil mendeteksi masing-masing bagian teks dengan cukup akurat.

Perbaikan kualitas citra dengan kondisi backlight dapat dilakukan melalui serangkaian teknik pengolahan citra seperti konversi ke grayscale, peningkatan kecerahan (brightness), dan peningkatan kontras. Hasil praktikum menunjukkan bahwa kombinasi teknik-teknik tersebut dapat secara signifikan meningkatkan visibilitas objek dalam citra backlight, memunculkan detail yang sebelumnya tidak terlihat.

Analisis histogram merupakan alat yang sangat berguna dalam memahami karakteristik citra. Dari histogram channel warna yang dihasilkan dalam praktikum, dapat diidentifikasi bahwa teks merah "FIRISKY" memberikan kontras lebih baik pada channel merah, sementara teks hijau dan biru cenderung menyatu dengan latar belakang pada channel masing-masing karena memiliki nilai intensitas yang tinggi.

DAFTAR PUSTAKA

[1] Firmansyah, D., & Permana, I. (2020). Implementasi Metode Contrast Stretching untuk Meningkatkan Kualitas Citra Digital. Jurnal Teknik Informatika dan Sistem Informasi, 6(1), 23-34.

[2] Hartono, R., Wahyudi, E., & Putri, A. R. (2022). Analisis Perbandingan Metode Segmentasi Warna HSV dan YCbCr untuk Deteksi Objek pada Sistem Computer Vision. Jurnal Teknologi Informasi dan Ilmu Komputer, 9(1), 127-136.

[3] Hidayatullah, P., & Hartati, S. (2020). Segmentasi Citra Digital Berbasis Clustering K-Means untuk Identifikasi Penyakit Daun Padi. Jurnal Ilmu Komputer dan Informatika, 6(2), 89-100.

[4] Mulyawan, H., Samsono, M. Z. H., & Setiawardhana. (2021). Peningkatan Kualitas Citra Menggunakan Teknik Histogram Equalization dan Contrast Limited Adaptive Histogram Equalization. Jurnal Nasional Teknik Elektro dan Teknologi Informasi, 10(1), 12-22.

[5] Nurraharjo, E. (2020). Implementasi Ruang Warna HSV untuk Segmentasi Warna pada Citra Digital. Jurnal Teknologi Informasi DINAMIK, 25(1), 23-32.

[6] Putra, I. K. G. D., Bhaskara, I. M. A., & Purnawan, I. K. A. (2020). Peningkatan Kualitas Citra Inframerah Menggunakan Kombinasi Metode Contrast Stretching dan Gaussian Filter. Jurnal Ilmiah Merpati (Menara Penelitian Akademika Teknologi Informasi), 8(3), 203-212.

[7] Putra, R. E., Sujaini, H., & Irwansyah, M. A. (2021). Segmentasi Otomatis Citra Warna Menggunakan K-Means Clustering dan Expectation-Maximization. Jurnal Coding, Sistem Komputer Untan, 9(1), 11-22.

[8] Wibowo, S. A., Hidayat, B., & Sunarya, U. (2021). Sistem Pendeteksi Tanaman Padi Terserang Penyakit Blast Menggunakan Segmentasi Warna HSV dan SVM. Jurnal Elektro dan Telekomunikasi Terapan, 8(1), 881-891.