UTS

PENGOLAHAN CITRA

NAMA : Hafidz Arif Budiono

NIM : 202331165

KELAS : F

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

NO.PC : 3

ASISTEN : 1. Sasikirana Ramadhanty Setiawan Putri

2. Rizqy Amanda

3. Ridho Chaerullah

4. Sakura Amastasya Salsabila Setiyanto

INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2024/2025

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Rumusan Masalah

1. Bagaimana cara mendeteksi warna merah, hijau, dan biru pada sebuah citra digital?
2. Bagaimana cara menganalisis dan menampilkan histogram dari citra berdasarkan deteksi warna?
3. Bagaimana menentukan nilai ambang batas (threshold) terkecil hingga terbesar untuk masing-masing warna pada citra?
4. Bagaimana teknik pengolahan citra dapat digunakan untuk memperbaiki gambar dengan kondisi backlight agar objek utama bisa terlihat jelas?

1.2 Tujuan Masalah

1. Mendeteksi warna merah, hijau, dan biru dalam citra digital dan menampilkan hasil segmentasi dari masing-masing warna.
2. Menganalisis perubahan citra setelah proses deteksi warna dan menginterpretasikan histogram dari masing-masing citra hasil deteksi.
3. Menentukan nilai ambang batas warna untuk membedakan kategori warna secara optimal pada citra.
4. Mengolah gambar backlight agar foto wajah menjadi lebih jelas dibandingkan latar belakang terang, dengan teknik konversi grayscale, peningkatan kecerahan, dan penyesuaian kontras.

1.3 Manfaat Masalah

1. Memberikan pemahaman praktis dalam penerapan deteksi warna menggunakan metode segmentasi warna dalam pengolahan citra digital.
2. Membantu mengasah kemampuan analisis visual dan statistik melalui interpretasi histogram warna.
3. Meningkatkan keterampilan dalam menentukan ambang batas warna yang efektif untuk klasifikasi objek dalam gambar.
4. Menyediakan solusi teknis terhadap masalah umum dalam fotografi digital, yaitu gambar yang terpengaruh oleh backlight, sehingga dapat meningkatkan kualitas visual dan fokus utama citra.

BAB II
LANDASAN TEORI

2.1 Pengolahan Citra Digital

Pengolahan citra digital merupakan suatu bidang dalam ilmu komputer yang berfokus pada pemrosesan gambar dalam format digital dengan menggunakan perangkat komputer. Gambar digital sendiri didefinisikan sebagai fungsi dua dimensi, f(x, y), di mana x dan y merepresentasikan koordinat spasial, sedangkan nilai dari fungsi tersebut pada titik (x, y) disebut sebagai intensitas atau tingkat keabuan. Ketika nilai x, y, dan intensitas tersebut bersifat terbatas dan diskrit, maka gambar tersebut disebut gambar digital.

Proses pengolahan citra digital melibatkan berbagai teknik untuk meningkatkan kualitas citra, mengekstrak informasi, maupun menginterpretasikan objek yang terdapat dalam citra. Setiap gambar digital tersusun atas elemen-elemen kecil yang dikenal sebagai piksel (picture elements), yang masing-masing memiliki lokasi dan nilai intensitas tertentu. Pengolahan citra digital meliputi proses-proses berlevel rendah hingga menengah, seperti perbaikan kualitas citra, segmentasi, dan pengenalan objek, dan dapat digunakan sebagai dasar untuk tahap lanjutan seperti analisis citra dan visi komputer.

2.2 Deteksi Warna RGB

Deteksi warna merupakan salah satu teknik dasar dalam pengolahan citra digital yang bertujuan untuk mengidentifikasi piksel-piksel berdasarkan komponen warnanya. Dalam ruang warna RGB (*Red, Green, Blue*), setiap piksel direpresentasikan oleh kombinasi tiga nilai intensitas, yaitu R (merah), G (hijau), dan B (biru), dengan masing-masing nilai biasanya berada dalam rentang 0 hingga 255. Deteksi warna RGB dilakukan dengan membandingkan nilai-nilai intensitas tersebut terhadap ambang batas tertentu yang telah ditentukan sebelumnya.

Secara umum, proses deteksi warna dimulai dengan pembacaan nilai RGB dari citra digital, kemudian dilakukan pemisahan (thresholding) terhadap masing-masing kanal warna. Sebagai contoh, untuk mendeteksi warna merah, maka nilai komponen R harus lebih tinggi dari komponen G dan B, dan berada dalam rentang ambang batas tertentu. Hal yang sama berlaku untuk deteksi warna hijau dan biru, dengan menyesuaikan dominansi nilai komponen warna yang relevan.

Deteksi warna RGB memiliki banyak aplikasi, di antaranya pelacakan objek, segmentasi citra, pengenalan warna dalam sistem robotik, dan sistem interaktif berbasis visual. Meskipun sederhana dan cepat, teknik ini sensitif terhadap pencahayaan dan kualitas sensor kamera, sehingga pada beberapa kasus perlu dilakukan konversi ke ruang warna lain (seperti HSV atau YCbCr) untuk mendapatkan hasil yang lebih stabil.

2.3 Histogram Pada Citra Digital

Histogram pada citra digital merupakan representasi grafik dari distribusi tingkat intensitas (atau keabuan) piksel dalam suatu citra. Histogram ini menggambarkan frekuensi kemunculan setiap tingkat intensitas, dari yang paling gelap hingga yang paling terang, sehingga memudahkan dalam memahami karakteristik visual dari citra tersebut. Terdapat dua jenis histogram yang umum digunakan, yaitu histogram tidak ternormalisasi yang menunjukkan jumlah piksel pada tiap tingkat intensitas tertentu, serta histogram ternormalisasi yang menyajikan probabilitas relatif kemunculan setiap intensitas dengan membagi jumlah piksel pada tiap tingkat dengan total jumlah piksel citra. Histogram sangat penting dalam pengolahan citra digital karena dapat digunakan untuk berbagai keperluan, seperti peningkatan kontras, deteksi objek, dan segmentasi citra. Bentuk dari histogram dapat mencerminkan kualitas visual citra; misalnya, citra gelap memiliki histogram yang terkonsentrasi di sisi kiri (intensitas rendah), sementara citra kontras tinggi menunjukkan distribusi intensitas yang merata di seluruh rentang intensitas.

2.4 Thresholding

Thresholding merupakan salah satu metode dasar dalam segmentasi citra digital yang digunakan untuk memisahkan objek dari latar belakang berdasarkan perbedaan nilai intensitas piksel. Teknik ini bekerja dengan menetapkan nilai ambang (threshold) tertentu. Piksel yang memiliki intensitas lebih tinggi dari nilai ambang diklasifikasikan sebagai objek, sementara yang lebih rendah diklasifikasikan sebagai latar belakang.

Jika nilai ambang diterapkan secara menyeluruh pada seluruh citra, maka disebut **global thresholding**. Namun, jika nilai ambang bervariasi tergantung pada kondisi lokal dalam citra, maka disebut **thresholding adaptif**. Metode ini sangat bergantung pada bentuk histogram intensitas citra. Sehingga, semakin jelas pemisahan antara objek dan latar belakang, semakin efektif proses thresholding dilakukan.

2.5 Morfologi GrayScale

Morfologi grayscale merupakan perluasan dari konsep morfologi biner yang diterapkan pada citra keabuan (grayscale image). Tujuan utama dari pendekatan ini adalah untuk menganalisis dan memodifikasi bentuk serta struktur dari objek dalam citra berdasarkan nilai intensitas piksel. Dalam morfologi grayscale, citra direpresentasikan sebagai fungsi dua variabel f(x,y)f(x, y)f(x,y), dengan setiap titik koordinat (x,y)(x, y)(x,y) memiliki nilai intensitas tertentu, dan elemen struktural dilambangkan sebagai b(x,y)b(x, y)b(x,y).

Operasi dasar dalam morfologi grayscale meliputi **dilasi**, **erosi**, **pembukaan (opening)**, dan **penutupan (closing)**. Sama seperti pada morfologi biner, elemen struktural digunakan sebagai alat ukur untuk memeriksa dan memodifikasi area lokal dalam citra. Namun, pada morfologi grayscale, intensitas piksel juga diperhitungkan, sehingga hasil transformasi mempertimbangkan baik bentuk maupun nilai terang-gelap dari area yang dianalisis.

Terdapat dua jenis elemen struktural dalam morfologi grayscale, yaitu:

1. **Elemen struktural datar (flat structuring element)**: merupakan elemen yang tidak memiliki perbedaan intensitas di dalamnya. Jenis ini paling sering digunakan karena kemudahan implementasinya secara digital.

2. **Elemen struktural tidak datar (non-flat structuring element)**: merupakan elemen dengan variasi intensitas, seperti bentuk setengah bola (hemisfer), yang memungkinkan analisis bentuk yang lebih kompleks namun kurang umum digunakan dalam praktik karena keterbatasan komputasi.

Dalam praktiknya, elemen struktural umumnya bersifat simetris dan memiliki titik pusat sebagai titik asal (origin). Operasi-operasi morfologi pada citra grayscale menggunakan pendekatan refleksi elemen struktural serta prinsip minimum dan maksimum lokal untuk mengubah nilai intensitas piksel berdasarkan lingkungan sekitarnya.

2.6 Fotografi Backlight

Fotografi backlight merupakan teknik visualisasi gambar fotografi dengan sumber pencahayaan yang mengarah langsung ke kamera. Dalam kondisi ini, cahaya berada di belakang objek utama, menyebabkan bagian depan objek tampak gelap atau tertutup bayangan. Umumnya, citra yang dihasilkan memiliki dynamic range yang buruk, yaitu rentang antara parameter shadow (bayangan) dan highlight (cahaya terang) sangat sempit. Akibatnya, detail pada area gelap maupun terang menjadi sulit terlihat secara seimbang.

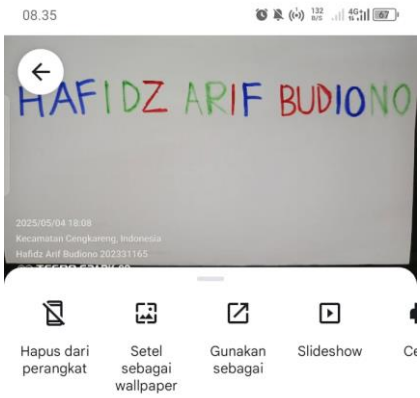
BAB III

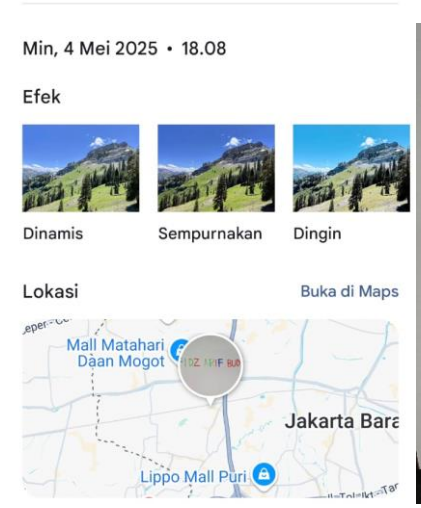
HASIL

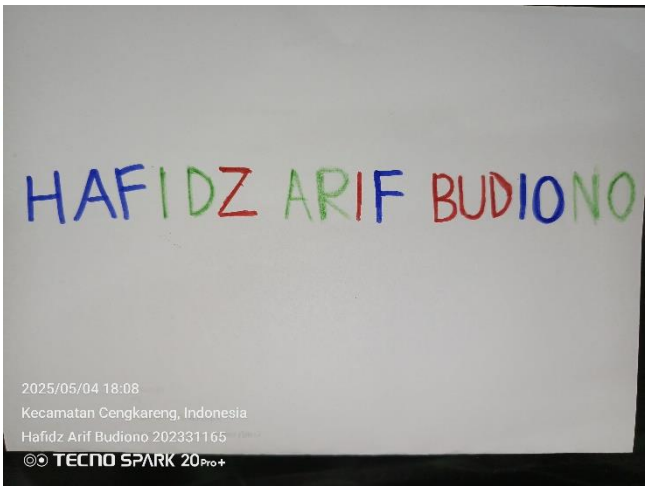
3.1 Deteksi Warna Pada Citra

3.1.1 Gambar yang Digunakan

Gambar yang digunakan dalam penelitian ini merupakan foto nama saya dengan komposisi objek yang memiliki warna dominan merah, biru, dan hijau. Citra tersebut dipilih karena memiliki variasi warna yang kontras sehingga dapat mempermudah proses segmentasi warna menggunakan metode pemisahan kanal.







3.1.2 Penjelasan Program

```
In [1]: import cv2
        import numpy as np
        import matplotlib.pyplot as plt
```

Baris ini mengimpor tiga pustaka penting. cv2 adalah pustaka OpenCV untuk pengolahan citra, numpy digunakan untuk operasi numerik dan manipulasi array, sedangkan matplotlib.pyplot digunakan untuk visualisasi citra dan histogram.

```
In [2]: img = cv2.imread('gambar_nama.jpg')
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

Saya membaca file gambar_nama.jpg. awalnya file ini disimpan dalam format BGR, kemudian saya membuat variabel img_rgb yang mana nilai menyimpan img yang diubah ke RGB

```
In [3]: red_channel = img_rgb[:, :, 2]
        green_channel = img_rgb[:, :, 1]
        blue_channel = img_rgb[:, :, 0]
```

Pada bagian ini digunakan untuk memisahkan masing-masing kanal warna dari citra RGB. Kanal merah diakses melalui indeks ke-2, kanal hijau melalui indeks ke-1, dan kanal biru melalui indeks ke-0.

```
In [4]: fig, axs = plt.subplots(2, 4, figsize=(20, 10))
```

Kemudian saya ingin menampilkannya dalam sebuah figure (kanvas) dengan ukuran 20 x 10 inci, yang terdiri dari 2 baris dan 4 kolom subgrafik.

```
        axs[0, 0].imshow(img_rgb)
        axs[0, 0].set_title('CITRA KONTRAS')
        axs[0, 0].axis('off')
```

Axs itu menggambar lokasi penempatannya

Subplot pada baris pertama dan kolom pertama (axs[0, 0]) saya menampilkan citra RGB.

Judul subplot diset menjadi 'CITRA KONTRAS'

sumbu koordinat dimatikan agar tampilan lebih bersih.

```
axs[1, 0].hist(img_rgb.ravel(), bins=256, range=[0, 256])
axs[1, 0].set_title('Histogram Citra Kontras')
```

Subplot pada baris kedua dan kolom pertama (axs[1, 0]) menampilkan histogram dari keseluruhan nilai piksel pada citra RGB.

Metode .ravel() digunakan untuk meratakan array menjadi satu dimensi.

Histogram menggunakan 256 bin, mewakili rentang intensitas 0 hingga 255.

Judul subplot diset menjadi ‘Histogram Citra Kontras’

```
axs[0, 1].imshow(blue_channel, cmap='gray')
axs[0, 1].set_title('BIRU')
```

Subplot pada baris pertama dan kolom kedua (axs[0, 1]) menampilkan kanal biru dalam skala keabuan (cmap='gray') agar kontras dapat terlihat lebih jelas.

Judul subplot diset menjadi 'BIRU'.

```
axs[1, 1].hist(blue_channel.ravel(), bins=256, range=[0, 256], color='blue')
axs[1, 1].set_title('Histogram Biru')
```

Subplot pada baris kedua dan kolom kedua (axs[1, 1]) menampilkan histogram dari nilai intensitas kanal biru. Histogram diberi warna biru agar distribusi intensitas biru lebih kelihatan

Histogram menggunakan 256 bin, mewakili rentang intensitas 0 hingga 255.

Judul subplot diset menjadi 'Histogram Biru'.

```
axs[0, 2].imshow(red_channel, cmap='gray')
axs[0, 2].set_title('MERAH')
```

Subplot pada baris pertama dan kolom ketiga (axs[0, 2]) menampilkan kanal merah dalam skala keabuan (cmap='gray') agar kontras dapat terlihat lebih jelas.

Judul subplot diset menjadi 'MERAH'.

```
axs[1, 2].hist(red_channel.ravel(), bins=256, range=[0, 256], color='red')
axs[1, 2].set_title('Histogram Merah')
```

Subplot pada baris kedua dan kolom ketiga (axs[1, 2]) menampilkan histogram dari nilai intensitas kanal merah. Histogram diberi warna merah agar distribusi intensitas merah lebih kelihatan

Histogram menggunakan 256 bin, mewakili rentang intensitas 0 hingga 255.

Judul subplot diset menjadi 'Histogram Merah'.

```
axs[0, 3].imshow(green_channel, cmap='gray')
axs[0, 3].set_title('HIJAU')
```

Subplot pada baris pertama dan kolom keempat (axs[0, 3]) menampilkan kanal hijau dalam skala keabuan (cmap='gray') agar kontras dapat terlihat lebih jelas.

Judul subplot diset menjadi 'HIJAU'.

```
axs[1, 3].hist(green_channel.ravel(), bins=256, range=[0, 256], color='green')
axs[1, 3].set_title('Histogram Hijau')
```

Subplot pada baris kedua dan kolom keempat (axs[1, 3]) menampilkan histogram dari nilai intensitas kanal hijau. Histogram diberi warna hijau agar distribusi intensitas hijau lebih kelihatan

Histogram menggunakan 256 bin, mewakili rentang intensitas 0 hingga 255.

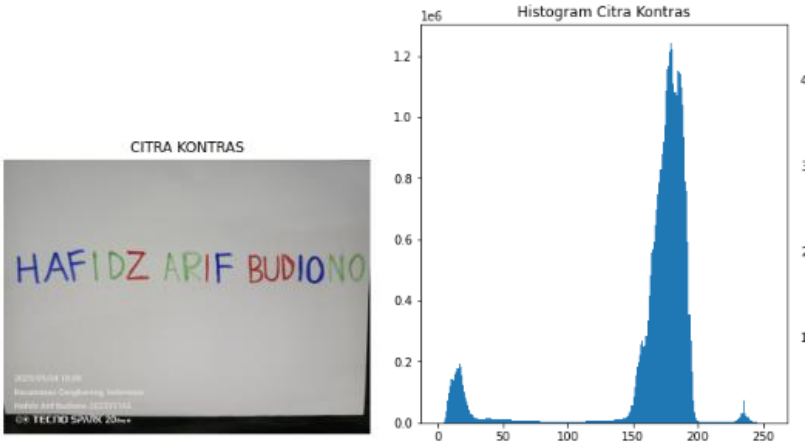
Judul subplot diset menjadi 'Histogram Hijau'.

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

Baris ini mengatur tata letak subplot agar tidak saling bertumpuk atau terlalu rapat, dan kemudian menampilkan keseluruhan grafik.

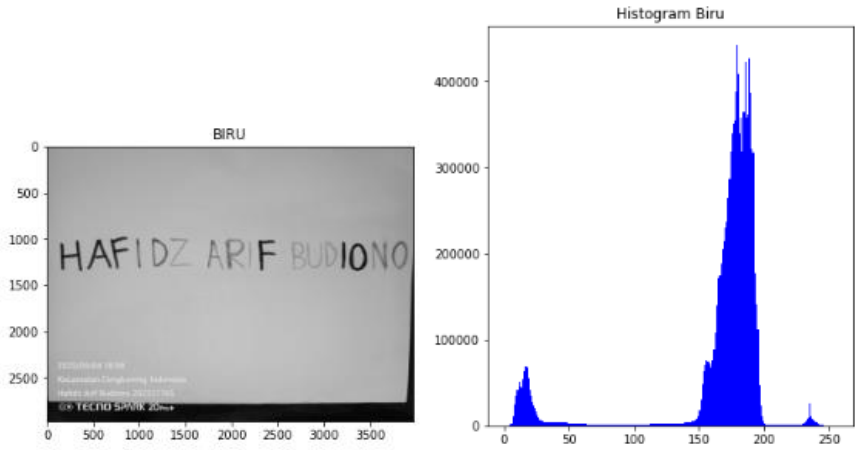
3.1.3 Analisa Hasil

Citra Kontras



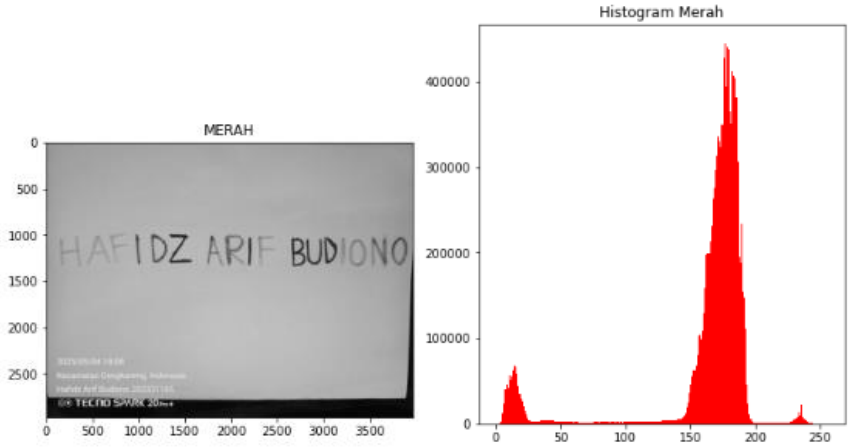
Pada Histogram Citra Kontras, terlihat dua puncak yaitu pada rentang nilai 10 – 30 dan 170 – 200. Pada histogram, arah horizontal (sumbu x) menunjukkan nilai intensitas piksel dari 0 hingga 255. 0 berarti hitam/gelap dan 255 berarti putih/terang. Sedangkan, Arah vertikal (sumbu y) menunjukkan jumlah piksel yang memiliki intensitas tersebut. Disini terlihat puncak kanan (170 – 200) lebih tinggi dibanding yang kiri (10 – 30). Menunjukkan bahwa area pada gambar yang digunakan banyak mendapat cahaya.

Biru



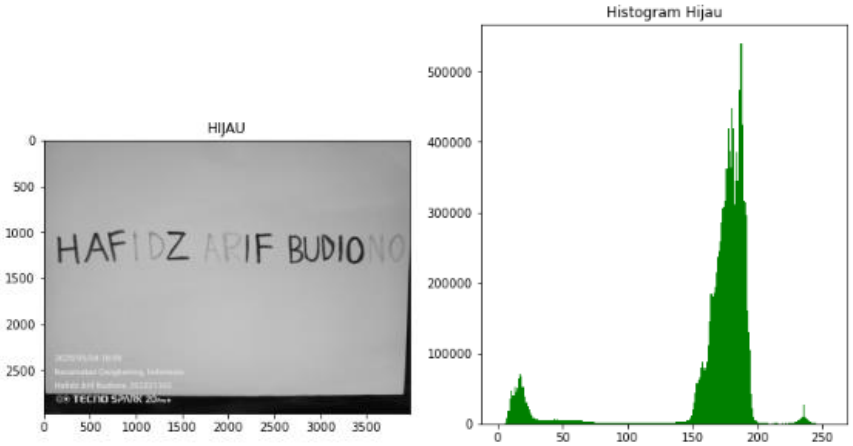
Pada histogram biru, Histogram ini memiliki puncak tajam di antara nilai 170–210, menandakan dominasi warna biru di area terang. Hampir tidak ada piksel biru di area gelap (intensitas < 50), berarti biru hanya muncul di bagian gambar yang terang, bukan pada bayangan gambar.

Merah



Pada histogram merah, Puncaknya sedikit bergeser ke kiri dibandingkan biru, yaitu di antara 160–200, yang menunjukkan intensitas merah sedikit lebih redup dibandingkan biru. Kurva distribusinya mirip biru, tapi tidak setajam biru. Ini bisa menunjukkan bahwa komponen merah hadir, tapi tidak dominan.

Hijau

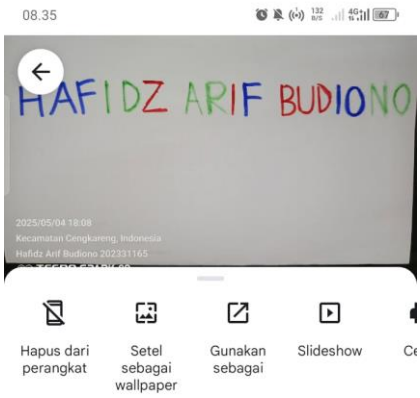


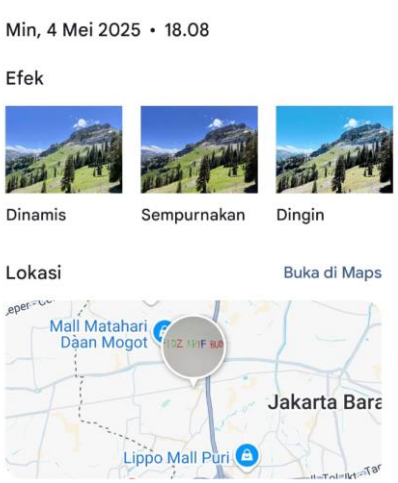
Pada histogram hijau Ini memiliki puncak dominan di sekitar 190–210, tetapi penyebaran intensitasnya lebih lebar. Ada cukup banyak piksel hijau di berbagai tingkat intensitas, termasuk sedikit di area gelap dan tengah. Ini menunjukkan bahwa warna hijau lebih tersebar di seluruh gambar, tidak hanya terkonsentrasi di area terang. Ini karena spidol hijau yang saya gunakan pudar.

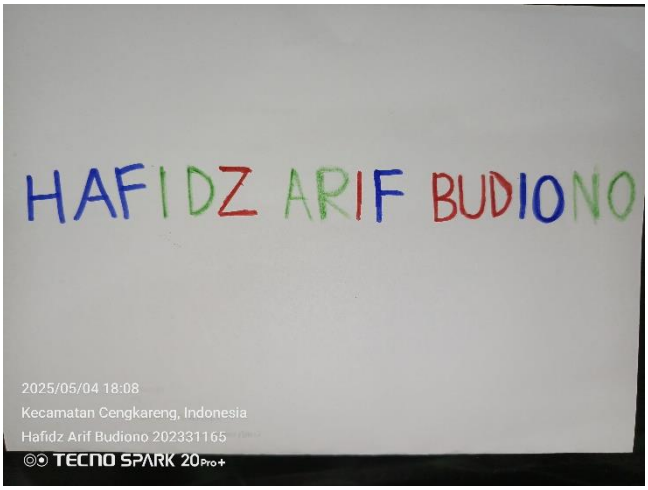
3.2 Ambang Batas

3.2.1 Gambar yang Digunakan

Gambar yang digunakan dalam penelitian ini merupakan foto nama saya yang mengandung objek berwarna merah, biru, dan hijau. Pemilihan gambar ini bertujuan untuk memudahkan proses segmentasi warna dengan metode masking pada ruang warna HSV.







3.2.2 Penjelasan Program

```
In [1]: import cv2
        import numpy as np
        import matplotlib.pyplot as plt
```

Baris ini mengimpor tiga pustaka penting. cv2 adalah pustaka OpenCV untuk pengolahan citra, numpy digunakan untuk operasi numerik dan manipulasi array, sedangkan matplotlib.pyplot digunakan untuk visualisasi citra dan histogram.

```
In [2]: img = cv2.imread('gambar_nama.jpg')
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        img_hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
```

Saya membaca file gambar_nama.jpg. awalnya file ini disimpan dalam format BGR, saya membuat variabel img_rgb yang mana nilai menyimpan img yang diubah ke RGB. Dan kemudian membuat variabel img_hsv untuk menyimpan nilai img yang diubah ke HSV, ini agar memudahkan proses segmentasi warna, karena HSV memisahkan informasi warna (hue) dari intensitas.

```
In [3]: mask_red = cv2.inRange(img_hsv, (0,100,100), (10,255,255)) | cv2.inRange(img_hsv, (160,100,100), (180,255,255))
```

Dalam konteks pemrosesan citra digital, thresholding adalah proses segmentasi yang bertujuan untuk memisahkan objek dari latar belakang berdasarkan intensitas piksel. Pada program ini, thresholding dilakukan bukan terhadap skala keabuan (grayscale), melainkan terhadap komponen warna dalam ruang HSV menggunakan fungsi cv2.inRange().

Pada ln 3 ini merupakan implementasi dari thresholding biner berbasis warna, di mana nilai minimum dan maksimum dari kanal HSV ditentukan sebagai batas threshold. Piksel yang nilai HSV-nya berada di dalam rentang tersebut akan diberi nilai 255 (putih), sedangkan piksel yang di luar rentang akan diberi nilai 0 (hitam).

Dalam hal ini, dua rentang digunakan untuk mendeteksi warna merah karena distribusinya berada di kedua ujung spektrum hue (0–10 dan 160–180 derajat).

```
mask_blue = cv2.inRange(img_hsv, (100,100,100), (140,255,255))
```

Untuk mask_blue, rentang HSV yang digunakan adalah (100,100,100) hingga (140,255,255). yang sesuai dengan kisaran hue warna biru. Pemilihan rentang ini bertujuan agar hanya piksel dengan warna biru yang cukup kuat dan terang yang dapat lolos dari proses thresholding.

```
mask_green = cv2.inRange(img_hsv, (40,100,100), (80,255,255))
```

Untuk mask_green, rentang HSV yang digunakan adalah (40,100,100) hingga (80,255,255), yang sesuai dengan kisaran hue warna hijau. Pemilihan rentang ini bertujuan agar hanya piksel dengan warna hijau yang cukup kuat dan terang yang dapat lolos dari proses thresholding.

```
In [4]: mask_red_blue = mask_red | mask_blue
        mask_all = mask_red | mask_blue | mask_green
```

Variabel mask_red_blue dibuat menggabungkan mask merah dan biru, serta mask_all untuk menggabungkan merah, biru, dan hijau.

Gabungan ini digunakan untuk menganalisis bagaimana histogram berubah ketika lebih dari satu warna disegmentasi.

```
In [5]: result_red = cv2.bitwise_and(img_rgb, img_rgb, mask=mask_red)
        result_red_blue = cv2.bitwise_and(img_rgb, img_rgb, mask=mask_red_blue)
        result_all = cv2.bitwise_and(img_rgb, img_rgb, mask=mask_all)
```

Disini dengan menggunakan operasi bitwise AND, hasil citra img_rgb disaring berdasarkan mask. Piksel yang sesuai dengan masker warna akan tetap ada, sementara piksel lainnya akan menjadi hitam. Hal ini supaya yang terlihat adalah warna yang sudah ditentukan.

```
In [6]: fig, axs = plt.subplots(3, 2, figsize=(12, 12))
```

Kemudian saya ingin menampilkannya dalam sebuah figure (kanvas) dengan ukuran 20 x 10 inci, yang terdiri dari 2 baris dan 4 kolom subgrafik.

```
axs[0, 0].imshow(result_red)
axs[0, 0].set_title('Hanya Merah')
```

Subplot pada baris pertama dan kolom pertama (axs[0, 0]) saya menampilkan hasil segmentasi warna merah.

Judul subplot diset menjadi ‘Hanya Merah’

```
for i, col in enumerate(['r', 'g', 'b']):
    axs[0, 1].plot(cv2.calcHist([img_rgb], [i], mask_red, [256], [0, 256]), color=col)
axs[0, 1].set_title('Histogram Merah')
```

Menghitung dan menampilkan histogram nilai intensitas dari citra RGB yang difilter oleh mask_red. Histogram dihitung untuk masing-masing kanal warna (r, g, b) menggunakan fungsi cv2.calcHist, namun hanya pada piksel-piksel yang masuk dalam masker merah.

Judul subplot diset menjadi 'Histogram Merah'.

```
axs[1, 0].imshow(result_red_blue)
axs[1, 0].set_title('Merah dan Biru')
```

Subplot pada baris kedua dan kolom pertama (axs[1, 0]) saya menampilkan hasil segmentasi warna merah dan biru.

Judul subplot diset menjadi ‘Merah dan Biru’

```
for i, col in enumerate(['r', 'g', 'b']):
    axs[1, 1].plot(cv2.calcHist([img_rgb], [i], mask_red_blue, [256], [0, 256]), color=col)
axs[1, 1].set_title('Histogram Merah dan Biru')
```

Menghitung dan menampilkan histogram nilai intensitas dari citra RGB yang difilter oleh mask_red_blue. Histogram dihitung untuk masing-masing kanal warna (r, g, b) menggunakan fungsi cv2.calcHist, namun hanya pada piksel-piksel yang masuk dalam masker merah.

Judul subplot diset menjadi 'Histogram Merah dan Biru'.

```
axs[2, 0].imshow(result_all)
axs[2, 0].set_title('Merah, Biru, dan Hijau')
```

Subplot pada baris ketiga dan kolom pertama (axs[2, 0]) saya menampilkan hasil segmentasi warna merah dan biru.

Judul subplot diset menjadi ‘Merah, Biru, dan Hijau’

```
for i, col in enumerate(['r', 'g', 'b']):
    axs[2, 1].plot(cv2.calcHist([img_rgb], [i], mask_all, [256], [0, 256]), color=col)
axs[2, 1].set_title('Histogram Semua Warna')
```

Menghitung dan menampilkan histogram nilai intensitas dari citra RGB yang difilter oleh mask_all. Histogram dihitung untuk masing-masing kanal warna (r, g, b) menggunakan fungsi cv2.calcHist, namun hanya pada piksel-piksel yang masuk dalam masker merah.

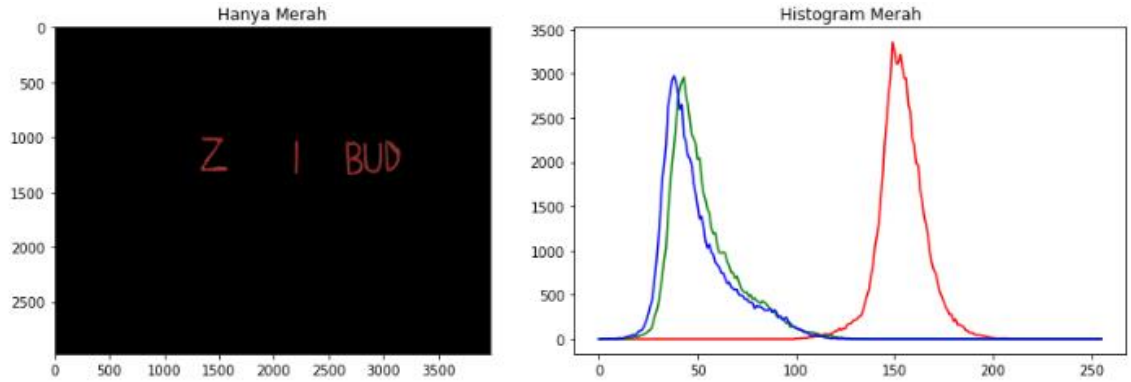
Judul subplot diset menjadi ‘Histogram Semua Warna

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

Baris ini mengatur tata letak subplot agar tidak saling bertumpuk atau terlalu rapat, dan kemudian menampilkan keseluruhan grafik.

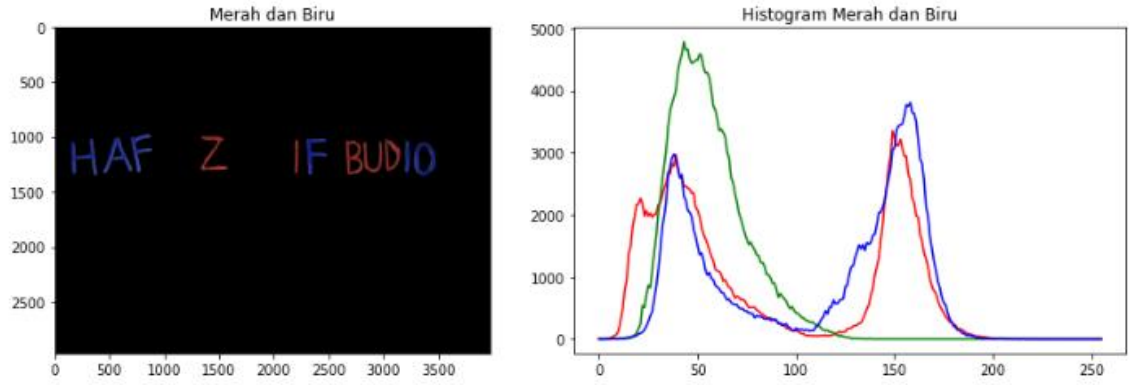
3.2.3 Analisis Hasil

Merah



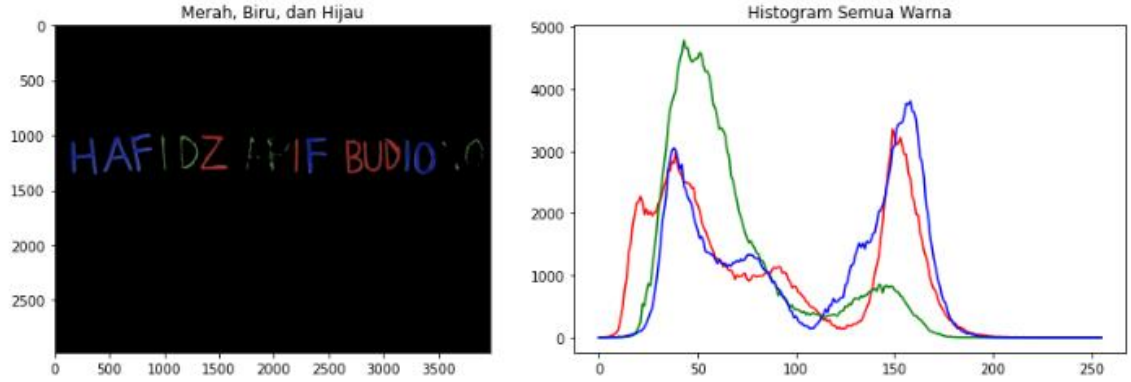
Pada histogram merah, puncak dominannya pada kanal merah di sekitar nilai intensitas 150–160. Kanal hijau dan biru berada di kiri menunjukkan hampir tidak ada kontribusi piksel (gelap). Distribusi sempit artinya warna merah cukup homogen.

Merah dan Biru



Pada histogram merah dan biru, intensitas merah sekitar 150 dan biru sekitar 140 – 150. Kanal hijau meningkat sedikit, menunjukkan ada beberapa kontribusi hijau dari area biru. Mungkin karena spidol biru saya memiliki komponen hijau sedikit.

Merah, Biru, dan Hijau

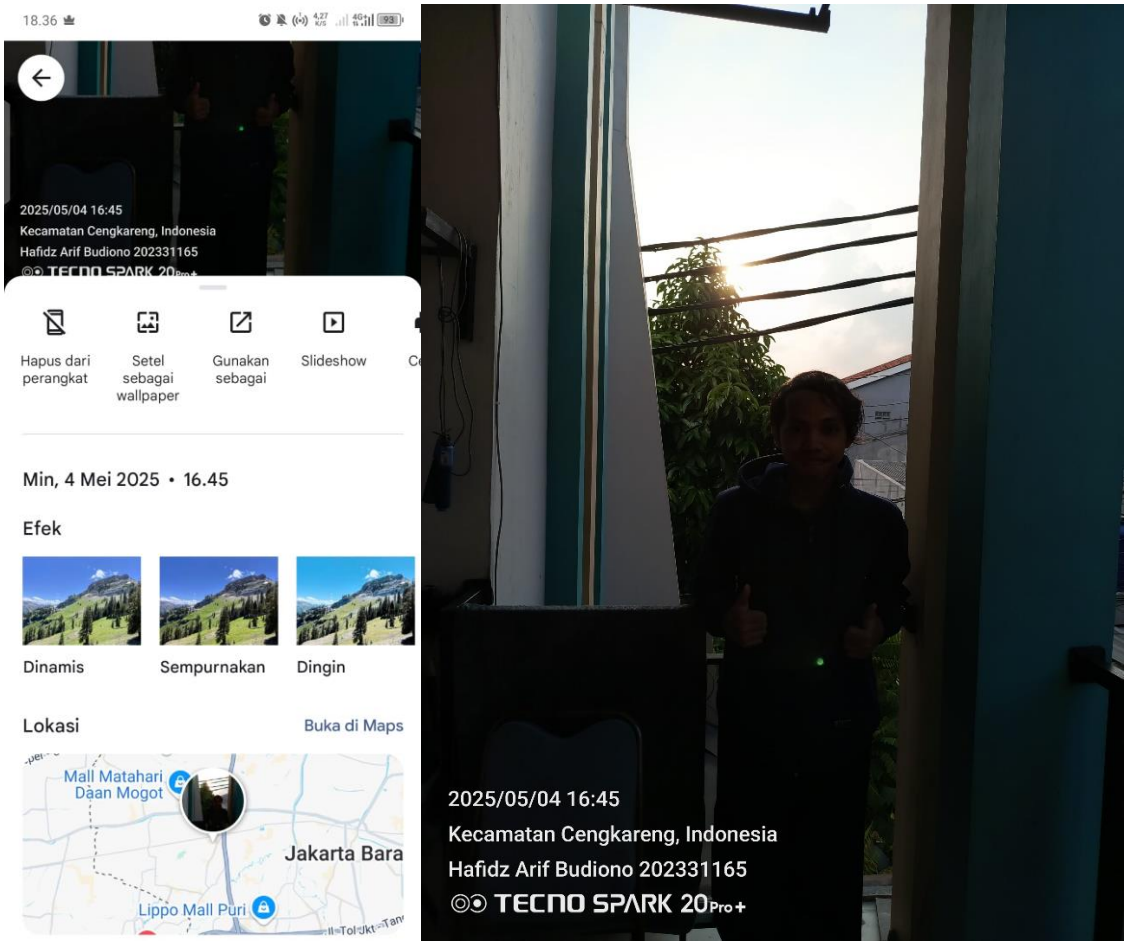


Pada histogram merah, biru, dan hijau. Hijau sangat dominan pada nilai intensitas rendah hingga menengah (sekitar 50). Lalu Biru memiliki puncak tinggi sekitar 140–150, dan Merah juga tampak cukup kuat, tapi tidak mendominasi seperti pada histogram pertama. Ini adalah histogram dengan distribusi warna paling luas karena mencakup ketiga warna utama. Kelihatan tumpang tindih antar kanal karena mungkin beberapa area mengandung warna campuran.

3.3 Memperbaiki Gambar Backlight

3.3.1 Gambar yang Digunakan

Gambar yang digunakan dalam percobaan ini adalah sebuah foto yang mengalami kondisi backlight, yaitu kondisi di mana sumber cahaya berada di belakang objek utama sehingga menyebabkan objek tampak gelap dan detailnya sulit terlihat. Gambar tersebut disimpan dengan nama file gambar_backlight.jpg. Gambar ini dipilih secara khusus untuk menguji efektivitas metode peningkatan kualitas citra melalui manipulasi tingkat kecerahan dan kontras.



3.3.2 Penjelasan Program

```
In [1]: import cv2
        import matplotlib.pyplot as plt
        import numpy as np
```

Baris ini mengimpor tiga pustaka penting. cv2 adalah pustaka OpenCV untuk pengolahan citra, numpy digunakan untuk operasi numerik dan manipulasi array, sedangkan matplotlib.pyplot digunakan untuk visualisasi citra dan histogram.

```
In [2]: img = cv2.imread('gambar_backlight.jpg')
```

Saya membaca file gambar_nama.jpg. awalnya file ini disimpan dalam format BGR, saya membuat variabel img_rgb yang mana nilai menyimpan img yang diubah ke RGB.

```
In [3]: img_asli_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

Awalnya file ini disimpan dalam format BGR, saya membuat variabel img_rgb_asli yang mana nilai menyimpan img yang diubah ke RGB.

```
plt.figure(figsize=(18,12))
plt.imshow(img_asli_rgb)
plt.title("Gambar Asli")
plt.xticks([]), plt.yticks([])
plt.show()
```

Bagian ini digunakan untuk menampilkan citra asli dalam format RGB.

Ukuran gambar ditentukan sebesar 18x12 inci.

Judul diberikan sebagai “Gambar Asli”.

sumbu x dan y disembunyikan supaya lebih rapi.

```
In [4]: gray = cv2.cvtColor(img_asli_rgb, cv2.COLOR_BGR2GRAY)
```

Saya membuat variabel baru, gray yang menyimpan nilai img_asli_rgb yang dikonversi ke dalam citra grayscale.

```
plt.figure(figsize=(18,12))
plt.imshow(gray, cmap='gray')
plt.title("Gambar Grayscale")
plt.xticks([]), plt.yticks([])
plt.show()
```

Bagian ini digunakan untuk menampilkan Citra hasil konversi grayscale.

Ukuran gambar ditentukan sebesar 18x12 inci.

Ditampilkan menggunakan colormap gray agar gambar enak dilihat dalam skala keabuan.

Judul diberikan sebagai “Gambar Grayscale”.

sumbu x dan y disembunyikan supaya lebih rapi.

.

```
In [5]: bright_gray = cv2.convertScaleAbs(gray, alpha=1, beta=100)
```

Saya membuat variabel baru, bright_gray yang menyimpan nilai gray yang ditingkatkan tingkat kecerahannya dengan menambahkan nilai beta=100 ke setiap piksel, tanpa mengubah kontras (karena alpha=1).

```
plt.figure(figsize=(18,12))
plt.imshow(bright_gray, cmap='gray')
plt.title("Grayscale + Cerah")
plt.xticks([]), plt.yticks([])
plt.show()
```

Bagian ini digunakan untuk menampilkan gambar grayscale yang telah ditingkatkan kecerahannya.

Ukuran gambar ditentukan sebesar 18x12 inci.

Ditampilkan menggunakan colormap gray agar gambar enak dilihat dalam skala keabuan.

Judul diberikan sebagai “Gambar Grayscale + Cerah”.

sumbu x dan y disembunyikan supaya lebih rapi.

```
In [6]: contrast_gray = cv2.convertScaleAbs(gray, alpha=3, beta=0)
```

Saya membuat variabel baru, contrast_gray yang menyimpan nilai gray yang meningkatkan kontras dengan mengalikan setiap nilai piksel dengan faktor alpha=3, tanpa menambahkan kecerahan tambahan (beta=0).

```
plt.figure(figsize=(18,12))
plt.imshow(contrast_gray, cmap='gray')
plt.title("Grayscale + Kontras")
plt.xticks([]), plt.yticks([])
plt.show()
```

Bagian ini digunakan untuk menampilkan gambar grayscale yang telah diperkuat kontrasnya .

Ukuran gambar ditentukan sebesar 18x12 inci.

Ditampilkan menggunakan colormap gray agar gambar enak dilihat dalam skala keabuan.

Judul diberikan sebagai “Gambar Grayscale + Kontras”.

sumbu x dan y disembunyikan supaya lebih rapi.

```
In [7]: contrast_bright_gray = cv2.convertScaleAbs(gray, alpha=3, beta=100)
```

Saya membuat variabel baru, contrast_bright_gray yang menyimpan nilai gray yang proses nya menggabungkan peningkatan kontras (alpha=3) dan kecerahan (beta=100) secara bersamaan untuk menghasilkan citra grayscale yang lebih tajam dan terang.

```
plt.figure(figsize=(18,12))
plt.imshow(contrast_bright_gray, cmap='gray')
plt.title("Grayscale + Cerah + Kontras")
plt.xticks([]), plt.yticks([])
plt.show()
```

Bagian ini digunakan untuk menampilkan citra hasil akhir dengan kombinasi peningkatan kontras dan kecerahan ditampilkan sebagai bentuk visualisasi akhir dari rangkaian proses peningkatan kualitas gambar.

Ukuran gambar ditentukan sebesar 18x12 inci.

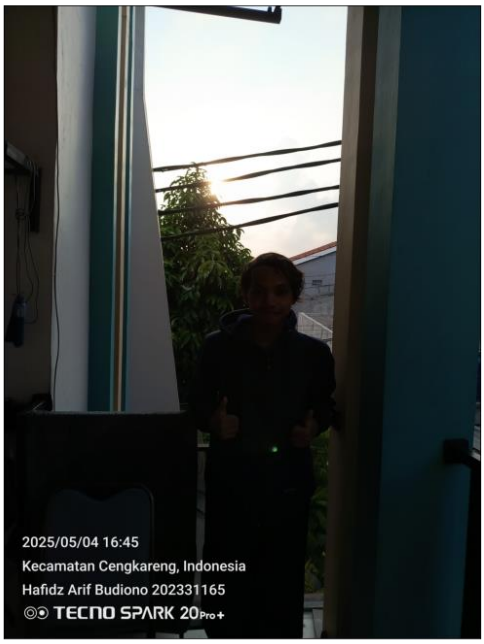
Ditampilkan menggunakan colormap gray agar gambar enak dilihat dalam skala keabuan.

Judul diberikan sebagai “Gambar Grayscale + Cerah + Kontras”.

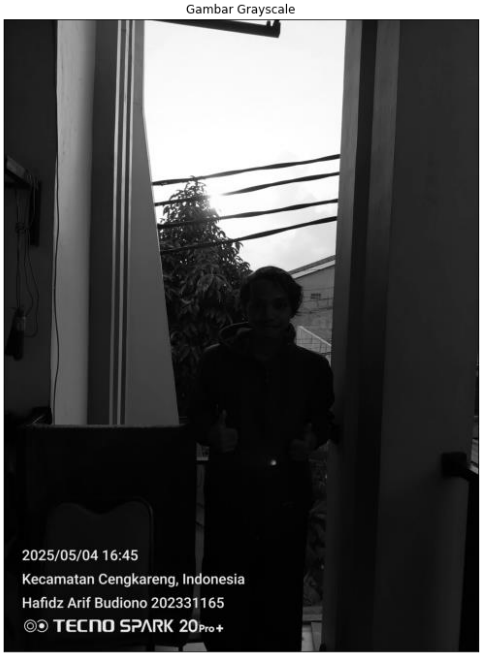
sumbu x dan y disembunyikan supaya lebih rapi.

3.3.3 Analisa Hasil

Gambar Asli

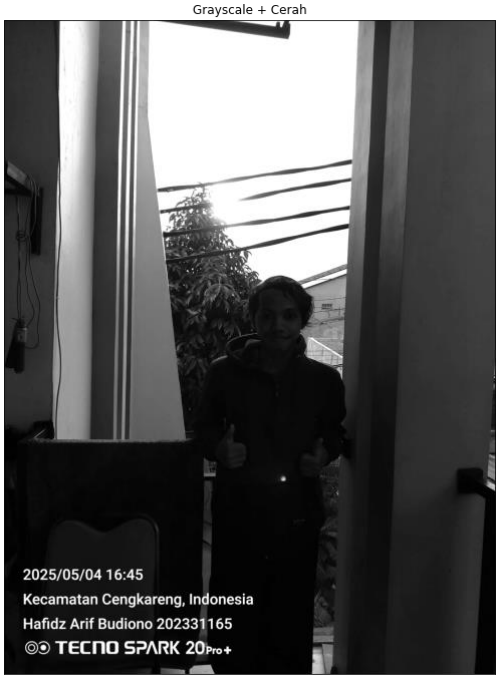


Gambar Grayscale



Pada Gambar Asli, Gambar ditampilkan dalam warna asli sesuai dengan warna aslinya. Mengandung informasi warna lengkap (merah, hijau, biru). Tampilan wajah tidak dapat dilihat dengan baik. Pada grayscale, gambar dikonversi menjadi hitam-putih berdasarkan intensitas cahaya.

Grayscale + Cerah



Grayscale + Kontras

Pada gambar Grayscale + Cerah, Gambar grayscale ditingkatkan kecerahannya dengan menambahkan nilai tetap (100) ke setiap piksel. Tampilan wajah sedikit lebih terlihat dibanding grayscale.

Pada gambar Grayscale + Kontras, Gambar grayscale diperkuat kontrasnya dengan mengalikan intensitas piksel. Disini lebih terlihat menonjolkan perbedaan antar objek. Tampilan wajah sedikit lebih terlihat dibanding Grayscale + Cerah.

Hasil akhir disini berupa Grayscale + Cerah + Kontras (alpha=3, beta=100), Kombinasi peningkatan kontras dan kecerahan sekaligus. Wajah pada gambar sangat bisa dilihat dibanding gambar asli, namun detail di beberapa area ada yang hilang seperti objek kabel listrik dan pohon pada gambar.

BAB IV
PENUTUP

4.1 Kesimpulan

Dari tugas ini saya memahami pengolahan citra digital merupakan bidang penting yang mempelajari cara memanipulasi dan menganalisis gambar menggunakan komputer, dengan berbagai teknik seperti deteksi warna, thresholding, analisis histogram, dan morfologi grayscale. Saya juga jadi paham bagaimana cara meningkatkan kualitas visual citra, memisahkan objek dari latar belakang, serta mengekstrak informasi dari gambar secara efisien. Selain itu, cara kerja brightness dan contrast dapat memperbaiki gambar backlight jadi menambah ilmu baru dalam menangani permasalahan umum dalam fotografi atau citra digital sehari-hari.

Penguasaan materi ini juga sangat bermanfaat karena menjadi dasar dari banyak aplikasi di dunia nyata, seperti sistem pengenalan wajah, pemantauan visual, pencitraan medis, hingga pengembangan teknologi berbasis kecerdasan buatan dan visi komputer. Selain memperkuat keterampilan teknis, pemahaman konsep-konsep ini juga melatih kemampuan berpikir logis dan analitis dalam menyelesaikan masalah berbasis data visual.

DAFTAR PUSTAKA

Gonzalez, R. C., & Woods, R. E. (2018). *Digital Image Processing* (4th ed.). Pearson Education.

Nurfitriani, R., & Arifianto, D. (2023). Deteksi Kemiripan Citra Batik Menggunakan Metode Color Moment dan Local Binary Pattern. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 7(12), 5467–5474.