

UTS
PENGOLAHAN CITRA



NAMA : Rafifah Nur Rifiyati

NIM : 202331003

KELAS : B

DOSEN : Ir. Darma Rusjdi, M. Kom

NO.PC : 13

ASISTEN : 1. Davina Najwa Ermawan
2. Fakhrol Fauzi Nugraha Tarigan
3. Viana Salsabila Fairuz Syahla
4. Muhammad Hanief Febriansyah

INSTITUT TEKNOLOGI PLN
TEKNIK INFORMATIKA
2024/2025

DAFTAR ISI

DAFTAR ISI	2
BAB I PENDAHULUAN	3
1.1 Rumusan Masalah	3
1.2 Tujuan Masalah	3
1.3 Manfaat Masalah	3
BAB II LANDASAN TEORI.....	4
2.1 Pengolahan Citra Digital.....	4
2.2 Deteksi warna RGB	4
2.3 Histogram Citra	4
2.4 Tresholding dinamis	5
2.5 Peningkatan citra backlight menggunakan kontras dan brightness	5
BAB III HASIL	6
BAB IV PENUTUP	16
DAFTAR PUSTAKA	17

BAB I

PENDAHULUAN

1.1 Rumusan Masalah

1. Bagaimana citra digital dapat mendeteksi warna merah, hijau dan biru?
2. Bagaimana cara untuk menampilkan histogram dari hasil deteksi warna untuk menganalisis penyebaran intensitas pixel?
3. Bagaimana cara untuk menentukan nilai ambang batas secara dinamis berdasarkan karakteristik citra?
4. Bagaimana melakukan perbaikan citra pada kondisi backlight sehingga objek utama dapat terlihat lebih jelas tanpa menyebabkan kerusakan warna (color burn)?

1.2 Tujuan Masalah

1. Menerapkan metode deteksi warna untuk menyoroti bagian tertentu dalam citra yang mengandung komponen merah, hijau, dan biru.
2. Menganalisis histogram warna untuk setiap hasil deteksi guna memahami sebaran intensitas warna dalam citra.
3. Mengembangkan metode perhitungan threshold dinamis untuk deteksi warna
4. Meningkatkan kualitas citra backlight dengan teknik peningkatan kecerahan dan kontras, sehingga objek utama menjadi lebih jelas.

1.3 Manfaat Masalah

1. Dapat memahami konsep dasar serta penerapan deteksi warna dalam pengolahan citra digital.
2. Dapat membaca dan menganalisis histogram
3. Dapat menentukan threshold atau batas ambang secara akurat.
4. memperoleh keterampilan dalam menangani permasalahan umum seperti backlight yang sering terjadi dalam fotografi digital.

BAB II

LANDASAN TEORI

2.1 Pengolahan Citra Digital

Citra adalah representasi objek dua dimensi dari dunia visual, menyangkut berbagai macam disiplin ilmu yang mencakup seni, human vision, astronomi, teknik, dan sebagainya. Merupakan suatu kumpulan piksel-piksel atau titik-titik yang berwarna yang berbentuk dua dimensi.

Pengolahan citra digital adalah teknik mengolah citra yang bertujuan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin komputer yang dapat berupa foto maupun gambar bergerak. Pengolahan citra merupakan cabang ilmu dalam Artificial Intelligence yang menggunakan objek citra dalam bentuk digital untuk penyelesaian kasusnya. Metode dalam citra dapat digunakan baik perhitungan matematis pada objek secara piksel ataupun geometris. Masing-masing objek citra memiliki nilai perbedaan yang dapat diperhitungkan secara matematis, sehingga menunjukkan ciri yang berbeda antara objek satu dengan yang lain.

2.2 Deteksi warna RGB

Cahaya merupakan sebagian dari gelombang elektromagnetik yang dapat dilihat mata dengan komponennya yaitu cahaya merah, jingga, hijau, biru, nila dan ungu. Panjang gelombang cahaya berada pada kisaran $0,2 \mu\text{m}$ - $0,5 \mu\text{m}$ yang bersesuaian dengan frekuensi antara 6×10^{15} Hz hingga 20×10^{15} Hz. Warna cahaya berhubungan dengan panjang gelombang atau frekuensi cahaya tersebut. Cahaya tampak yaitu cahaya yang sensitif pada mata kita jatuh pada kisaran 400nm - 750nm.

Warna RGB berasal dari spektrum cahaya, terdiri dari tiga warna dasar yaitu Merah (Red), Hijau (Green), dan Biru (Blue), dikenal sebagai RGB. Campuran setara dari ketiga warna dasar ini menghasilkan warna putih. Gabungan dua warna dasar menghasilkan warna sekunder, seperti biru dengan merah menghasilkan magenta, merah dengan hijau menghasilkan kuning, dan hijau dengan biru menghasilkan cyan. Kombinasi ketiga warna dasar bersama-sama menghasilkan warna putih.

Untuk monitor komputer, nilai rentangnya paling kecil = 0 dan paling besar = 255. Pilihan skala 256 ini didasarkan pada cara mengungkap 8 digit bilangan biner yang digunakan oleh mesin komputer. Dengan cara ini, diperoleh warna campuran sebanyak $255 \times 255 \times 255 = 16.581.375$ jenis warna. Sebuah jenis warna, dapat dibayangkan sebagai sebuah vektor di ruang dimensi 3 yang biasanya dipakai dalam matematika, koordinatnya dinyatakan dalam bentuk tiga bilangan, yaitu komponen-x, komponen-y dan komponen-z. Misalkan sebuah vektor dituliskan sebagai $r = (x, y, z)$. Untuk warna, komponen-komponen tersebut digantikan oleh komponen R(ed), G(reen), B(lue). Jadi, sebuah jenis warna dapat dituliskan sebagai berikut: warna = RGB (30, 75, 255). Putih = RGB (255,255,255), sedangkan untuk hitam= RGB (0,0,0).

2.3 Histogram Citra

Arti kata histogram berasal dari kata Yunani histos dan grama. Pengertian histogram dalam bidang statistika adalah representasi grafis dari posisi frekuensi intensitas piksel dalam bentuk grafik batang yang mewakili suatu kurva data. Dalam bidang pengolahan citra digital, histogram didefinisikan sebagai distribusi probabilitas statistik setiap tingkat keabuan dalam citra digital.

Histogram adalah grafik yang menunjukkan seberapa sering bayangan abu-abu muncul pada suatu gambar. Bentuk histogram suatu gambar dapat memberi tahu anda banyak hal tentang gambar tersebut. Untuk gambar gelap, tingkat keabuan gambar dikelompokkan ke dalam nilai tingkat keabuan. Saat ini, nilai skala abu-abu suatu gambar dipusatkan di sekitar nilai abu-abu yang tinggi pada gambar yang terang. Skala abu-abu tampaknya terdistribusi dengan baik. Kontras gambar ditentukan oleh rentang dinamis, yang didefinisikan sebagai rasio intensitas piksel terang dan gelap. Histogram memberikan informasi tentang pengambilan gambar dan distribusi intensitas. Dengan asumsi bahwa input $f(x,y)$ terdiri dari tingkat keabuan diskrit dalam rentang dinamis $[0, L-1]$.

2.4 Thresholding dinamis

Thresholding adalah salah satu metode segmentasi citra yang memisahkan antara obyek dengan background dalam suatu citra berdasarkan pada perbedaan kecerahannya atau gelap terangnya. Region citra yang cenderung gelap akan dibuat semakin gelap, sedangkan region citra yang cenderung terang akan dibuat semakin terang. Thresholding digunakan untuk men-segmentasi gambar dengan pengaturan semua piksel yang nilai intensitasnya di atas ambang batas nilai menjadi latar depan dan semua piksel yang tersisa menjadi latar belakang. Thresholding juga digunakan untuk membagi gambar menjadi bagian yang lebih mudah diamati. Metode ini mengubah gambar grayscale bergradasi menjadi gambar biner hitam-putih. Ini memisahkan objek dari latar belakang berdasarkan intensitas piksel.

2.5 Peningkatan citra backlight menggunakan kontras dan brightness

Citra backlight adalah kondisi di mana objek utama dalam citra tampak gelap karena berada di depan sumber cahaya yang sangat terang, seperti matahari atau lampu intens. Hal ini menyebabkan objek menjadi kurang terlihat jelas, sementara latar belakang justru terlalu terang. Untuk mengatasi hal ini, dilakukan peningkatan citra dengan mengatur nilai brightness dan kontras.

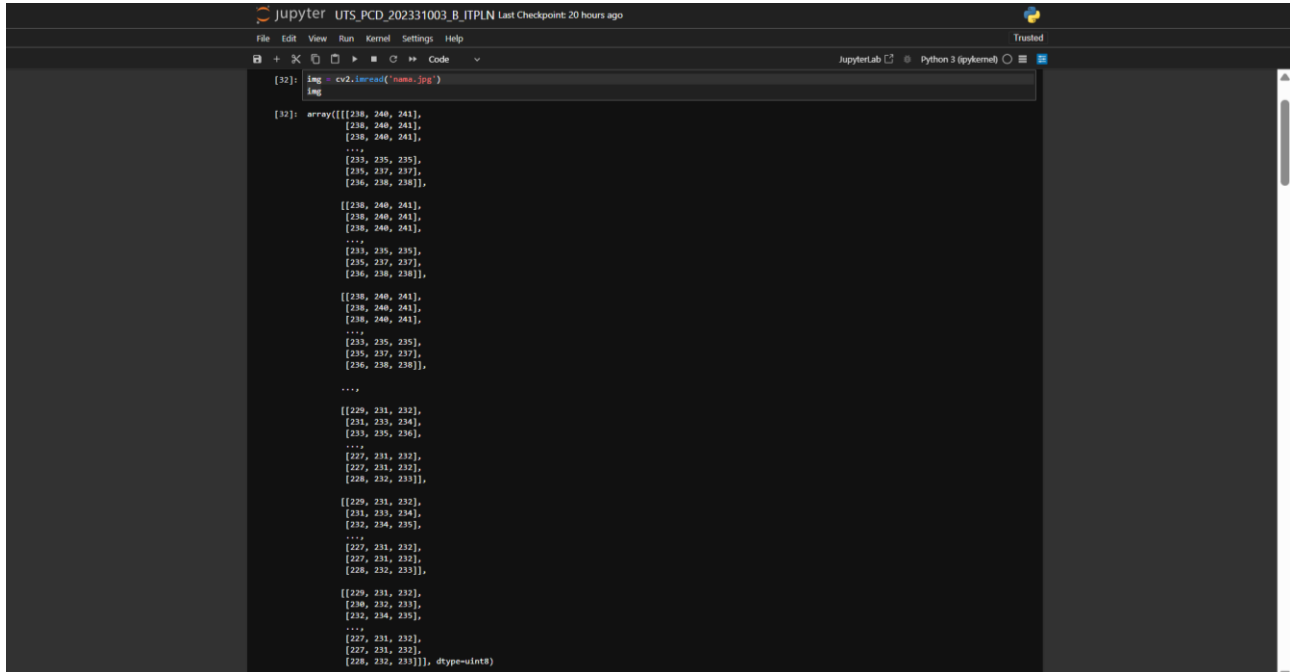
Brightness mengacu pada pencahayaan umum suatu gambar, yaitu seberapa terang atau gelap citra tersebut. Sementara itu, kontras berkaitan dengan perbedaan intensitas antara piksel terang dan gelap. Peningkatan brightness dilakukan dengan menambahkan nilai tetap pada intensitas piksel, sedangkan kontras ditingkatkan dengan mengalikan intensitas piksel dengan suatu faktor skala. Teknik ini memungkinkan tampilan objek yang sebelumnya tertutup bayangan (underexposed) menjadi lebih jelas tanpa harus mengubah keseluruhan komposisi warna gambar. Peningkatan kontras dan brightness sering dikombinasikan untuk mencapai hasil yang lebih optimal. Misalnya, pada citra backlight, menaikkan brightness saja mungkin tidak cukup karena latar belakang yang sudah terang akan menjadi overexposed. Namun dengan mengatur kontras bersamaan, perbedaan antara objek dan latar menjadi lebih tajam, sehingga objek utama menjadi fokus perhatian.

Dampak positif dari pengaturan brightness dan kontras adalah peningkatan visibilitas dan keterbacaan citra, terutama pada area penting seperti wajah atau tulisan. Namun jika tidak dikontrol dengan baik, efek samping seperti color burn atau hilangnya detail bisa terjadi. Oleh karena itu, proses ini memerlukan penyesuaian yang hati-hati agar hasil akhir tetap informatif dan nyaman dilihat. Teknik ini termasuk dalam kategori image enhancement dan sangat penting dalam banyak aplikasi pengolahan citra seperti pengenalan wajah, koreksi pencahayaan otomatis, serta dokumentasi digital di lingkungan pencahayaan tak ideal.

BAB III

HASIL

1. Setelah perintah untuk membaca gambar, terdapat kode `img`, ini berfungsi untuk menghasilkan array yang didapat dari gambar tersebut. array ini merupakan pixel yang terdapat pada gambar.



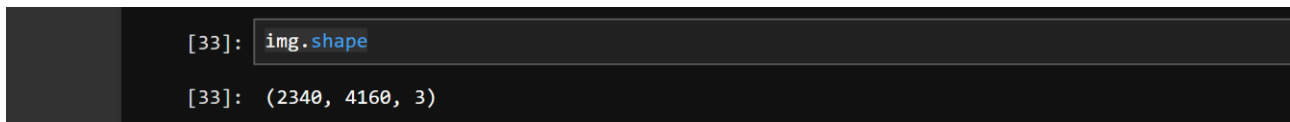
```

[32]: img = cv2.imread('nama.jpg')
      img

[32]: array([[238, 240, 241],
            [238, 240, 241],
            [238, 240, 241],
            ...,
            [235, 235, 235],
            [235, 237, 237],
            [236, 238, 238]],
          ...,
          [[238, 240, 241],
            [238, 240, 241],
            [238, 240, 241],
            ...,
            [235, 235, 235],
            [235, 237, 237],
            [236, 238, 238]],
          ...,
          [[238, 240, 241],
            [238, 240, 241],
            [238, 240, 241],
            ...,
            [235, 235, 235],
            [235, 237, 237],
            [236, 238, 238]],
          ...,
          [[229, 231, 232],
            [231, 233, 234],
            [233, 235, 236],
            ...,
            [227, 231, 232],
            [227, 231, 232],
            [228, 232, 233]],
          ...,
          [[229, 231, 232],
            [231, 233, 234],
            [233, 234, 235],
            ...,
            [227, 231, 232],
            [227, 231, 232],
            [228, 232, 233]],
          ...,
          [[229, 231, 232],
            [230, 232, 233],
            [232, 234, 235],
            ...,
            [227, 231, 232],
            [227, 231, 232],
            [228, 232, 233]]], dtype=uint8)

```

2. Kode `img.shape` bertujuan untuk mengetahui dimensi gambar, yang berupa ukuran panjang, lebar, dan juga total jumlah warna, yaitu 3



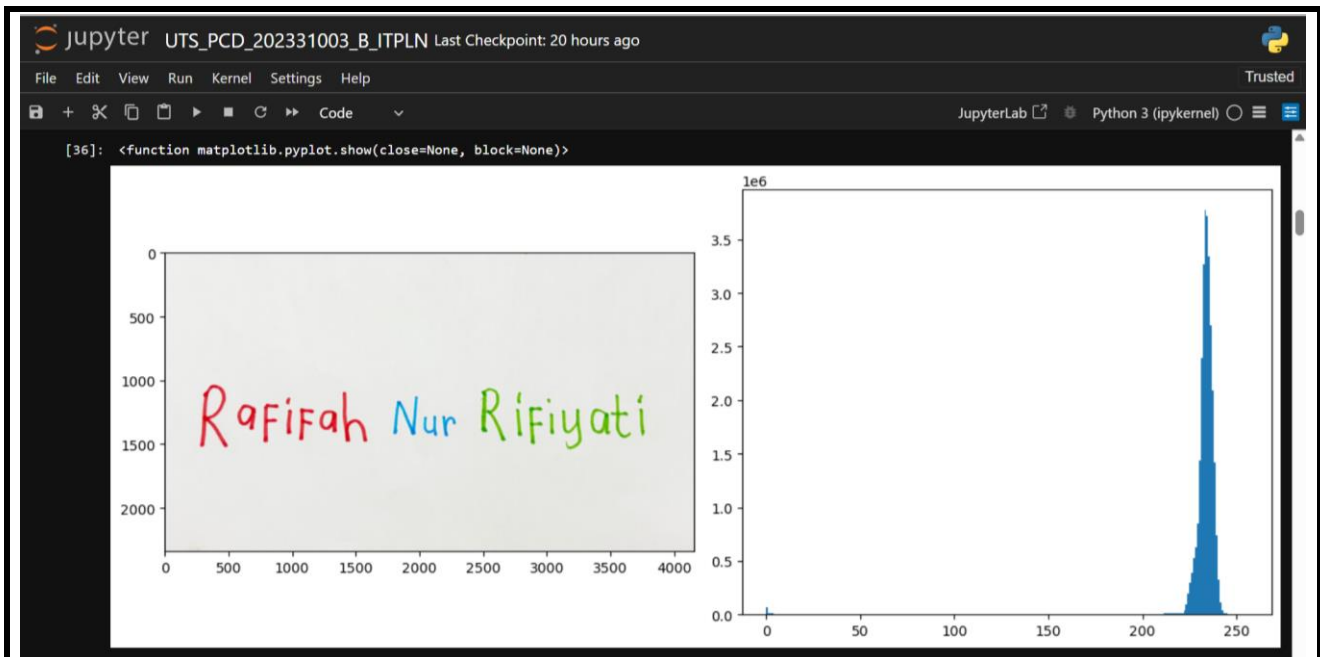
```

[33]: img.shape

[33]: (2340, 4160, 3)

```

3. Menampilkan gambar asli dan juga histogram dari gambar tersebut, dapat dilihat dari hasil output dibawah ini. untuk menampilkan gambar asli dan juga histogramnya, dimulai dari membuat kanvas berukuran 1 x 2, kemudian menempatkannya pada axis (axs) yang sesuai atau yang kita inginkan, untuk gambar asli, saya menempatkan di axs 0 dan untuk histogram saya menempatkannya di axs 1. Untuk menampilkan gambar asli kita menggunakan `imshow(img)`, dan untuk menampilkan gambar histogram kita menggunakan `hist(img.ravel(),256,[0,256])`, angka 256 yang pertama yaitu angka interval atau bin histogram, selanjutnya yaitu angka rentang pixel yang sedang kita analisis yakni 0-255. Dapat kita lihat bahwa grafik dominan disebelah kanan, ini dapat berarti bahwa persebaran rentang pixel dominan berwarna cerah karena mendekati angka 250 - 256 (putih).



4. Mendeteksi warna merah, hijau dan juga biru menggunakan masking untuk mendeteksi dan menyembunyikan warna tertentu.

- Pada citra kontras, menggunakan kode `img_contrast = cv2.normalize(img, None, 0, 255, cv2.NORM_MINMAX)`, untuk menyesuaikan kontras gambar agar nilai pixel tersebar dari 0 hingga 255. Sehingga, gambar yang awalnya gelap atau terang akan di sesuaikan ke seluruh rentang intensitas.
- Untuk deteksi warna hijau, menggunakan logika masking, yakni `if target == 'green': mask = (g > r + 30) & (g > b + 30)`, yang berarti pixel dianggap hijau jika nilai hijau (g) > merah (r) dan > biru (b) dengan selisih minimal 30.
- Untuk deteksi warna biru, menggunakan logika masking, yakni `elif target == 'blue': mask = (b > r + 30) & (b > g + 30)`, yang berarti sebuah piksel dianggap biru jika: nilai biru (B) > merah (R) + 30, nilai biru (B) > hijau (G) + 30 .
- Untuk deteksi warna merah, menggunakan logika masking, yakni `elif target == 'red': mask = (r > g + 30) & (r > b + 30)`, yang berarti sebuah piksel dianggap merah jika: nilai merah (R) > hijau (G) + 30, nilai merah (R) > biru (B) + 30
- `else: mask = np.zeros_like(r, dtype=bool)`, yang berarti jika parameter target tidak sama dengan 'green', 'blue', atau 'red', maka kode ini akan dieksekusi.

Kemudian, untuk non target, kita ubah ke grayscale dengan kode

- `gray = np.mean(img, axis=2).astype(np.uint8)` for `c in range(3): hasil[:, :, c][~mask] = gray[~mask]` : kode ini digunakan untuk megubah warna tulisan yang bukan target deteksi dengan grayscale.

Selanjutnya untuk warna yang terdeteksi, akan dihighlight, dengan kode

- `hasil[mask] = [240, 240, 240]` `return hasil.astype(np.uint8)` : kode ini merupakan operasi untuk masking, apabila ia true maka kode ini akan berjalan, sehingga warna yang terdeteksi akan diubah ke warna [240, 240, 240]. Mengembalikan `hasil.astype` menggunakan `np.uint8` yaitu konversi tipe data kembali ke `uint8`.

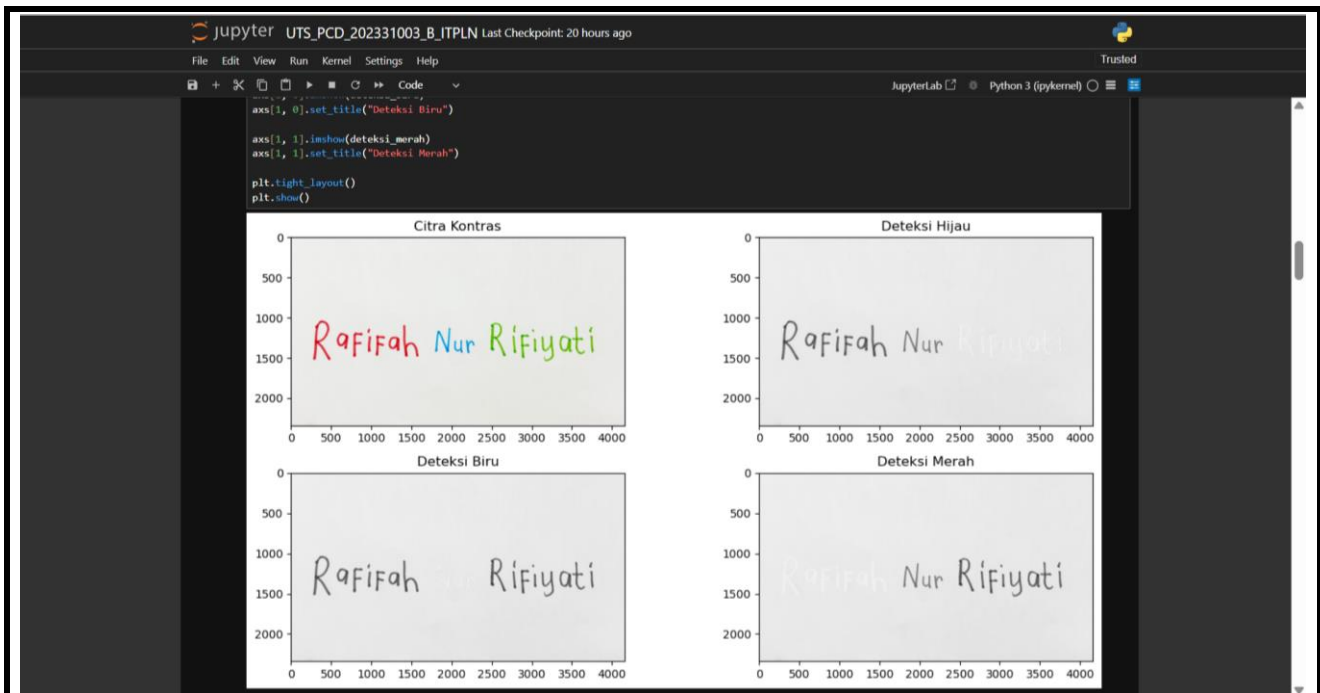
Setelah melakukan logika deteksi warna, maka kita akan memanggil fungsi untuk setiap deteksi supaya dapat kita gunakan untuk menampilkan hasilnya. Dengan kode:

- `deteksi_hijau = masking(img, 'green')`
- `deteksi_biru = masking(img, 'blue')`
- `deteksi_merah = masking(img, 'red')`

selanjutnya yaitu fungsi untuk menampilkan hasil dari deteksi warna tersebut, menggunakan kode :

- `fig, axs = plt.subplots(2, 2, figsize=(12, 6))` : untuk mengatur ukuran kanvas yaitu 2x2
- `axs[0, 0].imshow(img_contrast)` : menampilkan hasil gambar kontras
`axs[0, 0].set_title("Citra Kontras")` : membuat judul pada gambar
- `axs[0, 1].imshow(deteksi_hijau)` : menampilkan hasil deteksi hijau
`axs[0, 1].set_title("Deteksi Hijau")` : membuat judul pada gambar
- `axs[1, 0].imshow(deteksi_biru)` : menampilkan hasil deteksi biru
`axs[1, 0].set_title("Deteksi Biru")` : membuat judul pada gambar
- `axs[1, 1].imshow(deteksi_merah)` : menampilkan hasil deteksi merah
`axs[1, 1].set_title("Deteksi Merah")` : membuat judul pada gambar
- `plt.tight_layout()` : untuk memperluas kanvas supaya seukuran dengan ukuran terminal
`plt.show()` : supaya dapat menampilkan gambar

Analisis yang terjadi pada citra yaitu setelah diberi masking, apabila pada tulisan tersebut terdapat warna yang terdeteksi, maka warna pada tulisan tersebut akan memudar menjadi abu abu terang (240), sedangkan untuk warna yang tidak terdeteksi diubah menjadi warna abu gelap, sehingga membuat objek tulisan lebih menonjol dibandingkan warna yang dideteksi.



5. menampilkan histogram dari hasil deteksi warna, untuk menampilkan histogram ini, kita menggunakan kode `hist, ravel`. Kode `ravel`, berfungsi untuk mengubah array 2D menjadi 1D. kemudian, 256 merupakan jumlah kotak histogramnya, `[0, 256]` merupakan jumlah nilai rentang pixel.

Selain mengubah menjadi histogram, kode ini juga berisi tentang tata letak histogram pada kanvas. Setiap hasil deteksi warna dan histogramnya diletakkan pada satu kanvas berukuran 1 x 2 dengan kode: `fig, ax = plt.subplots(1, 2, figsize=(12, 4))` `ax` yang berarti axis (sumbu). Untuk hasil deteksi, sama seperti kode dan penjelasan sebelumnya.

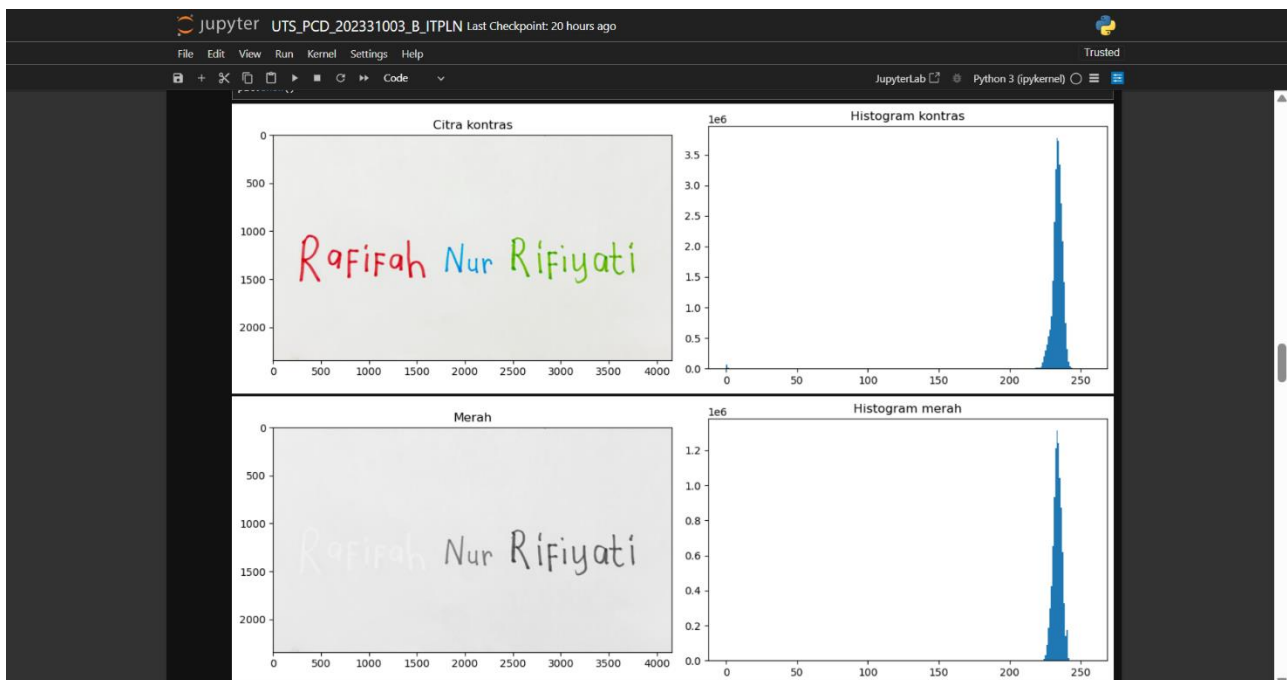
- Untuk histogram kontras :
`ax[1].hist(img_contrast.ravel(), 256, [0, 256])`
- Untuk histogram merah :
`ax[1].hist(gray_r.ravel(), 256, [0, 256])`
- Untuk histogram hijau :
`ax[1].hist(gray_g.ravel(), 256, [0, 256])`
- Untuk histogram biru:
`ax[1].hist(gray_b.ravel(), 256, [0, 256])`

Sebelum itu, kita harus membuat variabel baru untuk menyimpan hasil deteksi setiap warna, yang telah dikonversi ke grayscale:

- Untuk deteksi warna merah dan histogram merah :
`gray_r = cv2.cvtColor(deteksi_merah, cv2.COLOR_RGB2GRAY)`
- Untuk deteksi warna hijau dan histogram hijau :
`gray_g = cv2.cvtColor(deteksi_hijau, cv2.COLOR_RGB2GRAY)`
- Untuk deteksi warna biru dan histogram biru :
`gray_b = cv2.cvtColor(deteksi_biru, cv2.COLOR_RGB2GRAY)`

- Untuk deteksi kontras dan histogram kontras : `img_contrast = cv2.normalize(img, None, 0, 255, cv2.NORM_MINMAX)`

Analisis histogram deteksi warna : Histogram dari ketiga hasil deteksi memperlihatkan kecenderungan puncak intensitas pada nilai tinggi (240) yang berasal dari piksel warna yang berhasil terdeteksi. Sebaliknya, area non-target menghasilkan persebaran nilai intensitas lebih variatif karena dikonversi ke grayscale. Hal ini menunjukkan bahwa metode masking berhasil memisahkan bagian gambar berdasarkan warna, dan histogram mendukung analisis visual terhadap intensitas distribusi warna pada setiap mode deteksi.



6. Mencari ambang batas terkecil sampai dengan terbesar.

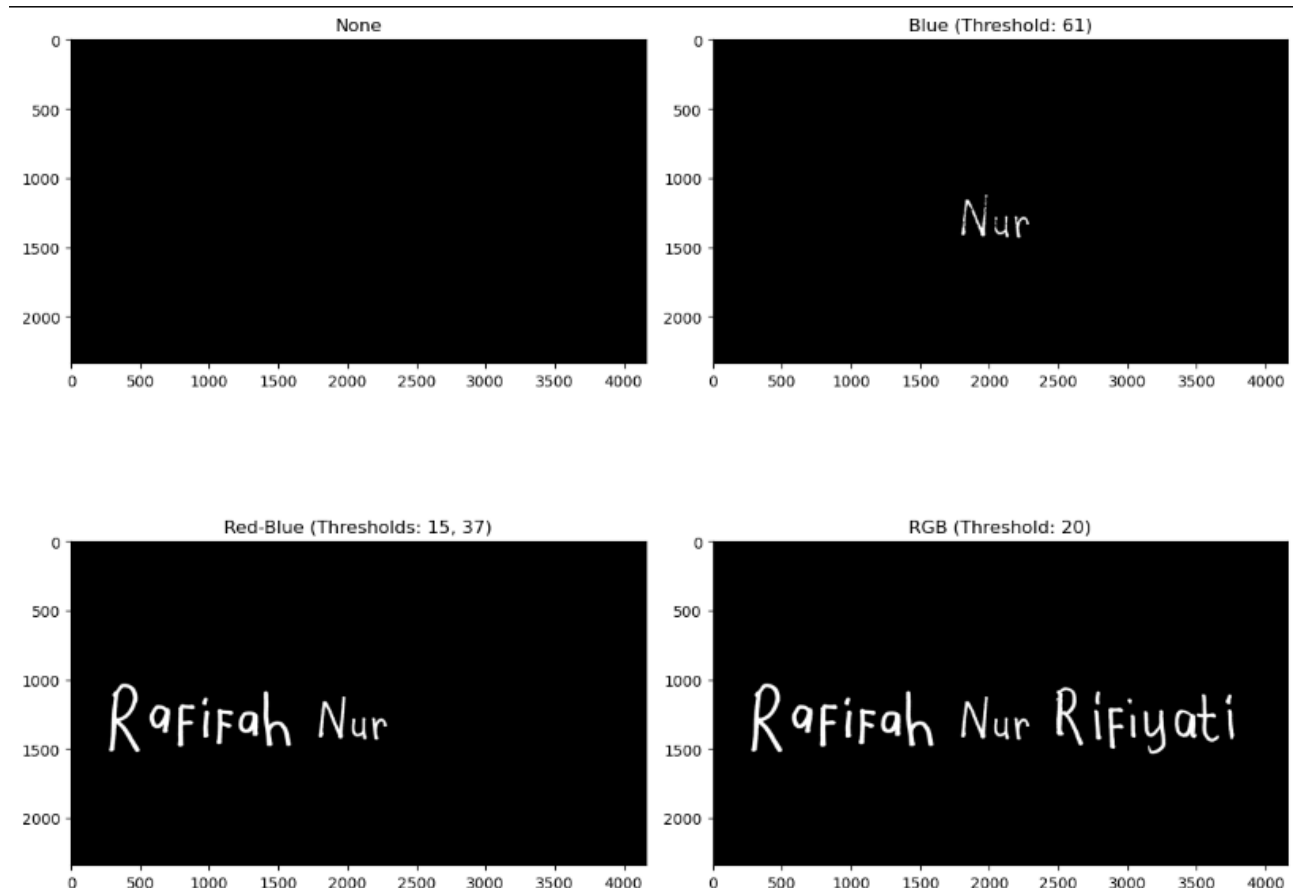
Kode ini melakukan proses deteksi tulisan berdasarkan warna dominan (biru, merah, hijau) menggunakan threshold dinamis yang dihitung dari gambar. Untuk mencari ambang batas, caranya yaitu dengan menghitung nilai rata-rata (mean) dan simpangan baku (standard deviation) untuk setiap channel warna: R, G, dan B. pada kode tersebut, ia menentukan ambang batas (threshold) untuk mendeteksi bagian gambar yang mencolok warnanya.

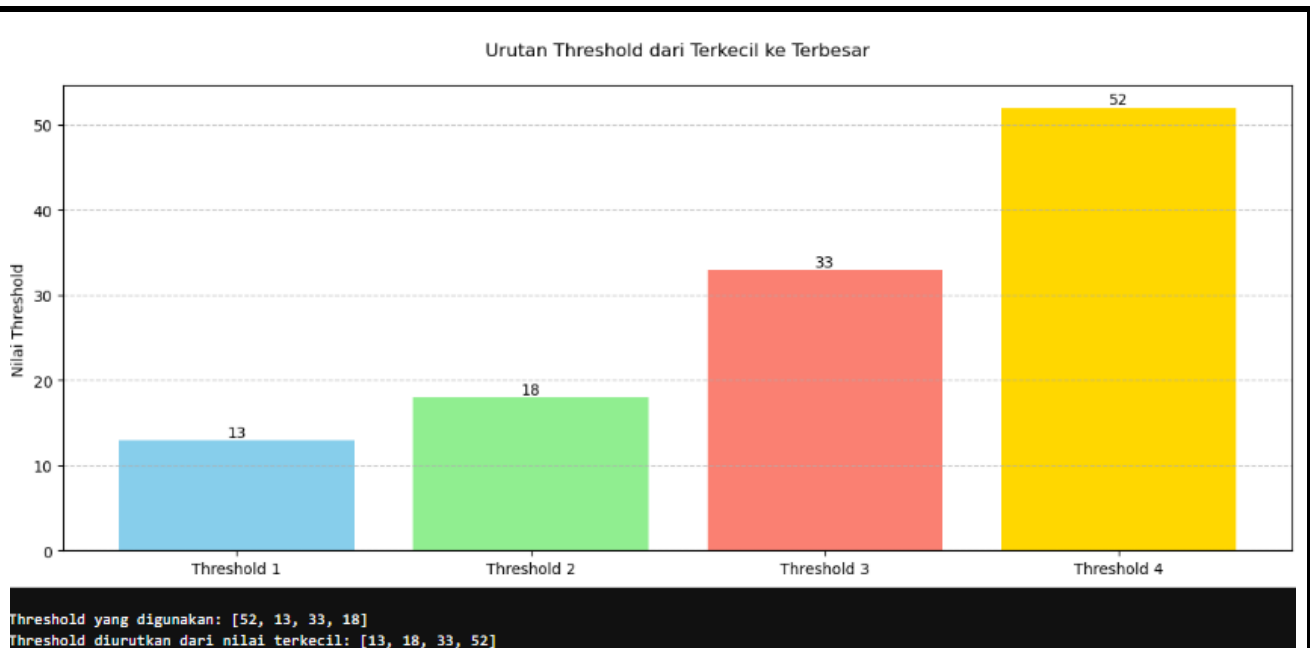
- Citra diolah dalam beberapa mode:
 1. none: citra kosong (semua hitam, tidak ada deteksi)
 2. blue: mendeteksi jika biru jauh lebih tinggi dari R dan G
 3. red_blue: mendeteksi kombinasi merah dan biru yang saling dominan
 4. rgb: mendeteksi jika salah satu channel RGB dominan dibanding dua lainnya.
- Penjelasan Perhitungan threshold :
 1. $\text{blue_thresh/ biru} : (\max(\text{mean_b} - \text{mean_r}, \text{mean_b} - \text{mean_g}) + 2 * \text{std_b})$: Nilai batas seberapa jauh biru harus lebih tinggi dari R & G untuk dianggap dominan biru.

2. red_blue_low/ merah biru rendah : $\min(\text{std_r}, \text{std_b}) * 0.7$: Ambang batas rendah untuk selisih deteksi kombinasi merah-biru, agar deteksi tidak terlalu ketat.
3. red_blue_high/ merah biru tinggi: $\max(\text{std_r}, \text{std_b}) * 1.2$: Ambang batas tinggi untuk deteksi lebih yakin terhadap dominasi warna merah/biru.
4. Rgb/ merah, hijau biru : $((\text{std_r} + \text{std_g} + \text{std_b})/3) * 0.8$: threshold menengah

Untuk mendeteksi tulisan biru: nilai biru harus lebih tinggi 74 dari merah dan hijau,
 Untuk mendeteksi tulisan merah atau biru: selisih R-G atau B-G harus di atas 66, tapi tidak lebih kecil dari 22, Untuk dominasi RGB secara umum: salah satu channel harus lebih tinggi dari dua lainnya dengan selisih minimal 41.

Gambar menjadi warna hitam putih karena : putih [255, 255, 255] diberikan ke piksel yang memenuhi kondisi threshold (terdeteksi tulisan), kemudian warna hitam [0, 0, 0] diberikan ke yang tidak memenuhi (latar belakang atau warna netral).





7. Memperbaiki gambar backlight

1. Gambar Asli (RGB)

Format dari gambar ini adalah RGB (Red, Green, Blue), yang menyimpan tiga channel warna. Gambar ini belum diubah sama sekali, sehingga masih menampilkan semua warna dan cahaya sebagaimana aslinya.

2. Gambar Grayscale

Proses: citra RGB diubah ke grayscale menggunakan `cv2.cvtColor`.

Saat dikonversi ke grayscale, warna dihilangkan dan tiap piksel direpresentasikan oleh satu nilai intensitas (0–255). Nilai ini didapat dari gabungan nilai R, G, dan B. maka hasilnya gambar menjadi hitam-putih (abu-abu) dan hanya memperlihatkan terang-gelap saja, tanpa warna.

3. Gambar Grayscale yang Dipercerah

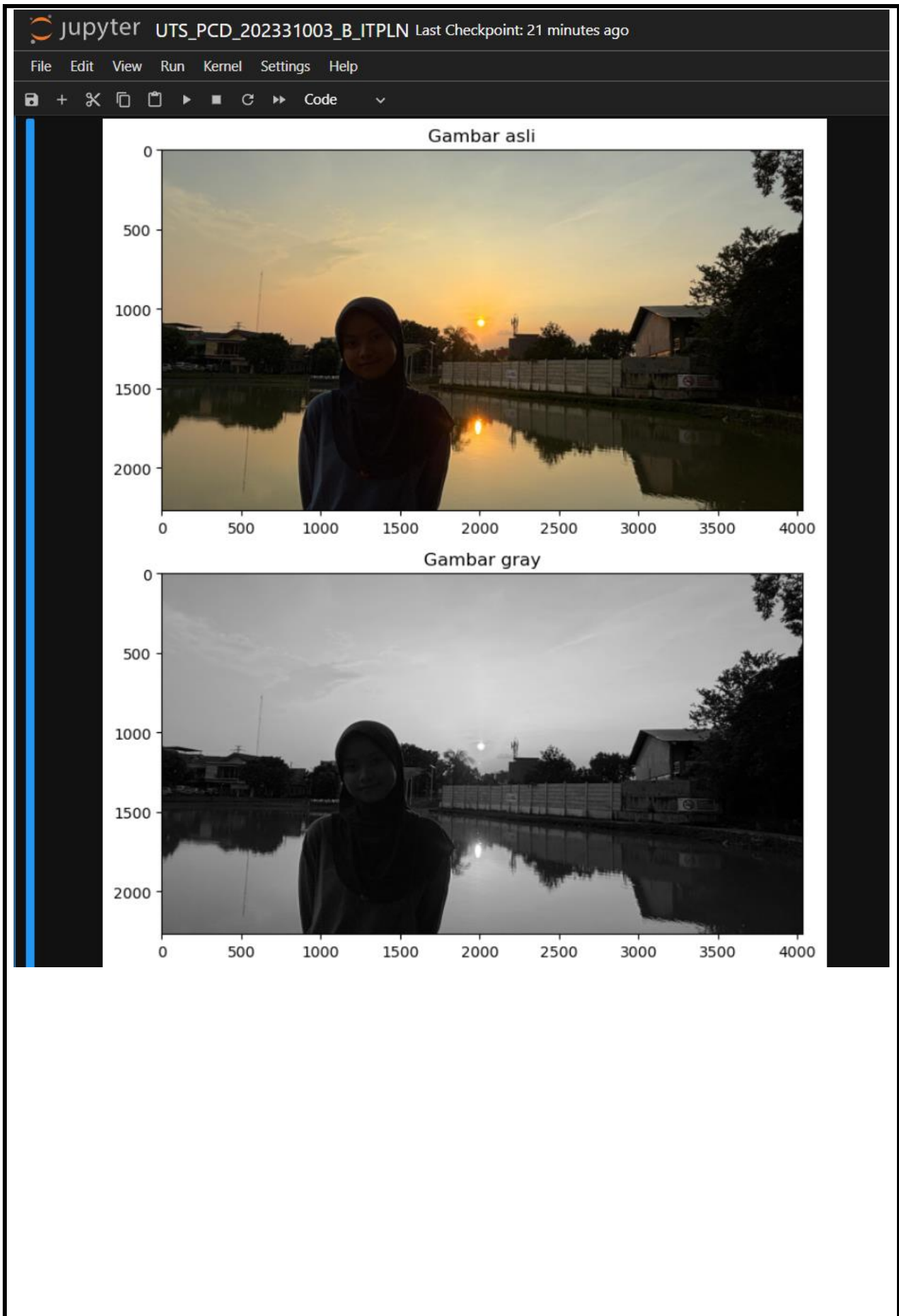
Proses: nilai brightness ditambahkan menggunakan `cv2.convertScaleAbs(gray, alpha=1.0, beta=80)`. Dalam proses ini, tiap piksel grayscale ditambah nilainya sebesar 80 ($\beta=80$). Sehingga, semua bagian gambar menjadi lebih terang. Area gelap yang sebelumnya samar jadi mulai terlihat.

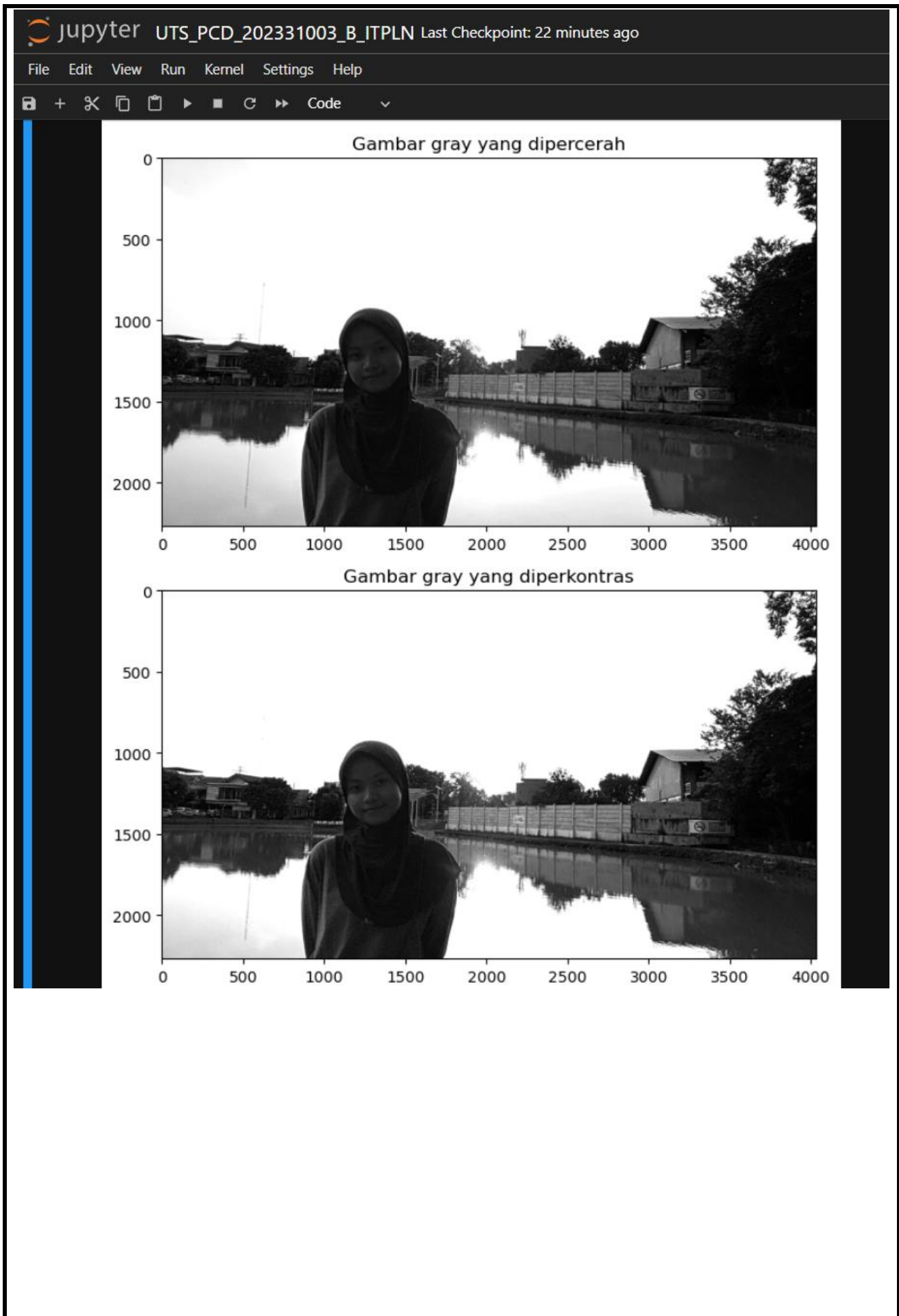
4. Gambar Grayscale yang Diperkontras

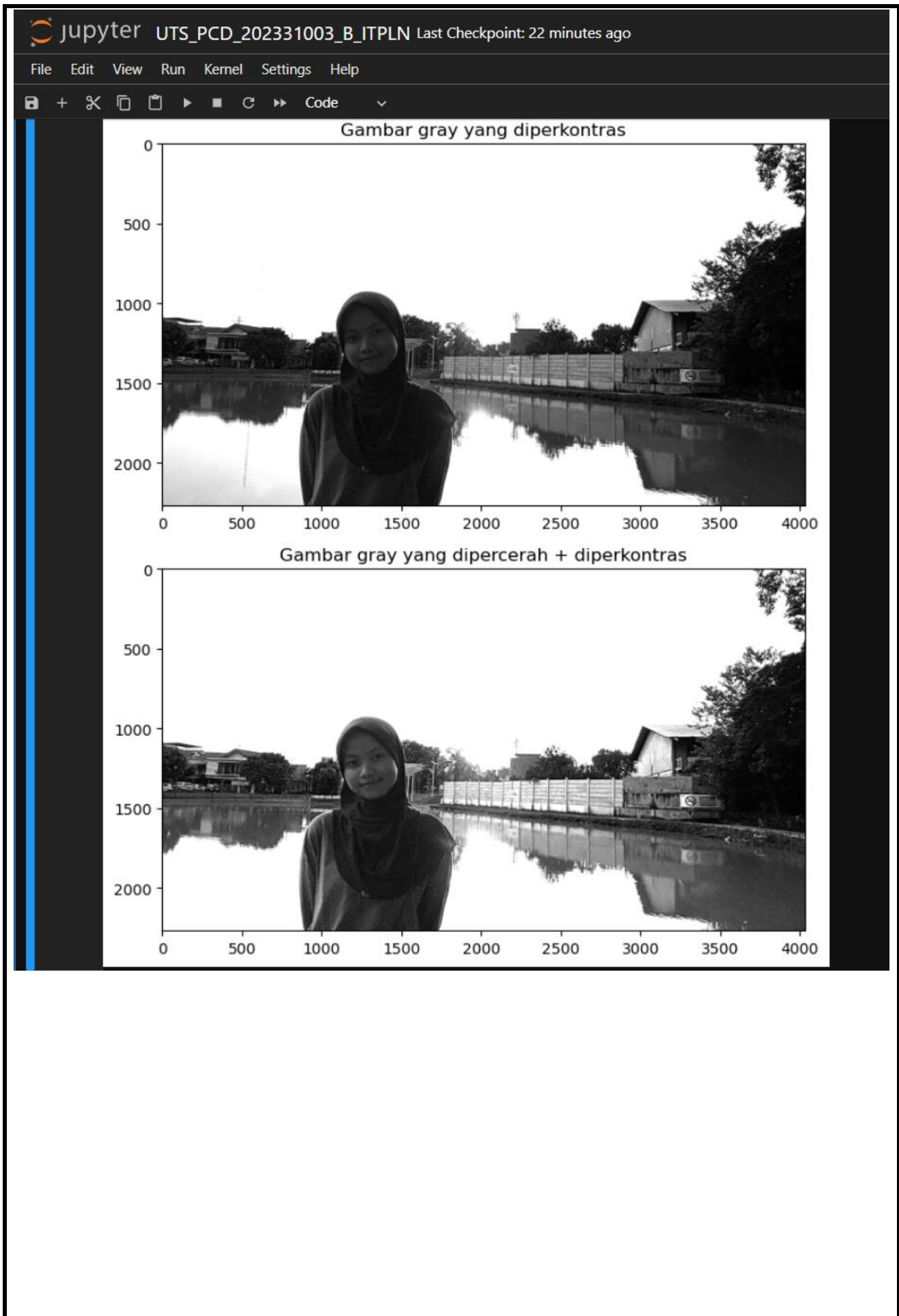
Proses: nilai kontras diperbesar dengan $\alpha=2.0$ dan $\beta=0$. Artinya, semua nilai piksel dikalikan 2x. sehingga, perbedaan antara piksel gelap dan terang menjadi semakin jelas. Area terang menjadi lebih terang, dan area gelap menjadi lebih gelap. Gambar tampak lebih tajam.

5. Gambar Grayscale yang Dipercerah + Diperkontras

Proses: kombinasi $\alpha=2.0$ dan $\beta=80$ (diperbesar dan diterangkan). Kode ini merupakan gabungan dari dua proses sebelumnya. Piksel dikalikan terlebih dahulu supaya kontrasnya naik, lalu ditambah 80 supaya lebih terang. Sehingga gambar menjadi lebih jelas, lebih terang, dan lebih tajam.







BAB IV

PENUTUP

Dari teori dan praktek yang sudah dilakukan, dapat diketahui bahwa pengolahan citra digital memiliki peran penting dalam meningkatkan kualitas visual suatu gambar serta membantu menonjolkan informasi penting yang mungkin tidak langsung terlihat. Melalui proses konversi ke grayscale, peningkatan brightness dan kontras, serta penerapan thresholding dinamis, citra yang awalnya kurang jelas seperti gambar backlight dapat diperbaiki sehingga objek utama menjadi lebih menonjol dibandingkan latar belakang.

Selain itu, dengan memanfaatkan logika dari nilai RGB, proses deteksi warna merah, hijau, dan biru dapat dilakukan secara otomatis menggunakan masking. Warna yang berhasil terdeteksi dapat disorot dengan lebih jelas, sedangkan warna lainnya dibuat netral agar fokus tetap terjaga pada objek yang diinginkan. Pada praktikum ini, histogram digunakan untuk menganalisis sebaran intensitas piksel pada gambar. Visualisasi histogram memberi gambaran apakah gambar terlalu gelap, terlalu terang, atau sudah seimbang.

Pada penerapan thresholding, melalui koding dan rumus, mampu menentukan nilai ambang batas secara otomatis menyesuaikan karakteristik setiap gambar. Dengan threshold ini, proses segmentasi atau deteksi tulisan bisa dilakukan dengan lebih akurat dan adaptif, tanpa harus mengatur nilai ambang secara manual. Hal ini sangat membantu ketika gambar memiliki pencahayaan yang tidak merata atau warna latar yang kompleks.

Melalui peningkatan brightness dan kontras, terutama pada citra dengan kondisi backlight, objek utama yang semula tampak gelap dapat dibuat lebih jelas dan menonjol. Karena menggunakan nilai alpha dan beta yang tepat, sehingga tidak terjadi color burn.

DAFTAR PUSTAKA

- [1]Anggraeni, D. T. (2021). Perbaikan Citra Dokumen Hasil Pindai Menggunakan Metode Simple, Adaptive-Gaussian, dan Otsu Binarization Thresholding. *Expert*, 11(2), 71-77.
<https://www.neliti.com/publications/391993/perbaikan-citra-dokumen-hasil-pindai-menggunakan-metode-simple-adaptive-gaussian>
- [2]BAITI A. (2023). CITRA RGB (RED, GREEN & BLUE), BINER, DAN GRAYSCALE DALAM IMAGE PROCESSING HILAL UNTUK PENENTUAN AWAL BULAN KAMARIAH (Thesis, UIN Walisongo Semarang).
https://eprints.walisongo.ac.id/23529/1/2002048023_ARSYITA%20BAITI%20MUSFIROH_Full%20Tesis%20-%20Arsyita%20Baiti%20Musfiroh.pdf
- [3]Ikhsan, M., Supiyandi, S., & Hakiki, A. W. (2024). ANALISIS PERBANDINGAN METODE HISTOGRAM EQUALIZATION DAN GAUSSIAN FILTER UNTUK PERBAIKAN KUALITAS CITRA. *JOURNAL OF SCIENCE AND SOCIAL RESEARCH*, 7(2),487-492.
<https://www.jurnal.goretanpena.com/index.php/JSSR/article/view/1865>
- [4]Jumadi, J., Yupianti, Y., & Sartika, D. (2021). PENGOLAHAN CITRA DIGITAL UNTUK IDENTIFIKASI OBJEK MENGGUNAKAN METODE HIERARCHICAL AGGLOMERATIVE CLUSTERING. *JST (Jurnal Sains Dan Teknologi)*, 10(2), 148–156. <https://doi.org/10.23887/jstundiksha.v10i2.33636>
- [5]Kusnadi, K., Riana, D., & Syahrani, M. (2022). PENGELOLAAN CITRA DIGITAL DENGAN MENGGUNAKAN METODE TRANSFORMASI GRAYSCALE DAN PEMERATAAN HISTOGRAM. *JTIK (Jurnal Teknik Informatika Kaputama)*, 6(1), 108–119. <https://doi.org/10.59697/jtik.v6i1.369>
- [6]Maulana, A., Auliatunnajah, F., Rosidin, N., Darmawan, M. R. R., & Rosyani, P. (2024). Implementasi OpenCV dengan Metode Image Thresholding pada Gambar. *Jurnal Artificial Intelligent dan Sistem Penunjang Keputusan*, 2(1).
<https://jurnalmahasiswa.com/index.php/aidanspk/article/download/1168/990/3059>
- [7]OPTIMALISASI FILE RAW PADA GAMBAR BACKLIGHT DENGAN MENERAPKAN TEKNIK EDITING HIGH DYNAMIC RANGE (HDR). (2024). *Jurnal Informatika Teknologi Dan Sains (Jinteks)*, 6(1), 21-25.
<https://doi.org/10.51401/jinteks.v6i1.3162>
- [8]RAHMAT PIDI, P. U. T. R. A. (2024). *PENGOLAHAN CITRA DIGITAL UNTUK MENGIDENTIFIKASI TINGKAT KEMATANGAN BUAH KELAPA SAWIT BERDASARKAN WARNA RGB DAN HSV DENGAN MENGGUNAKAN METODE SELF ORGANIZING MAP (SOM)* (Doctoral dissertation, Universitas Dehasen Bengkulu). <http://repository.unived.ac.id/2015/>