

TUGAS INDIVIDU
PENGOLAHAN CITRA DIGITAL



Dosen Pengampu :

Achmad Junaidi, S.Kom, M.Kom

Nama :

Ahmad Sofian Aris S (21081010211)

Kelas Pengolahan Citra Digital – C081
UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN” JAWA TIMUR
SURABAYA 2024

SOAL

Buatlah program untuk menampilkan hasil pengolahan gambar yang telah diinputkan (RGB) menjadi :

1. Pisahkan menjadi 3 buah gambar dari masing-masing layer



Gambar diatas merupakan gambar yang akan saya gunakan untuk mengerjakan tugas ini. Berikut code dan hasil dari pemisahan gambar diatas :

Layer Merah



Layer Hijau



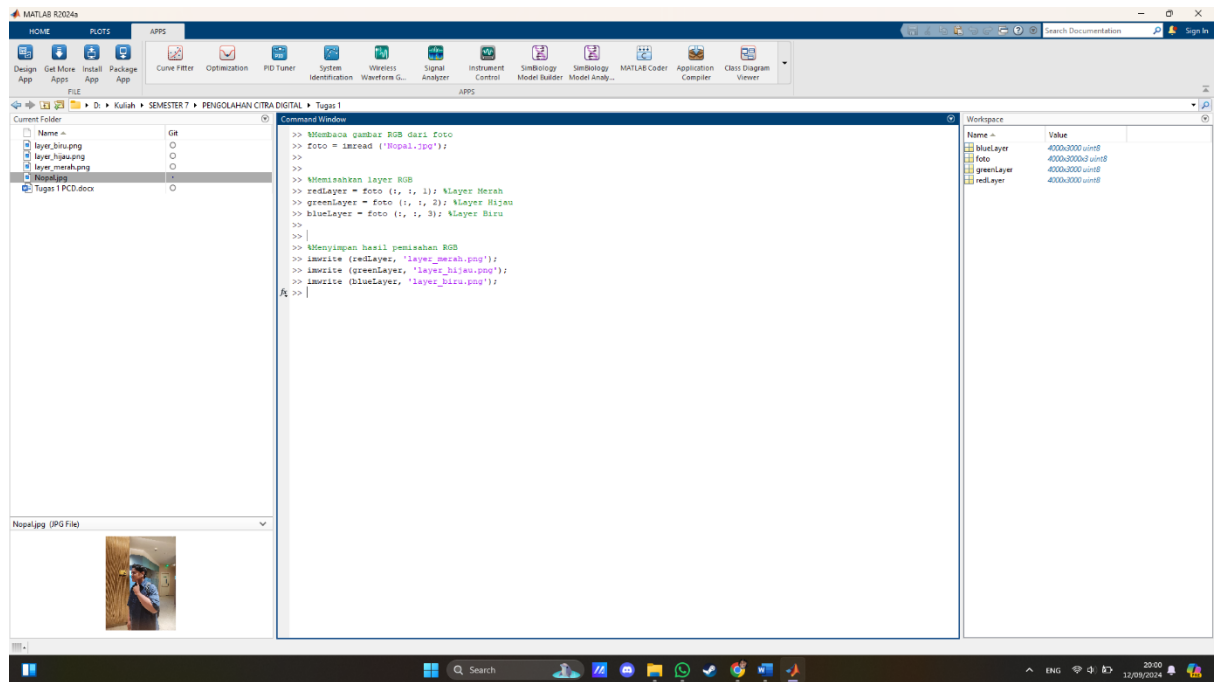
Layer Biru



Setelah gambar RGB dibaca, kita bisa memisahkan gambar tersebut menjadi tiga layer terpisah (merah, hijau, dan biru). Gambar RGB biasanya direpresentasikan sebagai matriks 3D dengan ukuran $m \times n \times 3$, di mana m adalah jumlah baris, n adalah jumlah kolom, dan angka 3 menunjukkan tiga layer (R, G, dan B).

Gambar dapat dipisahkan menjadi 3 layer dengan syarat gambar harus dalam format RGB (3 kanal). Jika Anda mencoba ini pada gambar grayscale atau CMYK, hasilnya mungkin berbeda. Layer merah, hijau, dan biru yang dihasilkan hanya akan berisi intensitas masing-masing warna tanpa warna asli.

```
>> %Membaca gambar RGB dari foto
>> foto = imread ('Nopal.jpg');
>>
>>
>> %Memisahkan layer RGB
>> redLayer = foto (:, :, 1); %Layer Merah
>> greenLayer = foto (:, :, 2); %Layer Hijau
>> blueLayer = foto (:, :, 3); %Layer Biru
>>
>>
>> %Menyimpan hasil pemisahan RGB
>> imwrite (redLayer, 'layer_merah.png');
>> imwrite (greenLayer, 'layer_hijau.png');
>> imwrite (blueLayer, 'layer_biru.png');
```



2. Grayscale

a. Manual

Berikut merupakan code dan hasil dari menghitung Grayscale secara manual



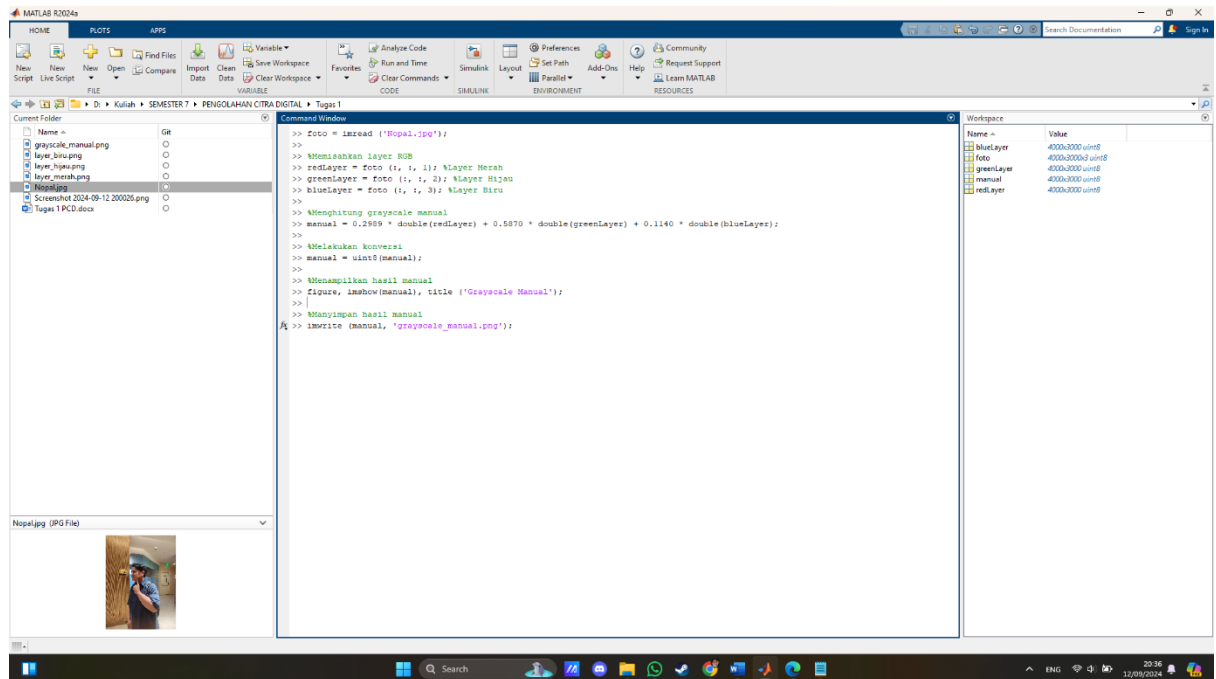
Untuk mengonversi gambar RGB menjadi grayscale secara manual bisa menggunakan rumus luminance standar, yaitu:

$$\text{Grayscale} = 0.2989 \times R + 0.5870 \times G + 0.1140 \times B$$

Ini adalah metode yang memperhitungkan persepsi manusia terhadap warna, di mana hijau lebih dominan dibandingkan merah dan biru. Bilangan diatas merupakan koefisien luminance yang ditentukan oleh standar ITU-R BT.601. Koefisien ini menunjukkan seberapa besar kontribusi masing-masing warna (R, G, B) terhadap persepsi manusia terhadap kecerahan (luminance).

- Merah (R): Kontribusinya adalah 0.2989. Walaupun warna merah mungkin terlihat lebih mencolok, kontribusinya terhadap kecerahan tidak sebesar hijau.
- Hijau (G): Kontribusinya paling besar yaitu 0.5870, karena mata manusia lebih sensitif terhadap warna hijau.
- Biru (B): Kontribusinya paling kecil yaitu 0.1140, karena mata manusia kurang sensitif terhadap warna biru.

```
>> foto = imread ('Nopal.jpg');
>>
>> %Memisahkan layer RGB
>> redLayer = foto (:, :, 1); %Layer Merah
>> greenLayer = foto (:, :, 2); %Layer Hijau
>> blueLayer = foto (:, :, 3); %Layer Biru
>>
>> %Menghitung grayscale manual
>> manual = 0.2989 * double(redLayer) + 0.5870 *
double(greenLayer) + 0.1140 * double(blueLayer);
>>
>> %Melakukan konversi
>> manual = uint8(manual);
>>
>> %Menampilkan hasil manual
>> figure, imshow(manual), title ('Grayscale
Manual');
>>
>> %Menyimpan hasil manual
>> imwrite (manual, 'grayscale_manual.png');
```



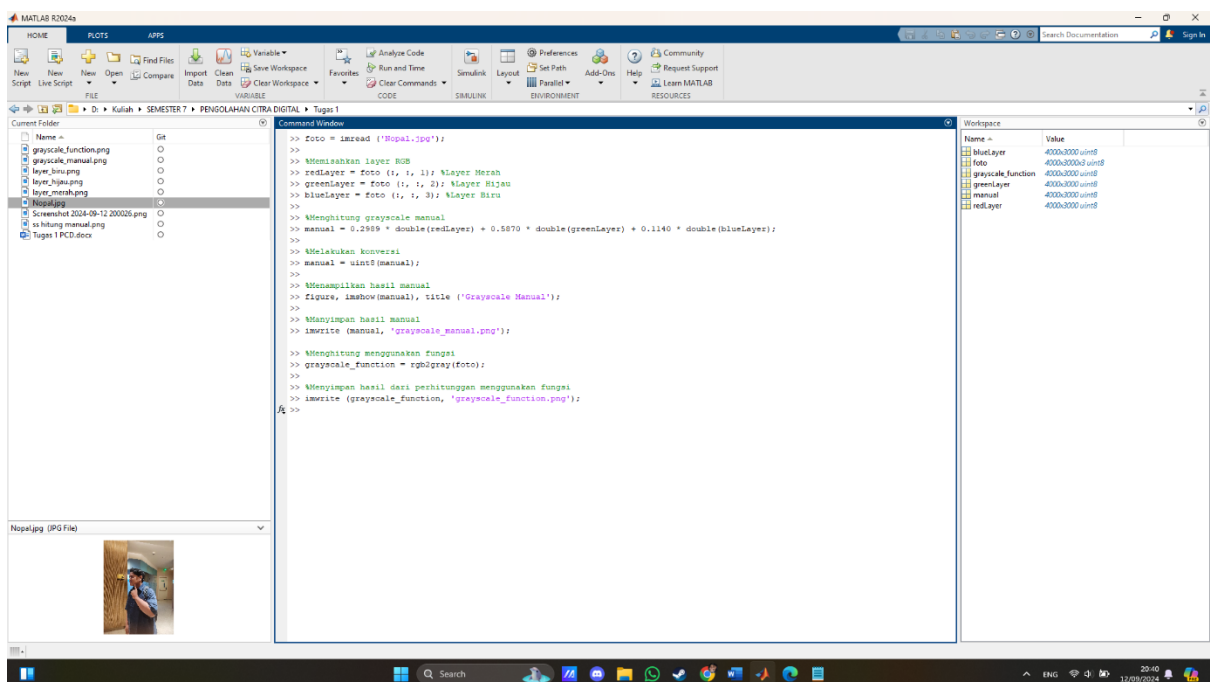
b. Menggunakan Matlab function

Berikut merupakan code dan hasil dari menghitung Grayscale menggunakan Matlab function



MATLAB memiliki fungsi bawaan bernama `rgb2gray` yang dapat langsung mengonversi gambar RGB menjadi grayscale. Grayscale menggunakan `rgb2gray` lebih cepat dan langsung menggunakan fungsi bawaan MATLAB, di mana implementasinya juga menggunakan konsep luminance yang sama.

```
>> foto = imread ('Nopal.jpg');
>>
>> %Memisahkan layer RGB
>> redLayer = foto (:, :, 1); %Layer Merah
>> greenLayer = foto (:, :, 2); %Layer Hijau
>> blueLayer = foto (:, :, 3); %Layer Biru
>>
>> %Menghitung menggunakan fungsi
>> grayscale_function = rgb2gray(foto);
>>
>> %Menyimpan hasil dari perhitungan menggunakan
fungsi
>> imwrite (grayscale_function,
'grayscale_function.png');
```



3. Negative

Berikut merupakan code dan hasil dari pengolahan gambar menjadi negative



255 - gambar: Pada setiap piksel, kita mengurangi nilai intensitas warna dari 255. Untuk gambar dengan tipe data uint8, nilai piksel berada di antara 0 dan 255. Operasi ini membalik nilai intensitas sehingga:

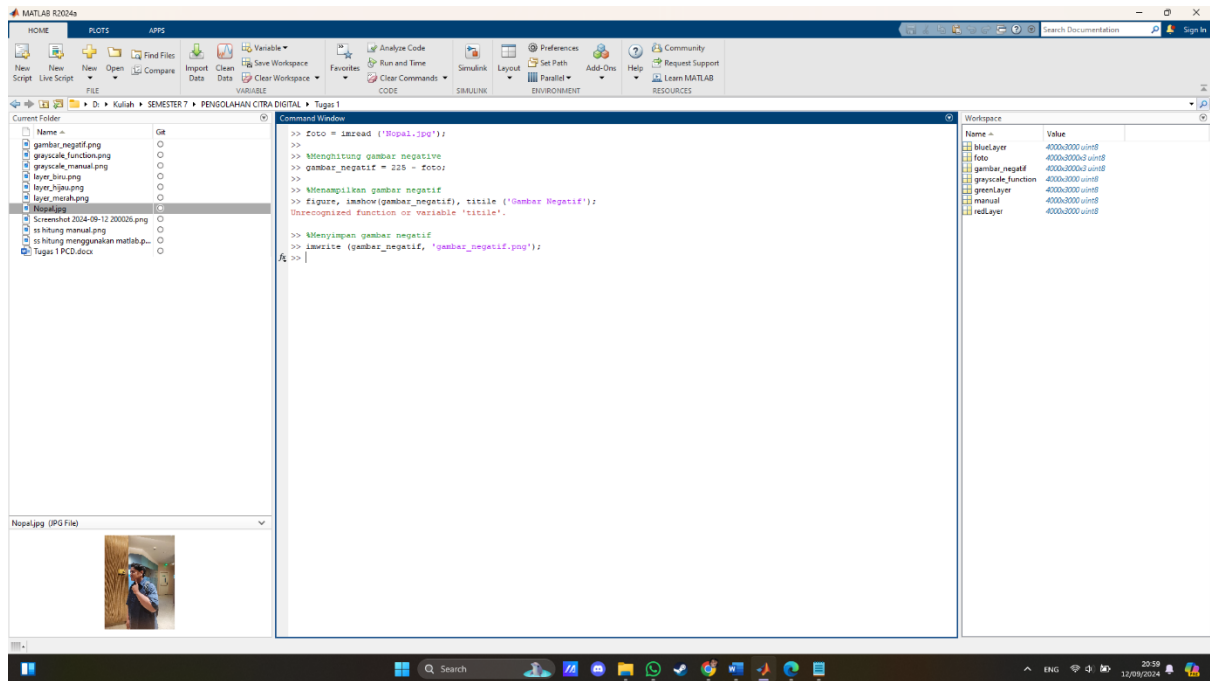
- Piksel yang awalnya cerah (misalnya, nilai mendekati 255) menjadi gelap (mendekati 0).
- Piksel yang awalnya gelap (misalnya, nilai mendekati 0) menjadi cerah (mendekati 255).
- Gambar negatif akan memperlihatkan efek seolah warna-warna dalam gambar asli dibalik menjadi warna komplementernya.

Kesimpulannya adalah gambar asli yang diinputkan akan ditampilkan terlebih dahulu sedangkan gambar negatif adalah gambar yang sudah diolah menjadi negatif yang akan ditampilkan di jendela terpisah.

```
>> foto = imread ('Nopal.jpg');  
>>  
>> %Menghitung gambar negative  
>> gambar_negatif = 255 - foto;  
>>
```

```
>> %Menampilkan gambar negatif
>> figure, imshow(gambar_negatif), titile ('Gambar
Negatif');
Unrecognized function or variable 'titile'.

>> %Menyimpan gambar negatif
>> imwrite (gambar_negatif, 'gambar_negatif.png');
```



4. Log Transformation

Berikut merupakan code dan hasil dari pengolahan gambar menjadi Log Transformation



Pada gambar RGB di MATLAB, prinsip utamanya adalah mengubah nilai intensitas piksel gambar menggunakan logarithmic scaling. Log transformation bertujuan untuk memperluas intensitas piksel rendah dan mengompres intensitas piksel tinggi sehingga bisa lebih jelas dalam gambar yang kontrasnya tinggi. Persamaan dasar untuk transformasi logaritmik adalah:

$$\text{Output} = c \cdot \log(1 + \text{Input})$$

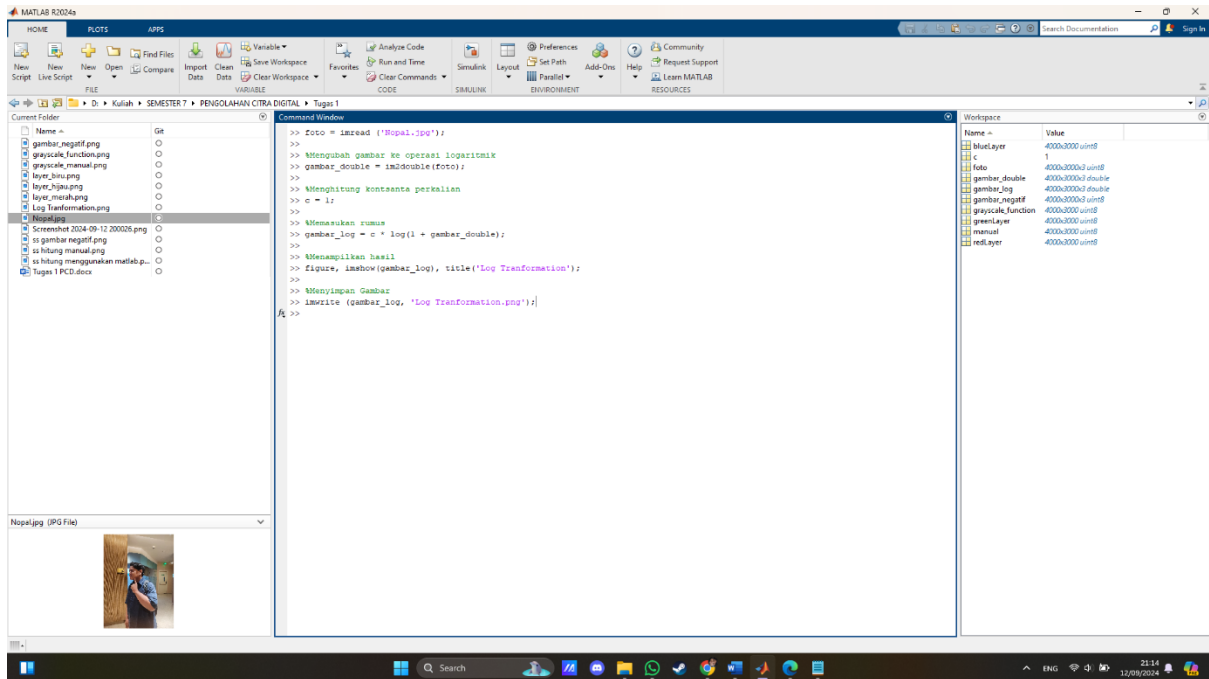
Di mana c adalah konstanta penskalaan, dan Input adalah intensitas piksel yang akan ditransformasikan.

```
>> foto = imread ('Nopal.jpg');
>>
>> %Mengubah gambar ke operasi logaritmik
>> gambar_double = im2double(foto);
>>
>> %Menghitung kongsanta perkalian
>> c = 1;
>>
>> %Memasukan rumus
>> gambar_log = c * log(1 + gambar_double);
>>
```

```

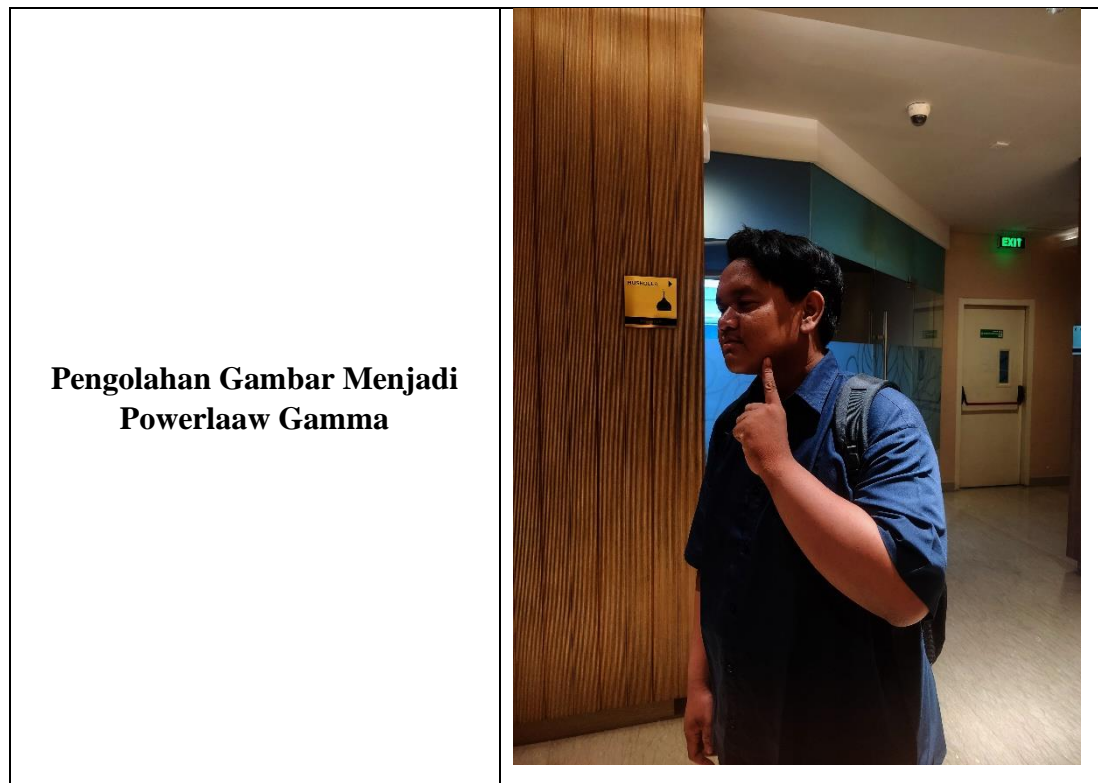
>> %Menampilkan hasil
>>     figure,     imshow(gambar_log),     title('Log
Transformation');
>>
>> %Menyimpan Gambar
>> imwrite (gambar_log, 'Log Transformation.png');

```



5. Powerlaaw Gamma

Berikut merupakan code dan hasil dari pengolahan gambar menjadi Powerlaaw Gamma



Untuk menerapkan Power-Law Transformation (Gamma Correction) pada gambar RGB di MATLAB, prinsip dasarnya adalah mengubah intensitas piksel menggunakan rumus berikut:

$$\text{Output} = c * (\text{Input}) ^\gamma$$

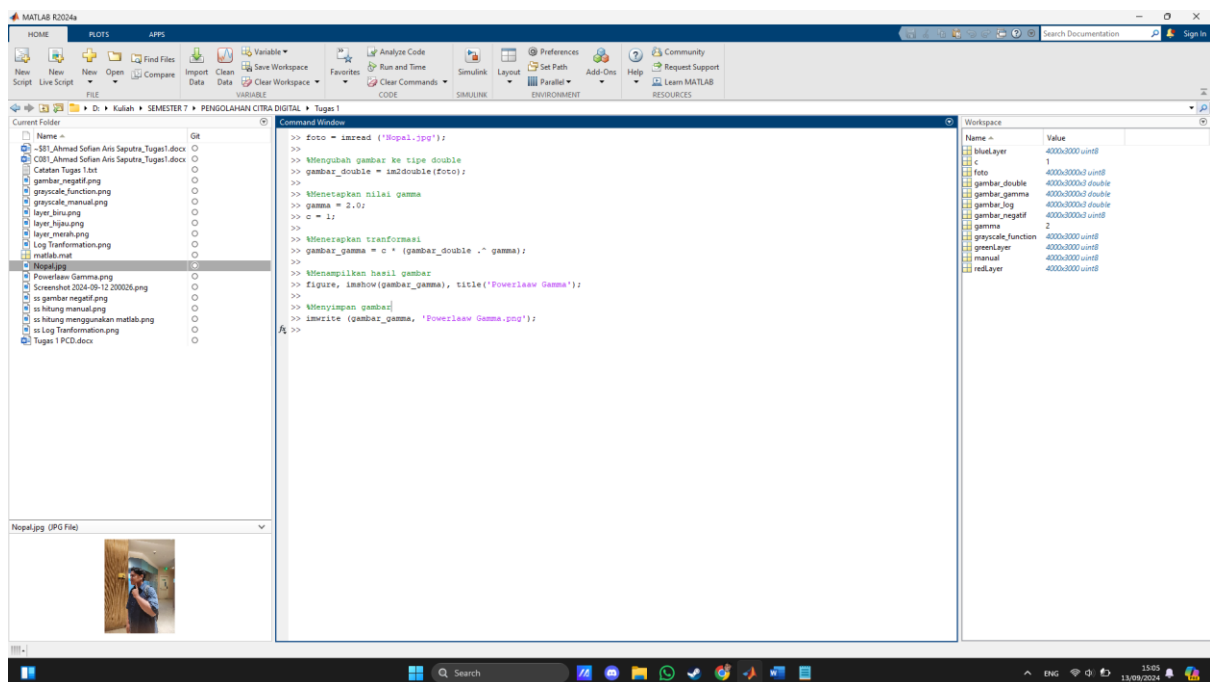
- c : konstanta penskalaan (biasanya 1 jika tidak diperlukan penskalaan tambahan).
- Input : nilai intensitas piksel gambar.
- γ (gamma) : nilai gamma yang mengontrol kecerahan gambar. Nilai $\gamma > 1$ membuat gambar lebih gelap, sedangkan $\gamma < 1$ membuat gambar lebih cerah.

Gamma Correction adalah proses yang melibatkan penggunaan eksponen pada nilai intensitas piksel untuk mengatur kecerahan gambar. Nilai gamma menentukan tingkat koreksi, di mana jika gamma lebih besar dari 1 ($\gamma > 1$), gambar akan menjadi lebih gelap, sedangkan jika gamma lebih kecil dari 1 ($\gamma < 1$), gambar akan terlihat lebih cerah. Sebelum melakukan operasi ini, gambar dikonversi terlebih dahulu ke tipe data double agar operasi eksponensial dapat dilakukan dengan presisi yang lebih tinggi. Dalam proses koreksi gamma, operasi eksponensial diterapkan secara element-wise pada setiap piksel di gambar menggunakan operator $.^{\gamma}$, sehingga setiap piksel mendapatkan efek koreksi sesuai nilai gamma yang diterapkan.

```

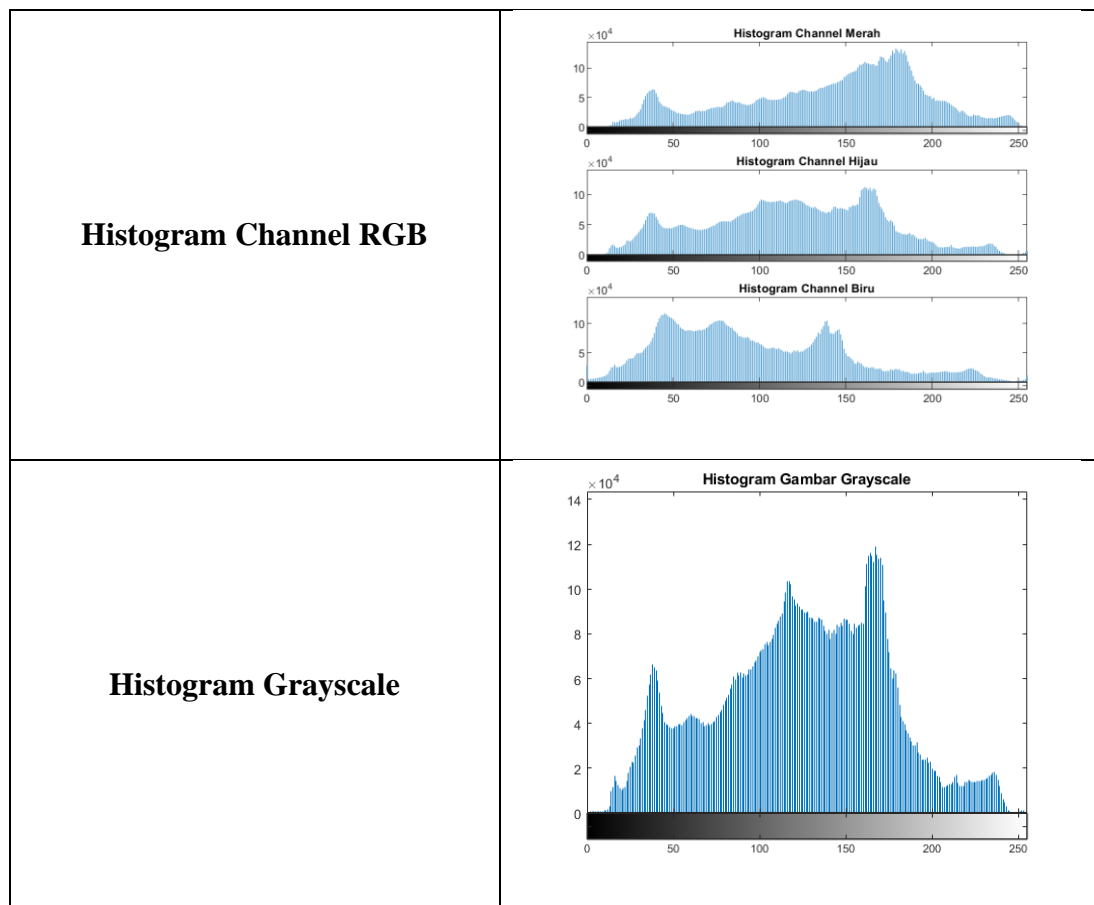
>> foto = imread ('Nopal.jpg');
>>
>> %Mengubah gambar ke tipe double
>> gambar_double = im2double(foto);
>>
>> %Menetapkan nilai gamma
>> gamma = 2.0;
>> c = 1;
>>
>> %Menerapkan tranformasi
>> gambar_gamma = c * (gambar_double .^ gamma);
>>
>> %Menampilkan hasil gambar
>> figure, imshow(gambar_gamma), title('Powerlaaw
Gamma');
>>
>> %Menyimpan gambar
>> imwrite (gambar_gamma, 'Powerlaaw Gamma.png');

```



6. Tampilkan Histogramnya

Berikut merupakan code dan hasil dari menampilkan Histogram gambarnya Setiap Channel RGB dan Grayscale



Histogram menunjukkan distribusi intensitas piksel dalam gambar, dan sangat berguna untuk analisis citra, seperti melihat kontras atau distribusi kecerahan gambar. Histogram untuk gambar grayscale menampilkan distribusi intensitas piksel dari 0 (hitam) hingga 255 (putih), yang menggambarkan tingkat kecerahan gambar. Sementara itu, histogram untuk gambar RGB menampilkan distribusi intensitas masing-masing channel warna (merah, hijau, dan biru), sehingga membantu memahami distribusi warna dalam gambar.

Pada hasil akhirnya, histogram gambar grayscale menunjukkan distribusi kecerahan, sedangkan histogram untuk gambar RGB (Red, Green, Blue) menunjukkan bagaimana intensitas warna merah, hijau, dan biru tersebar di dalam gambar tersebut.

```
>> foto = imread ('Nopal.jpg');  
>>  
>> %Mengkonversi gambar ke Grayscale  
>> gambar_gray = rgb2gray(foto);  
>>  
>> %Menampilkan hasil Histogram gambar Grayscale  
>> figure, imhist(gambar_gray), title('Histogram  
Gambar Grayscale');
```

```

>>
>> %Menyimpan Histogram gambar Grayscale
>> saveas(gcf, 'histogram_grayscale.png');
>>
>> %Memisahkan channel RGB
>> redLayer = foto(:, :, 1); %Layer Merah
>> greenLayer = foto(:, :, 2); %Layer Hijau
>> blueLayer = foto(:, :, 3); %Layer Biru
>>
>> %Menampilkan histogram untuk masing-masing channel
RGB
>> figure,
>> subplot(3,1,1), imhist(redLayer), title('Histogram
Channel Merah');
>>         subplot(3,1,2),           imhist(greenLayer),
title('Histogram Channel Hijau');
>>         subplot(3,1,3),           imhist(blueLayer),
title('Histogram Channel Biru');
>>
>> %Menyimpan hasil histogram dari RGB
>> saveas(gcf, 'histogram_RGB.png');

```

