

Tugas 1 IF4073 Pemrosesan Citra Digital

Image Enhancement



Disusun Oleh:

13521042 Kevin John Wesley Hutabarat

13521059 Arleen Chrysantha Gunardi

**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2024**

Daftar Isi

Daftar Isi.....	2
Daftar Tabel.....	3
Daftar Gambar.....	4
1. Screenshot GUI.....	6
2. Penjelasan Program.....	6
a. Histogram.....	6
b. Perbaikan Kualitas Citra.....	10
i. Image Brightening.....	10
ii. Citra Negatif dan Balikannya.....	13
iii. Transformasi Log.....	16
iv. Transformasi Pangkat.....	20
v. Peregangan Kontras.....	23
c. Perataan Histogram.....	26
d. Perbaikan Citra dengan Histogram Specification (Histogram Matching).....	30
3. Alamat GitHub.....	35

Daftar Tabel

Tabel 2.1 Kode program histogram.....	6
Tabel 2.2 Kode program brightening.....	10
Tabel 2.3 Kode program citra negatif.....	13
Tabel 2.4 Kode program transformasi log.....	17
Tabel 2.5 Kode program transformasi pangkat/eksponen.....	20
Tabel 2.6 Kode program transformasi peregangan kontras.....	23
Tabel 2.7 Kode program perataan histogram.....	26
Tabel 2.8 Kode program perataan histogram.....	30

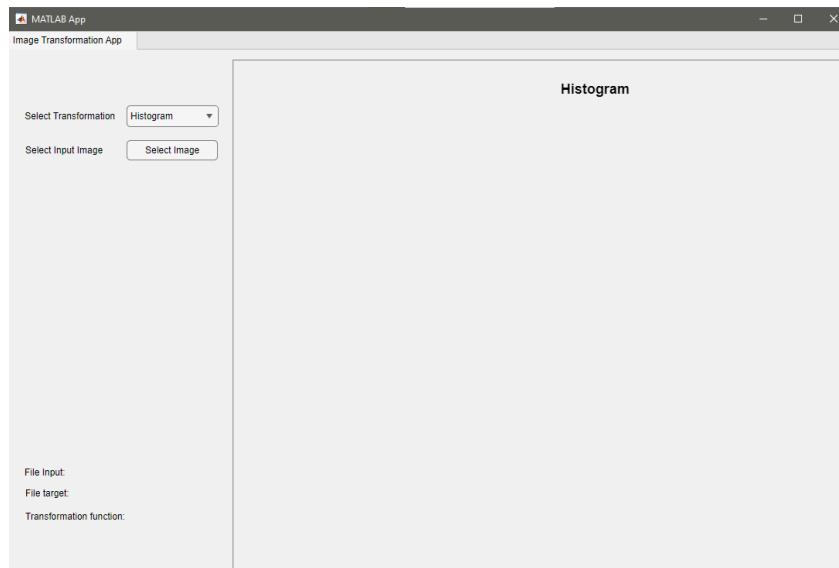
Daftar Gambar

Gambar 1.1 Screenshot GUI.....	6
Gambar 2.1.1.1 Histogram dari citra boat.bmp.....	7
Gambar 2.1.1.2 Histogram dari citra galaxy_pair_original.jpg.....	8
Gambar 2.1.1.3 Histogram dari citra peppers.bmp.....	8
Gambar 2.1.1.4 Histogram dari citra golhill.bmp.....	8
Gambar 2.1.2.1 Histogram dari citra baboon24.bmp.....	8
Gambar 2.1.2.2 Histogram dari citra butterfly.bmp.....	9
Gambar 2.1.2.3 Histogram dari citra strawberries_fullcolor.jpg.....	9
Gambar 2.1.2.4 Histogram dari citra caster_stand_original.jpg.....	9
Gambar 2.2.1.1 Pencerahan dari citra peppers512warna.bmp.....	10
Gambar 2.2.1.2 Pencerahan dari citra city_lowcontrast.jpg.....	11
Gambar 2.2.1.3 Pencerahan dari citra caster_stand_original.jpg.....	11
Gambar 2.2.1.4 Pencerahan dari citra pollens_lowcontrast.jpg.....	11
Gambar 2.2.1.5 Pencerahan dari citra lofotenisland_lowcontrast.jpg.....	11
Gambar 2.2.1.6 Pencerahan dari citra flower.jpg.....	12
Gambar 2.2.1.7 Pencerahan dari citra tugas1.jpg.....	12
Gambar 2.2.2.1 Negatif dari citra baboon.bmp.....	14
Gambar 2.2.2.2 Negatif dari citra caster_stand_original.jpg.....	14
Gambar 2.2.2.3 Negatif dari citra flower.jpg.....	14
Gambar 2.2.2.4 Negatif dari citra strawberries_fullcolor.jpg.....	15
Gambar 2.2.2.5 Negatif dari citra goldhill.bmp.....	15
Gambar 2.2.2.6 Negatif dari citra peppers.bmp.....	15
Gambar 2.2.2.7 Negatif dari citra pollens_lowcontrast.jpg.....	16
Gambar 2.2.2.8 Negatif dari citra mountain_negative.bmp.....	16
Gambar 2.2.3.1 Transformasi log dari citra bottle.bmp.....	17
Gambar 2.2.3.2 Transformasi log dari citra island.bmp.....	18
Gambar 2.2.3.3 Transformasi log dari citra building.bmp.....	18
Gambar 2.2.3.4 Transformasi log dari citra caster_stand_original.jpg.....	18
Gambar 2.2.3.5 Transformasi log dari citra pollens_lowcontrast.jpg.....	19
Gambar 2.2.3.6 Transformasi log dari citra peppers.bmp.....	19
Gambar 2.2.3.7 Transformasi log dari citra goldhill.bmp.....	19
Gambar 2.2.4.1 Transformasi pangkat dari citra beach.bmp.....	21
Gambar 2.2.4.2 Transformasi pangkat dari citra pollens_lowcontrast.jpg.....	21
Gambar 2.2.4.3 Transformasi pangkat dari citra pollens_lowcontrast.jpg.....	21
Gambar 2.2.4.4 Transformasi pangkat dari citra pollens_lowcontrast.jpg.....	22
Gambar 2.2.4.5 Transformasi pangkat dari citra peppers.bmp.....	22

Gambar 2.2.4.6 Transformasi pangkat dari citra pollens_lowcontrast.bmp.....	22
Gambar 2.2.4.7 Transformasi pangkat dari citra galaxy_pair_original.jpg.....	23
Gambar 2.2.5.1 Peregangan kontras dari citra city_lowcontrast.jpg.....	24
Gambar 2.2.5.2 Peregangan kontras dari citra pollens_lowcontrast.jpg.....	24
Gambar 2.2.5.3 Peregangan kontras dari citra girl.bmp.....	25
Gambar 2.2.5.4 Peregangan kontras dari citra city_during_day.jpg.....	25
Gambar 2.2.5.5 Peregangan kontras dari citra city_lowcontrast.jpg.....	25
Gambar 2.2.5.6 Peregangan kontras dari citra galaxy_pair_original.jpg.....	26
Gambar 2.3.1 Perataan histogram dari citra pollens_lowcontrast.jpg.....	27
Gambar 2.3.2 Perataan histogram dari citra lofotenisland_lowcontrast.jpg.....	27
Gambar 2.3.3 Perataan histogram dari citra pollens_lowcontrast.jpg.....	27
Gambar 2.3.4 Perataan histogram dari citra pollens_lowcontrast.jpg.....	28
Gambar 2.3.5 Perataan histogram dari citra barbara.bmp.....	28
Gambar 2.3.6 Perataan histogram dari citra horse_in_fog.png.....	28
Gambar 2.3.7 Perataan histogram dari citra field.png.....	29
Gambar 2.4.1 Histogram matching citra bridge.png terhadap citra goldhill.bmp.....	31
Gambar 2.4.2 Histogram matching citra bridge.png terhadap citra baboon24.bmp.....	32
Gambar 2.4.3 Histogram matching citra girl.bmp terhadap citra goldhill.bmp.....	32
Gambar 2.4.4 Histogram matching citra peppers.bmp terhadap citra city_lowcontrast.jpg.....	33
Gambar 2.4.5 Histogram matching citra lofotenisland_lowcontrast.jpg terhadap citra baboon24.bmp.....	33
Gambar 2.4.6 Histogram matching citra lofotenisland_lowcontrast.jpg terhadap citra pollens_lowcontrast.jpg.....	34
Gambar 2.4.7 Histogram matching citra flower.jpg terhadap citra peppers.bmp.....	34
Gambar 2.4.8 Histogram matching citra bridge.bmp terhadap citra sunflower.bmp.....	35

1. Screenshot GUI

Berikut adalah tangkapan layar untuk tampilan GUI.



Gambar 1.1 Screenshot GUI

2. Penjelasan Program

a. Histogram

i. Kode Program

Tabel 2.1 Kode program histogram

```
% Plot the histogram
function plot_histogram(hist_data, img_title, color)
    bar(0:255, hist_data, 'FaceColor', color, 'EdgeColor', color);
    title(img_title);
    xlabel('Intensity Level');
    ylabel('Frequency');
    xlim([0 255]);
end

% Compute histogram data
function hist_data = compute_histogram(channel)
    hist_data = zeros(1, 256);

    for i = 1:size(channel, 1)
        for j = 1:size(channel, 2)
            intensity = channel(i, j);
            hist_data(intensity + 1) = hist_data(intensity + 1) + 1;
        end
    end
end

% Show histogram on the app
```

```

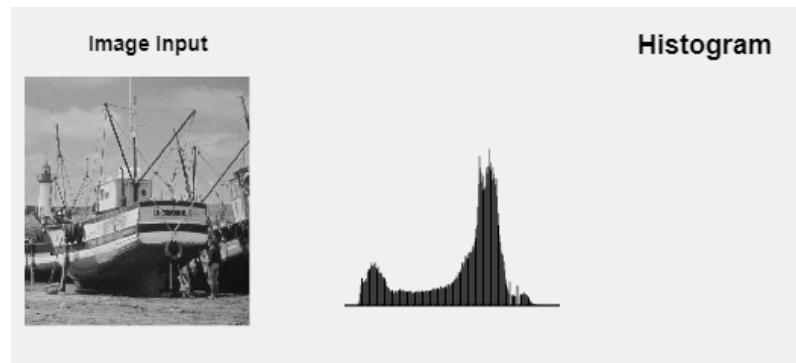
function hist_app(app, image, redAxes, greenAxes, blueAxes)
    if size(image, 3) == 1
        cla(redAxes);
        cla(greenAxes);
        cla(blueAxes);
        % Grayscale image, hanya menggunakan redAxes
        histogram_data = compute_histogram(image);
        plot_histogram_on_axes(histogram_data, 'Grayscale', redAxes,
    'k');
    else
        cla(redAxes);
        cla(greenAxes);
        cla(blueAxes);
        % Colored image, menggunakan ketiga axes
        r_hist = compute_histogram(image(:, :, 1));
        g_hist = compute_histogram(image(:, :, 2));
        b_hist = compute_histogram(image(:, :, 3));

        % Plot pada axes yang diberikan
        plot_histogram_on_axes(r_hist, 'Red', redAxes, 'r');
        plot_histogram_on_axes(g_hist, 'Green', greenAxes, 'g');
        plot_histogram_on_axes(b_hist, 'Blue', blueAxes, 'b');
    end
end

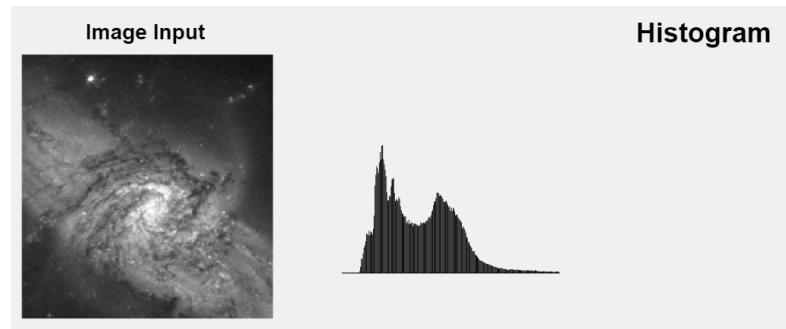
% Plot histogram on axes
function plot_histogram_on_axes(histogram_data, color_name,
axesHandle, color)
    bar(axesHandle, histogram_data, 'FaceColor', color);
    title(axesHandle, [color_name, ' Histogram']);
    xlabel(axesHandle, 'Intensity');
    ylabel(axesHandle, 'Frequency');
end

```

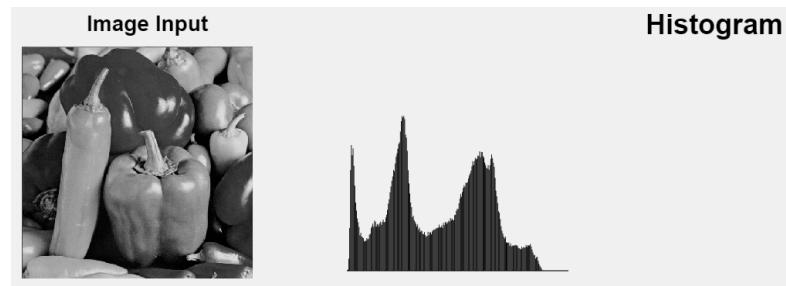
ii. Citra *Grayscale*



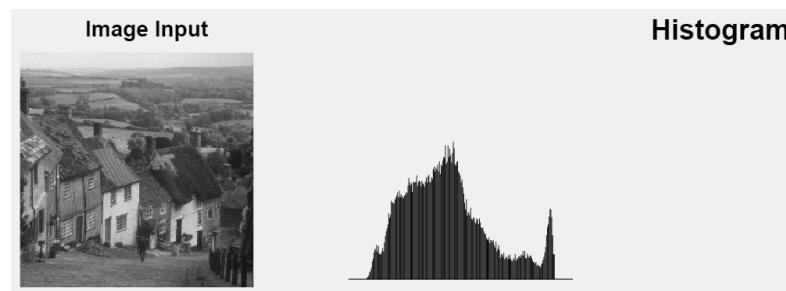
Gambar 2.1.1.1 Histogram dari citra boat.bmp



Gambar 2.1.1.2 Histogram dari citra galaxy_pair_original.jpg

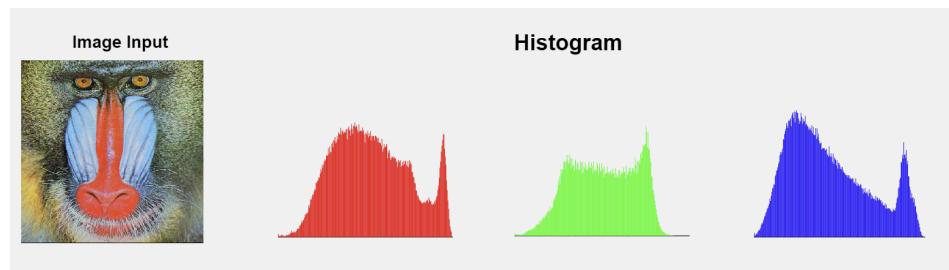


Gambar 2.1.1.3 Histogram dari citra peppers.bmp

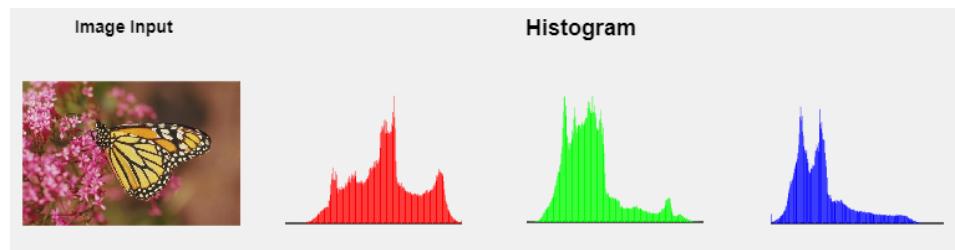


Gambar 2.1.1.4 Histogram dari citra golhill.bmp

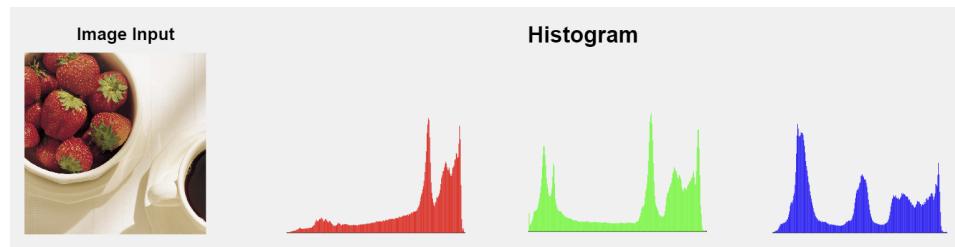
iii. Citra Berwarna



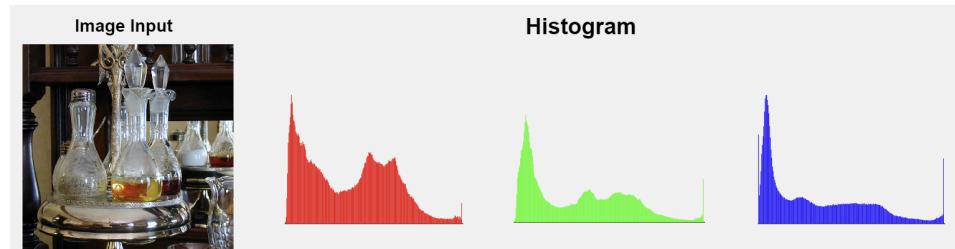
Gambar 2.1.2.1 Histogram dari citra baboon24.bmp



Gambar 2.1.2.2 Histogram dari citra butterfly.bmp



Gambar 2.1.2.3 Histogram dari citra strawberries_fullcolor.jpg



Gambar 2.1.2.4 Histogram dari citra caster_stand_original.jpg

iv. Analisis Cara Kerja Program

Histogram citra merepresentasikan persebaran nilai intensitas piksel dari suatu citra. Cara menghitung setiap nilai intensitas pada histogram adalah dengan mengiterasi setiap baris dan kolom citra dan menghitung frekuensi kemunculan intensitas tersebut di setiap piksel. Perhitungan frekuensi kemunculan ini dilakukan pada fungsi `compute_histogram`.

Setelah dihitung, histogram ditampilkan dalam bentuk grafik batang melalui fungsi `plot_histogram_on_axes`. Untuk citra `grayscale`, hanya ditampilkan satu grafik dengan warna batang abu karena hanya memiliki satu kanal warna, sementara citra berwarna menampilkan tiga grafik untuk kanal warna merah, hijau, dan biru.

Grafik histogram terdiri dari sumbu-X dan sumbu-Y. Sumbu-X merepresentasikan derajat keabuan citra. Untuk citra dengan 256 derajat keabuan, terdapat 256 nilai *axis* yang ditampilkan pada histogram. Sementara itu, sumbu-Y adalah nilai frekuensi kemunculan intensitas (derajat keabuan) pada citra.

b. Perbaikan Kualitas Citra

i. Image Brightening

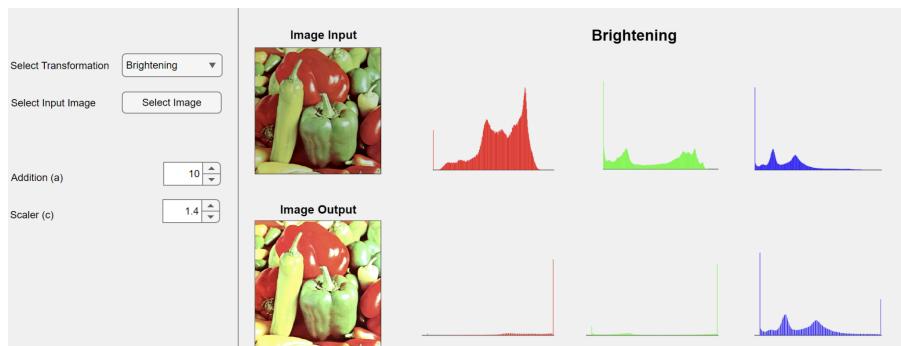
Berikut fungsi yang digunakan untuk mengatur kecerahan citra.

Tabel 2.2 Kode program *brightening*

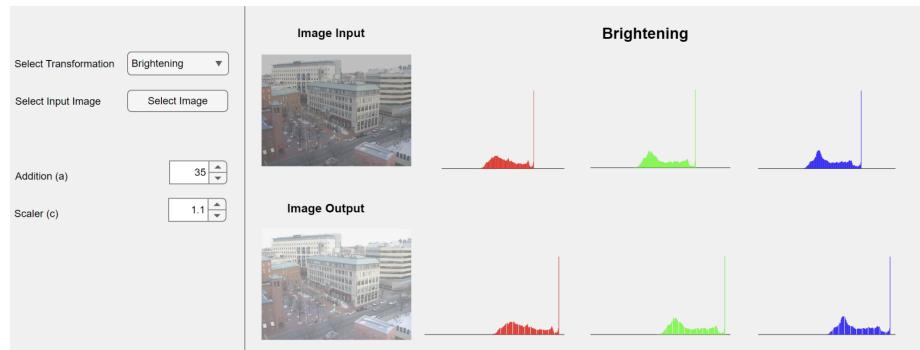
```
% brightening.m
% Image brightening s = a*r + b
function output_image = brightening(image, a, b)
    [rows, cols, color_channels] = size(image);
    output_image = zeros(rows, cols, 'uint8');
    for i = 1:rows
        for j = 1:cols
            for k = 1:color_channels
                temp = double(a * image(i,j,k) + b); % pixel dikalikan dengan a dan ditambah b
                % Clipping
                if temp < 0
                    output_image(i,j,k) = 0;
                elseif temp > 255
                    output_image(i,j,k) = 255;
                else
                    output_image(i,j,k) = uint8(temp);
                end
            end
        end
    end
end
```

Fungsi *brightening* menerima parameter berupa *image*, *a*, dan *b*. *image* adalah citra yang akan diubah, *a* adalah pengali pada citra, dan *b* adalah konstanta penambah kecerahan citra.

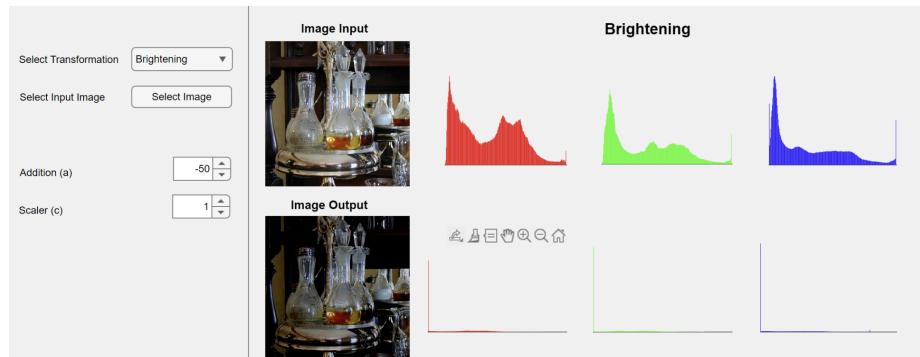
Berikut adalah contoh ketika program dijalankan:



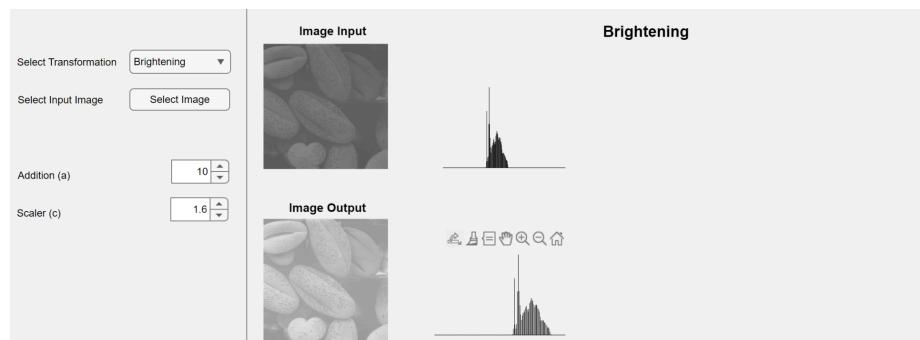
Gambar 2.2.1.1 Pencerahan dari citra peppers512warna.bmp



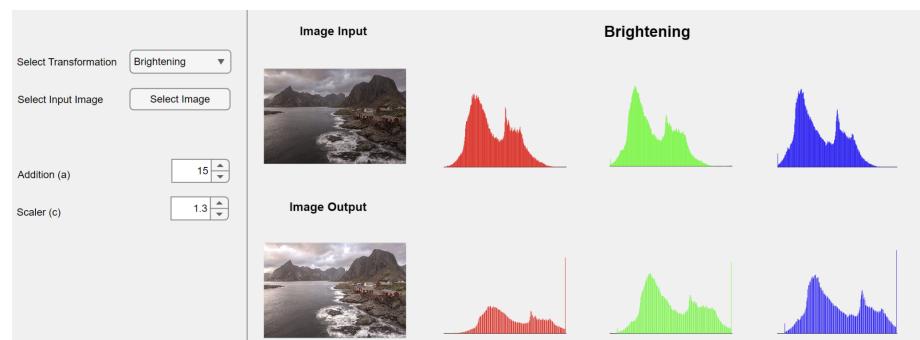
Gambar 2.2.1.2 Pencerahan dari citra city_lowcontrast.jpg



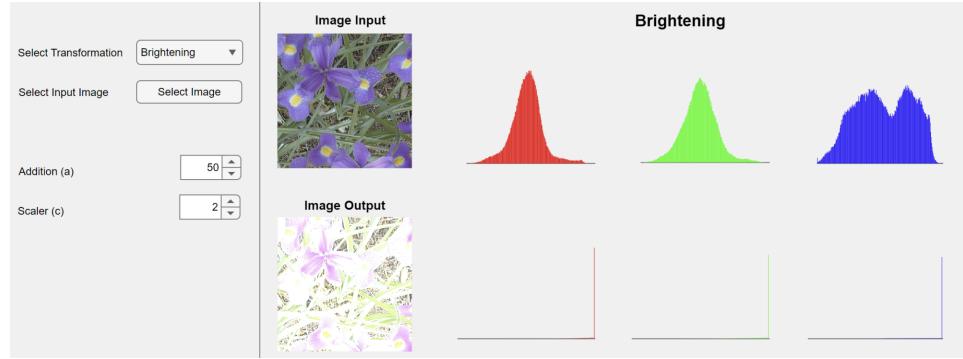
Gambar 2.2.1.3 Pencerahan dari citra caster_stand_original.jpg



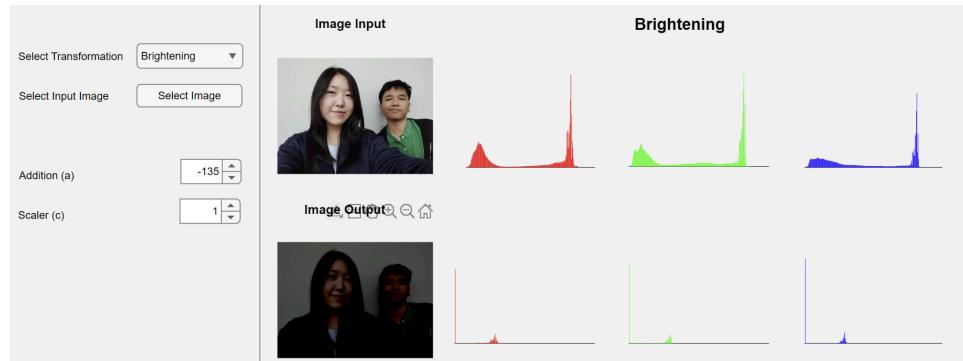
Gambar 2.2.1.4 Pencerahan dari citra pollens_lowcontrast.jpg



Gambar 2.2.1.5 Pencerahan dari citra lofotenisland_lowcontrast.jpg



Gambar 2.2.1.6 Pencerahan dari citra flower.jpg



Gambar 2.2.1.7 Pencerahan dari citra tugas1.jpg

Analisis:

Operasi pencerahan citra dilakukan dengan transformasi linier dengan rumus sebagai berikut:

$$s = ar + b$$

di mana r adalah nilai derajat keabuan (intensitas) piksel citra aslinya, a adalah faktor pengali yang mengatur kontras citra, dan b adalah konstanta penambah atau pengurang kecerahan. Faktor a merupakan bilangan riil positif. Untuk nilai a lebih dari 1, maka kontras citra akan meningkat, sedangkan untuk nilai a kurang dari 1, maka kontras citra akan menurun. Sementara itu, konstanta b merupakan bilangan riil. Untuk nilai b positif (lebih dari 0), maka kecerahan citra meningkat, sedangkan untuk nilai b negatif (kurang dari 0), maka kecerahan citra menurun.

Citra masukan diproses secara iteratif baris dan kolomnya untuk menghitung nilai baru setiap piksel untuk setiap kanal warna. Nilai baru dihitung dengan mengalikan nilai derajat keabuan piksel dengan faktor a , lalu menambahkannya dengan konstanta b .

Setelah dihitung, proses *clipping* dilakukan untuk memastikan nilai derajat keabuannya tetap berada dalam rentang 0 hingga 255. Untuk nilai

yang lebih dari 255 akan di-*clip* menjadi 255 dan nilai yang kurang dari 0 akan di-*clip* menjadi 0.

Untuk mendapatkan kecerahan dan kontras citra yang sesuai, perlu melakukan penyesuaian nilai faktor a dan konstanta b dengan tepat, sehingga hasilnya tidak terlalu gelap (Gambar 2.8) atau terlalu terang (Gambar 2.7).

ii. Citra Negatif dan Balikannya

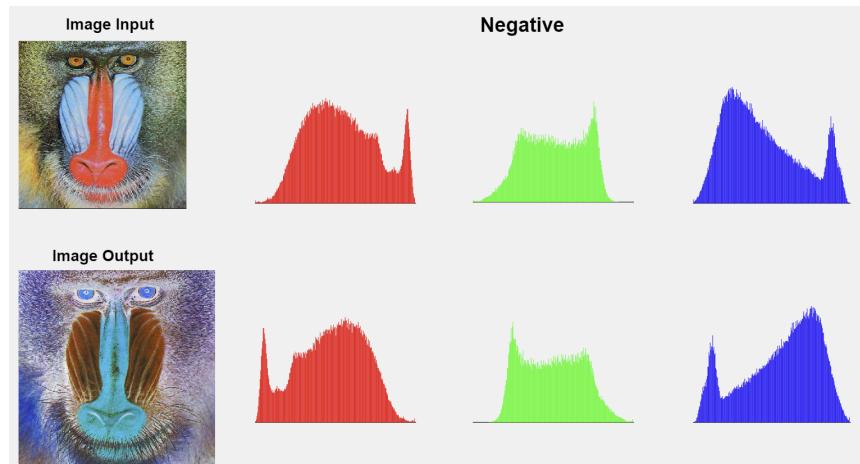
Berikut fungsi yang digunakan untuk membuat citra menjadi negatif.

Tabel 2.3 Kode program citra negatif

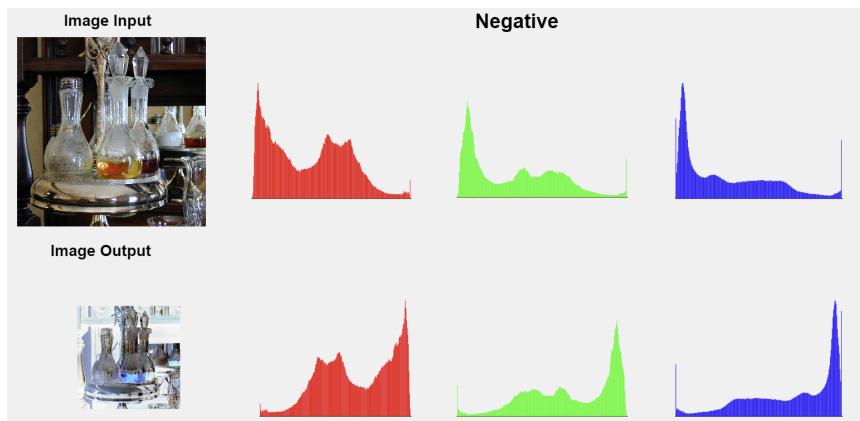
```
% negativeandinverse.m
% Negative image: s = 255 - s
function output_image = negativeandinverse(image)
    [rows, cols, color_channels] = size(image);
    output_image = zeros(rows, cols, 'uint8');
    % iterasi untuk tiap pixel pada citra
    for i = 1:rows
        for j = 1:cols
            for k = 1:color_channels
                % melakukan operasi 255 - s untuk setiap pixel
                temp = 255 - image(i,j,k);
                output_image(i,j,k) = uint8(temp);
            end
        end
    end
end
```

Fungsi `negativeandinverse` menerima parameter berupa `image` yang berupa citra yang akan diubah. Untuk menegatifkan citra, setiap pixel akan diubah nilainya menjadi 255 dikurangi dengan pixel itu sendiri.

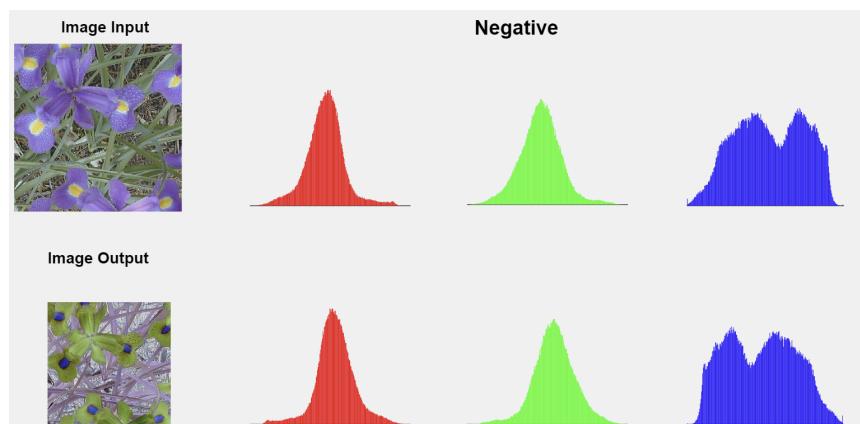
Berikut adalah contoh ketika program dijalankan:



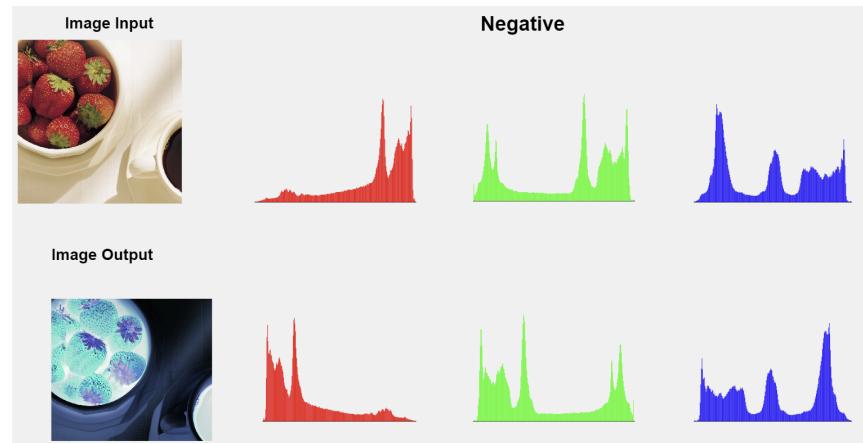
Gambar 2.2.2.1 Negatif dari citra baboon.bmp



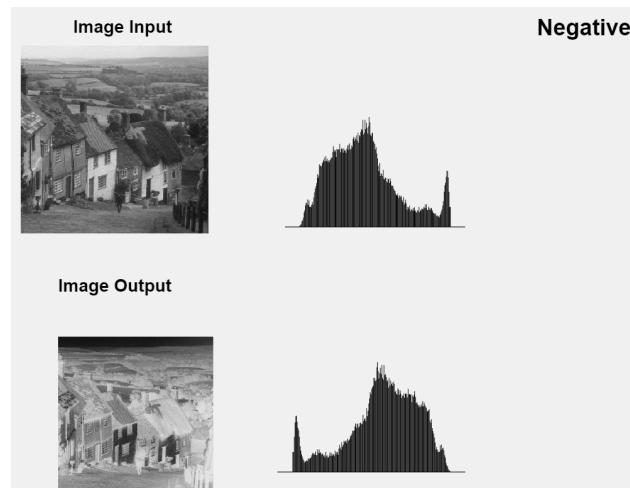
Gambar 2.2.2.2 Negatif dari citra caster_stand_original.jpg



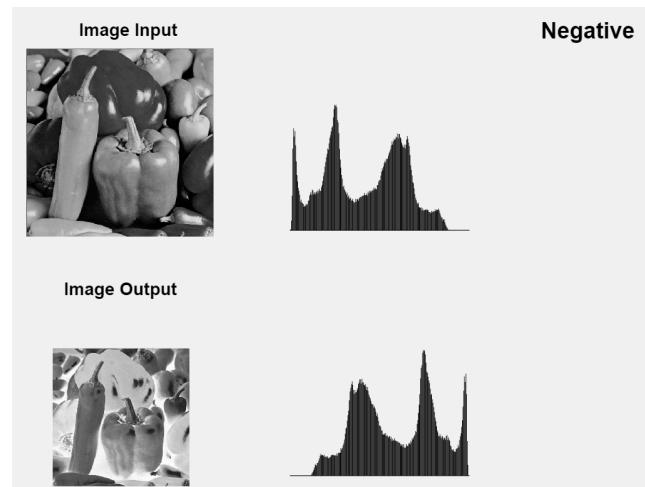
Gambar 2.2.2.3 Negatif dari citra flower.jpg



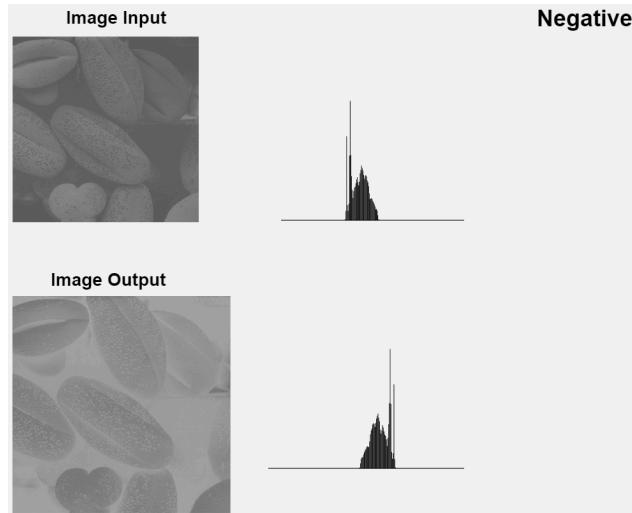
Gambar 2.2.2.4 Negatif dari citra strawberries_fullcolor.jpg



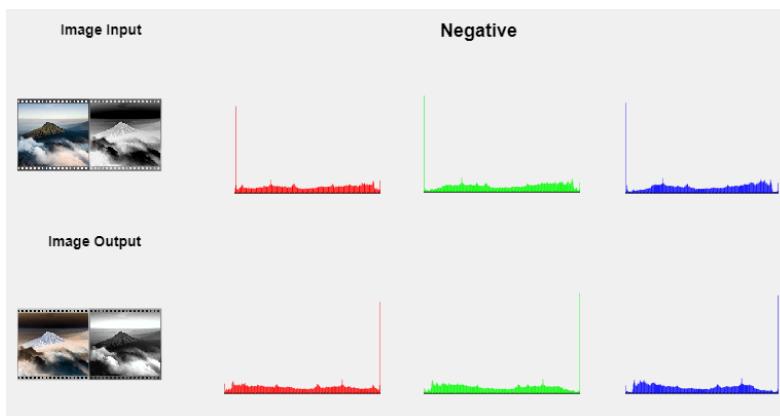
Gambar 2.2.2.5 Negatif dari citra goldhill.bmp



Gambar 2.2.2.6 Negatif dari citra peppers.bmp



Gambar 2.2.2.7 Negatif dari citra pollens_lowcontrast.jpg



Gambar 2.2.2.8 Negatif dari citra mountain_negative.bmp

Analisis:

Operasi negatif citra dilakukan dengan mengubah nilai setiap piksel nya dengan $255 - \text{piksel}$ itu sendiri. Operasi ini akan menghasilkan citra yang terkesan berkebalikan dengan citra semula. Dapat dilihat pada masing-masing histogram, histogram hasil adalah pencerminan dari histogram citra awal terhadap sumbu tengah (sekitar 127,5). Jika mengoperasikan fungsi ini ke citra negatif, fungsi akan mengembalikan citra yang serupa dengan citra masukan awal, seperti pada Gambar 2.2.2.8. Hal ini juga berlaku untuk citra berwarna, di mana fungsi akan diterapkan untuk masing-masing histogram pada tiap kanal.

iii. Transformasi Log

Berikut fungsi yang digunakan untuk mentransformasikan citra dengan fungsi log.

Tabel 2.4 Kode program transformasi log

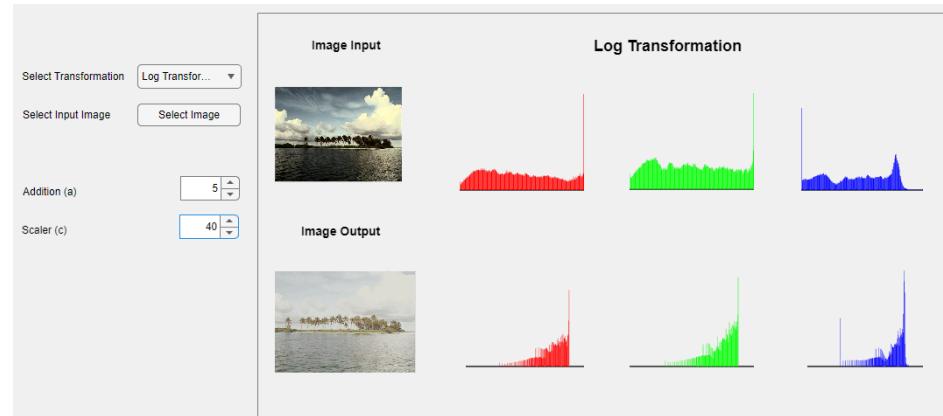
```
% logtransformation.m
% s = c*log(r+1+a)
function output_image = logtransformation(image,c,r)
[rows, cols, color_channels] = size(image);
output_image = zeros(rows, cols, 'uint8');
% iterasi untuk setiap pixel pada citra
for i = 1:rows
    for j = 1:cols
        for k = 1:color_channels
            % menjalankan operasi s = c*log(r+1+a) untuk setiap
            pixel
            temp = c*log(double(image(i,j,k)) + r +1));
            % Clipping
            if temp < 0
                output_image(i,j,k) = 0;
            elseif temp > 255
                output_image(i,j,k) = 255;
            else
                output_image(i,j,k) = uint8(temp);
            end
        end
    end
end
end
```

Fungsi `logtransformation` menerima parameter berupa `image`, `c`, dan `r`. `Image` adalah citra yang akan diubah, sedangkan `c` adalah pengali dan `r` adalah penambah pada pixel di dalam log.

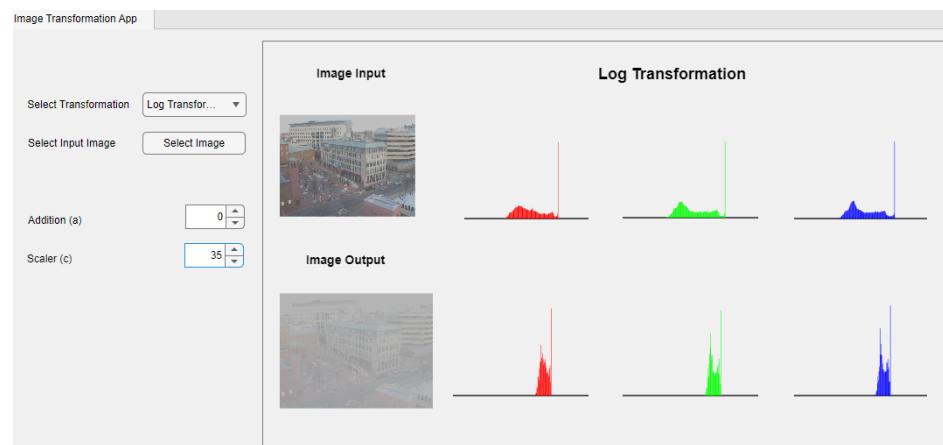
Berikut adalah contoh ketika program dijalankan:



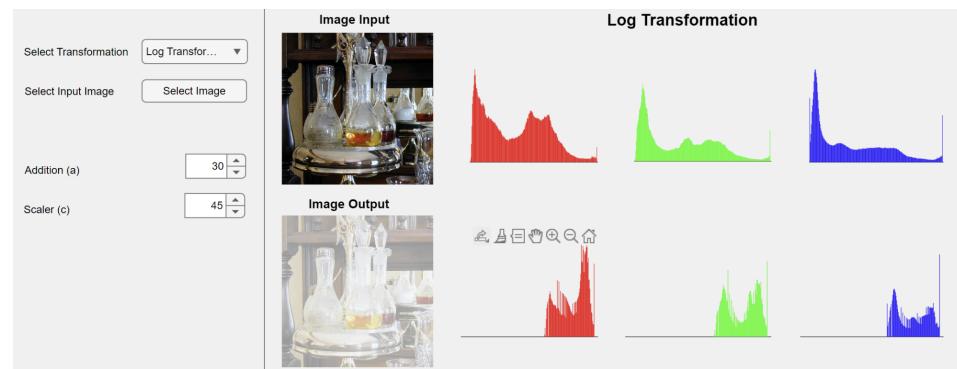
Gambar 2.2.3.1 Transformasi log dari citra bottle.bmp



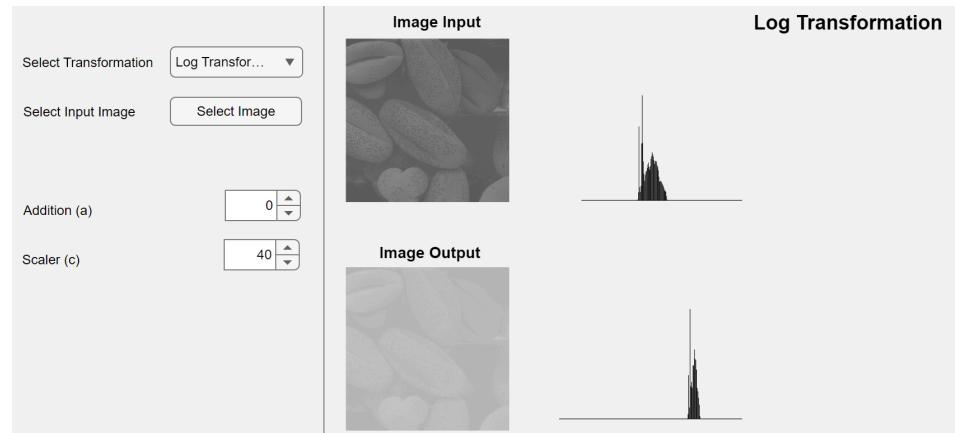
Gambar 2.2.3.2 Transformasi log dari citra island.bmp



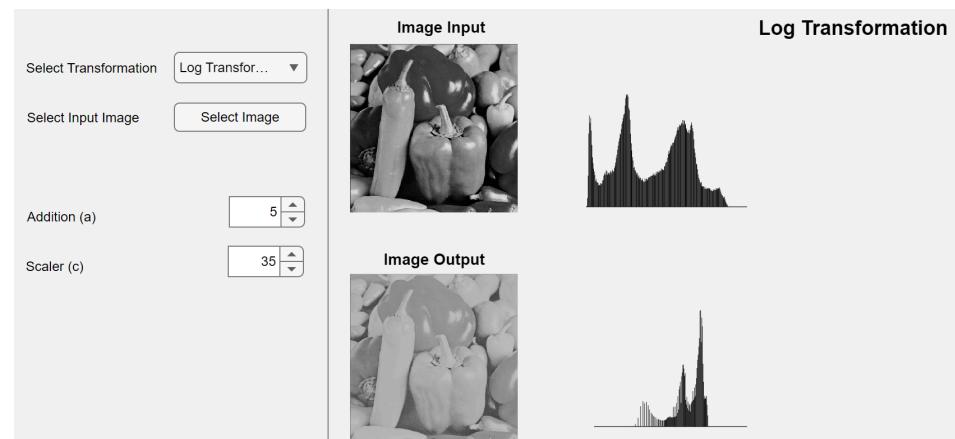
Gambar 2.2.3.3 Transformasi log dari citra building.bmp



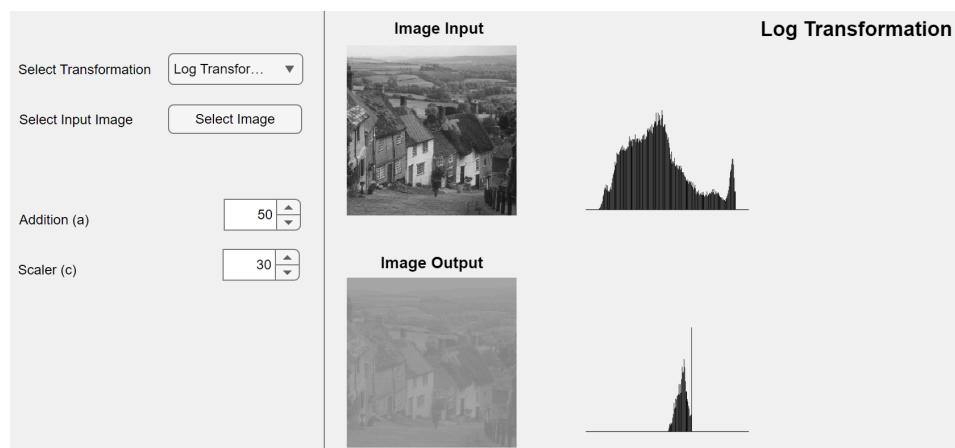
Gambar 2.2.3.4 Transformasi log dari citra caster_stand_original.jpg



Gambar 2.2.3.5 Transformasi log dari citra pollens_lowcontrast.jpg



Gambar 2.2.3.6 Transformasi log dari citra peppers.bmp



Gambar 2.2.3.7 Transformasi log dari citra goldhill.bmp

Analisis:

Fungsi log membuat jarak antar piksel menjadi lebih dekat. Hal ini membuat histogram hasil fungsi log akan lebih rapat dan memiliki kontras yang lebih rendah. Fungsi ini cocok untuk mengolah citra dengan kontras

yang terlalu tinggi. Dengan fungsi ini, citra tersebut dapat diturunkan kontrasnya.

iv. Transformasi Pangkat

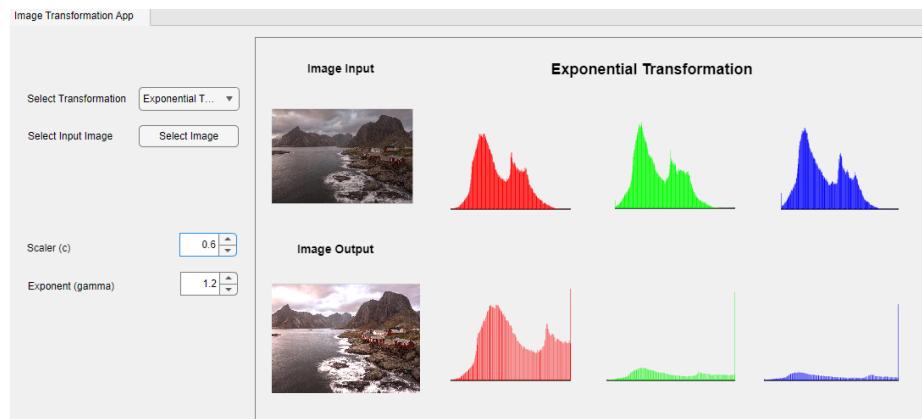
Berikut fungsi yang digunakan untuk mentransformasikan citra dengan fungsi eksponen.

Tabel 2.5 Kode program transformasi pangkat/eksponen

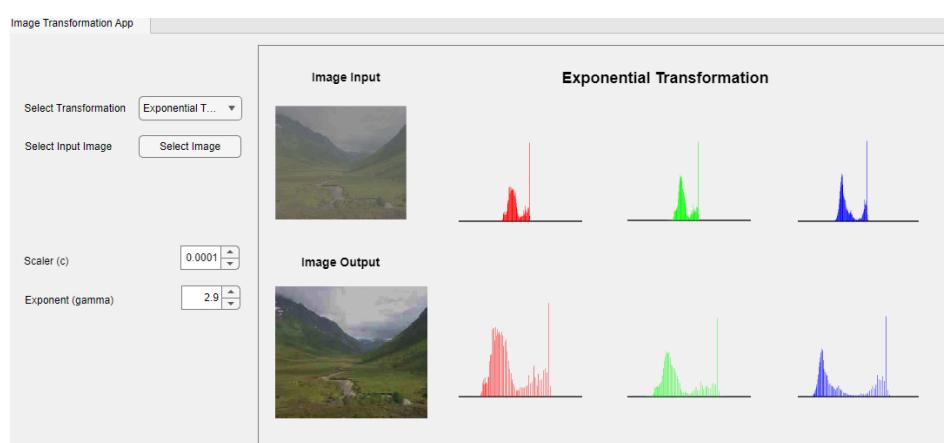
```
% exponent.m
% s = c*s^r
function output_image = exponent(image,c,r)
    [rows, cols, color_channels] = size(image);
    output_image = zeros(rows, cols, 'uint8');
    % iterasi untuk setiap pixel pada citra
    for i = 1:rows
        for j = 1:cols
            for k = 1:color_channels
                % menjalankan operasi s = c*s^r untuk setiap pixel
                temp = c*(double(image(i,j,k))^r); %s = c*s^r
                % Clipping (agar tidak ada nilai pixel yang di luar
                batas)
                if temp < 0
                    output_image(i,j,k) = 0;
                elseif temp > 255
                    output_image(i,j,k) = 255;
                else
                    output_image(i,j,k) = uint8(temp);
                end
            end
        end
    end
end
```

Fungsi exponent menerima parameter berupa `image`, `c`, dan `r`. `Image` adalah citra yang akan diubah, sedangkan `c` adalah pengali dan `r` adalah bilangan yang menjadi derajat pangkat dari pixel pada saat transformasi.

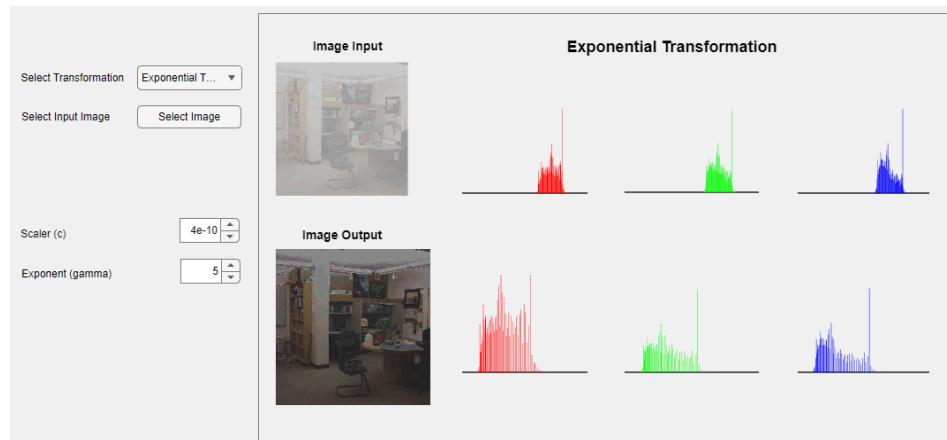
Berikut adalah contoh ketika program dijalankan:



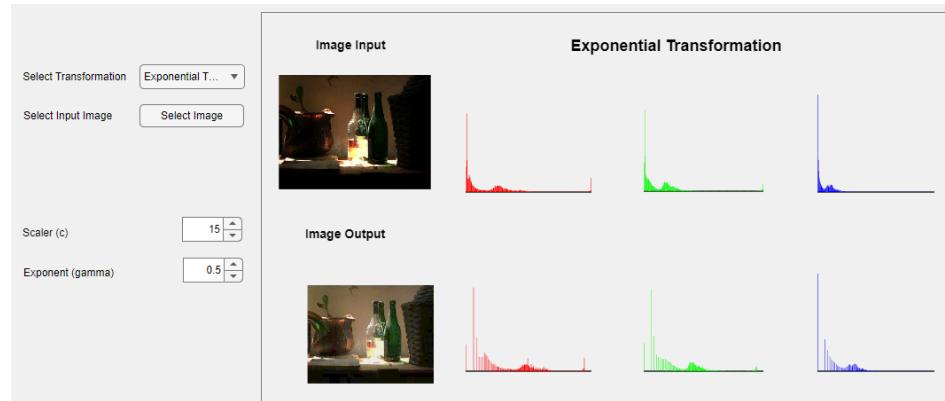
Gambar 2.2.4.1 Transformasi pangkat dari citra beach.bmp



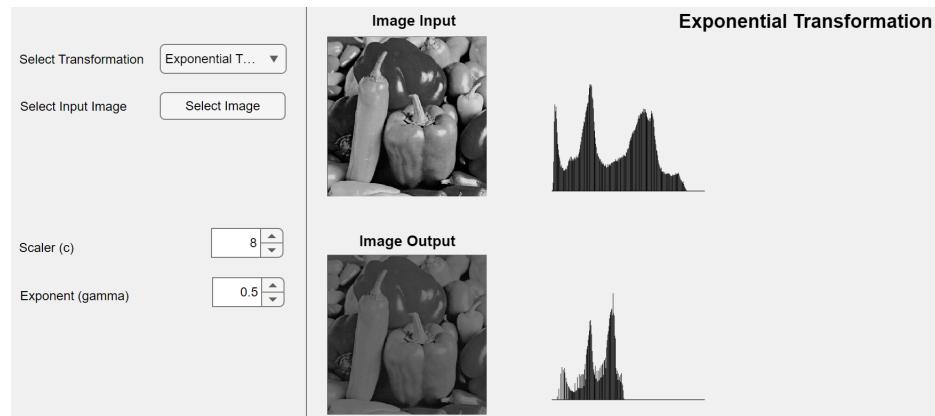
Gambar 2.2.4.2 Transformasi pangkat dari citra pollens_lowcontrast.jpg



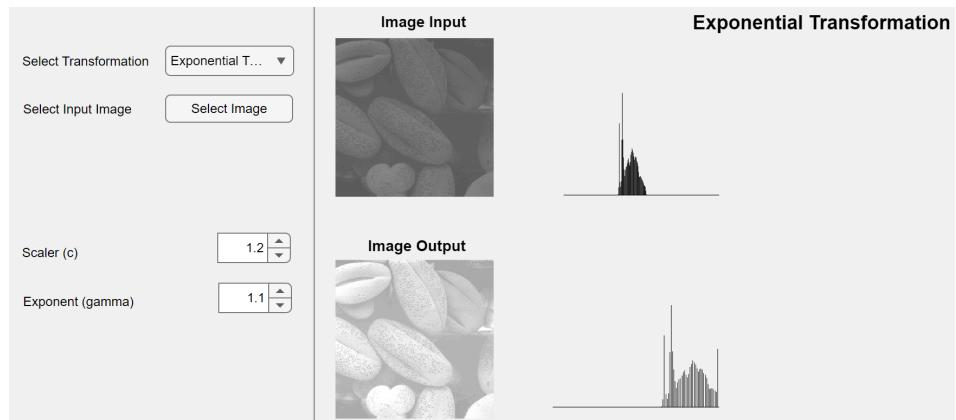
Gambar 2.2.4.3 Transformasi pangkat dari citra pollens_lowcontrast.jpg



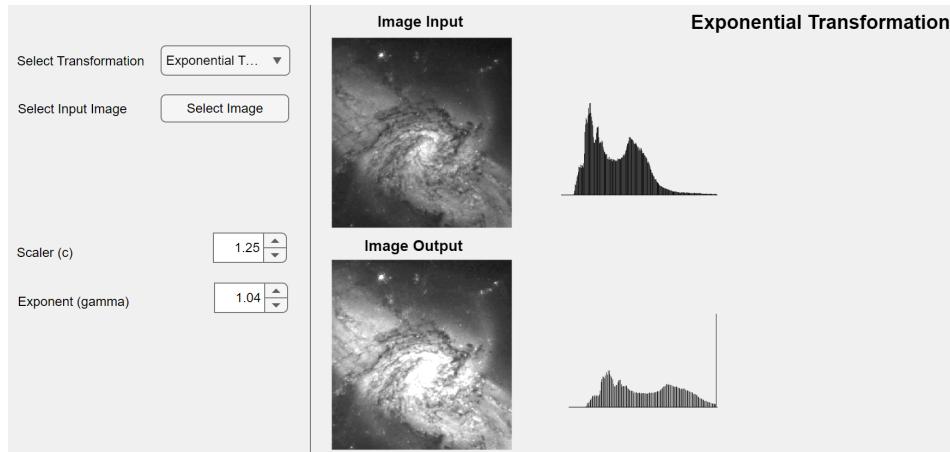
Gambar 2.2.4.4 Transformasi pangkat dari citra pollens_lowcontrast.jpg



Gambar 2.2.4.5 Transformasi pangkat dari citra peppers.bmp



Gambar 2.2.4.6 Transformasi pangkat dari citra pollens_lowcontrast.bmp



Gambar 2.2.4.7 Transformasi pangkat dari citra galaxy_pair_original.jpg

Analisis:

Operasi perpangkatan ini membuat jarak antar piksel menjadi lebih lebar untuk $\text{gamma} > 1$ dan menjadi lebih sempit untuk $\text{gamma} < 1$. Hal ini membuat kontras citra hasil menjadi lebih high-contrast untuk $\text{gamma} > 1$ dan low-contrast untuk $\text{gamma} < 1$. Hasil dari perpangkatan ini akan cenderung melebar seiring meningkatnya nilai awal dari piksel, dan cenderung menyempit apabila nilai semakin kecil.

v. Peregangan Kontras

Berikut fungsi yang digunakan untuk mentransformasikan citra dengan fungsi peregangan kontras.

Tabel 2.6 Kode program transformasi peregangan kontras

```
% stretching.m
% s = 255*(r - rmin)/(rmax-rmin)
function output_image = stretching(image)
    [rows, cols, color_channels] = size(image);
    output_image = zeros(rows, cols, 'uint8');

    % mencari nilai pixel max dan min dari gambar
    rmin = min(image,[],'all');
    rmax = max(image,[],'all');

    % iterasi untuk setiap pixel
    for i = 1:rows
        for j = 1:cols
            for k = 1:color_channels
                % menjalankan operasi s = 255*(r - rmin)/(rmax-rmin)
untuk
                % tiap pixel
                temp = (double(image(i,j,k)) - rmin)*(255/(rmax-rmin));
                % Clipping
```

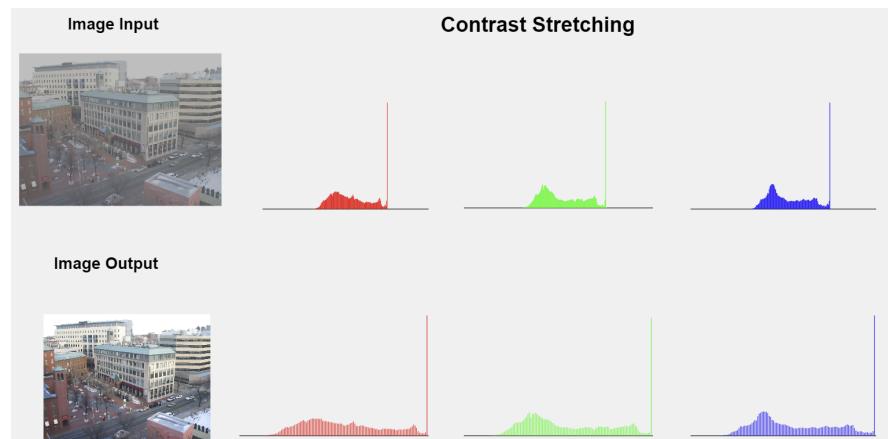
```

        if temp < 0
            output_image(i,j,k) = 0;
        elseif temp > 255
            output_image(i,j,k) = 255;
        else
            output_image(i,j,k) = uint8(temp);
        end
    end
end

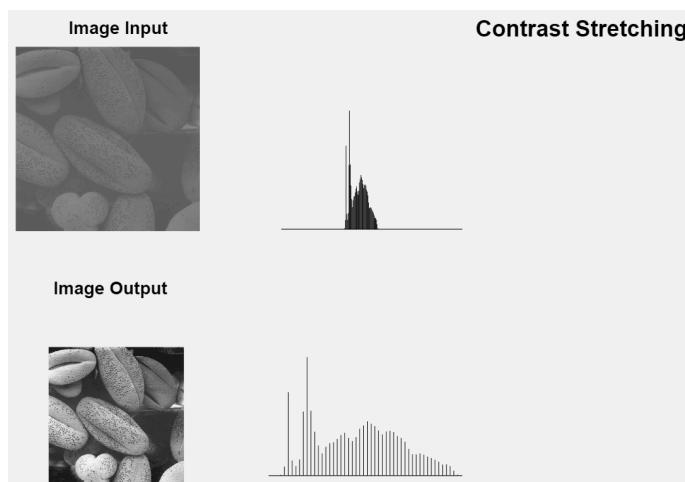
```

Fungsi `stretching` menerima parameter berupa `image` yang berupa citra yang akan diubah. Fungsi ini akan mencari pixel terendah dan tertinggi yang ada pada `image`, kemudian akan meregangkannya dari 0 sampai 255.

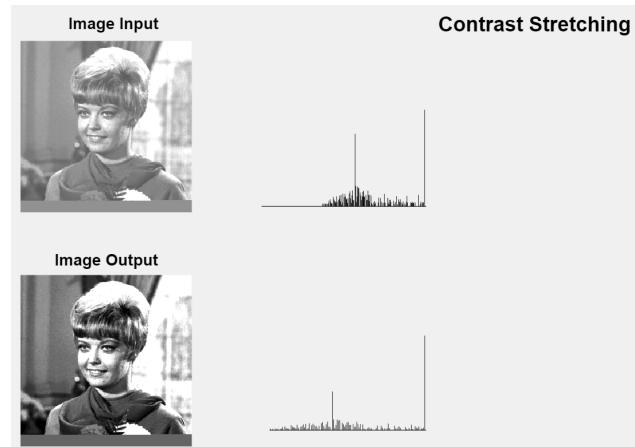
Berikut adalah contoh ketika program dijalankan:



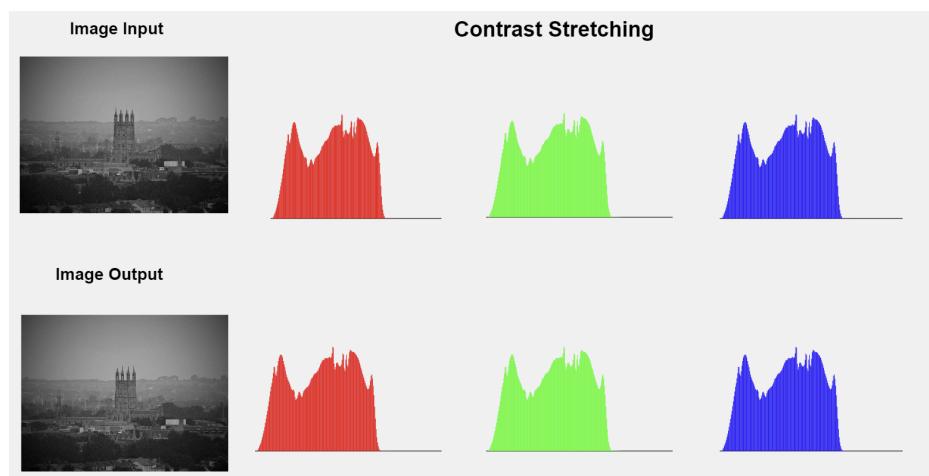
Gambar 2.2.5.1 Peregangan kontras dari citra city_lowcontrast.jpg



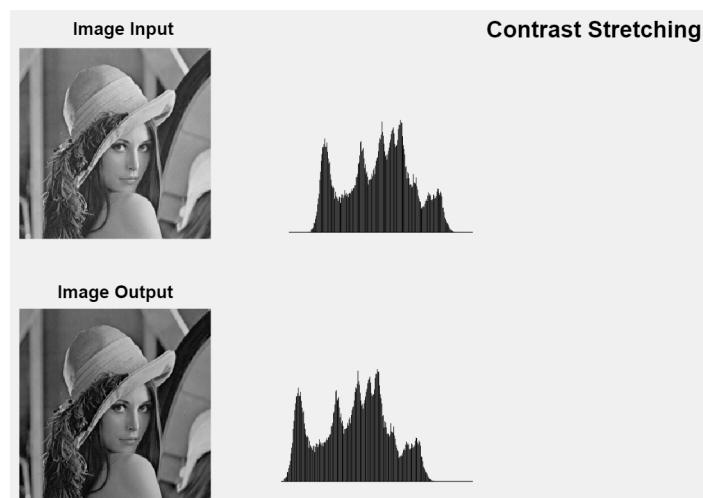
Gambar 2.2.5.2 Peregangan kontras dari citra pollens_lowcontrast.jpg



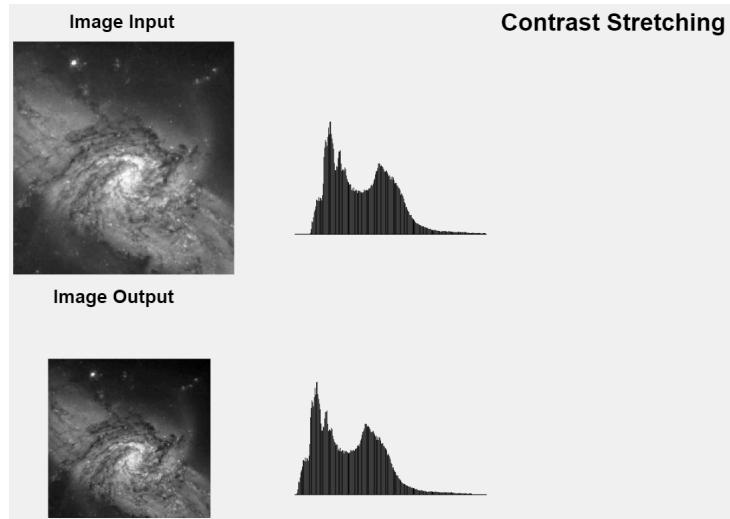
Gambar 2.2.5.3 Peregangan kontras dari citra girl.bmp



Gambar 2.2.5.4 Peregangan kontras dari citra city_during_day.jpg



Gambar 2.2.5.5 Peregangan kontras dari citra city_lowcontrast.jpg



Gambar 2.2.5.6 Peregangan kontras dari citra galaxy_pair_original.jpg

Analisis:

Operasi peregangan kontras dilakukan untuk mengubah citra dengan kontras rendah menjadi lebih tinggi. Seperti yang terlihat pada contoh, citra input adalah citra dengan kontras rendah, yang artinya perbedaan gelap dan terangnya tidak besar/mencolok (cenderung sempit dan mengumpul). Operasi dilakukan dengan transformasi linear, di mana fungsi yang digunakan adalah $(255 - r) / (r_{max} - r_{min})$. r_{max} dan r_{min} yang digunakan adalah 0 dan 255, sehingga fungsi ini akan menyebarluaskan piksel yang tadinya memiliki rentang sempit menjadi rentang 0-255. Hasil dari operasi ini dapat dilihat pada contoh, di mana citra hasilnya memiliki perbedaan gelap dan terang yang lebih mencolok. Dapat dilihat juga histogram hasil yang pikselnya lebih menyebar dibanding dengan citra masukannya.

c. Perataan Histogram

Tabel 2.7 Kode program perataan histogram

```

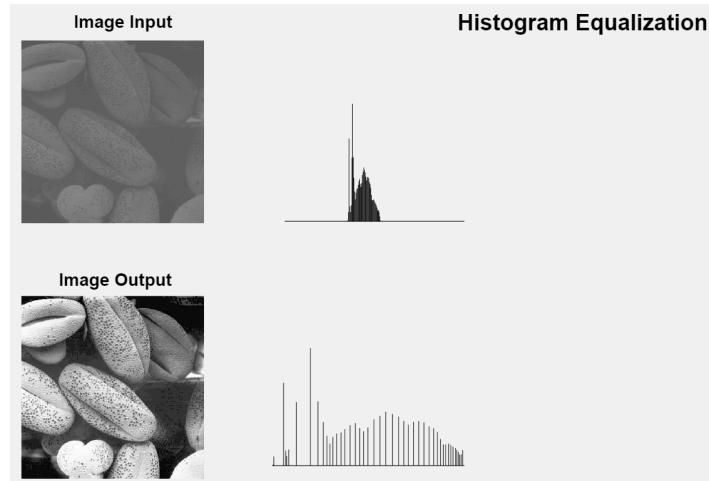
function output_image = histogram_equalization(image)
    equalized_image = compute_equalization(image);
    output_image = equalized_image(double(image) + 1);
end

function hist_eq = compute_equalization(image)
    hist_frequency = compute_histogram(image);

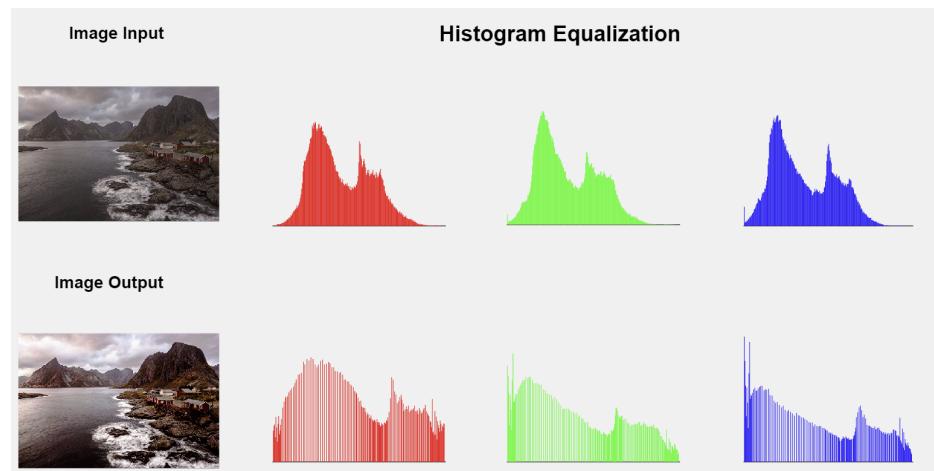
    cdf = cumsum(hist_frequency) / numel(image) * size(image,3);
    hist_eq = uint8(floor(cdf * 255));
end

```

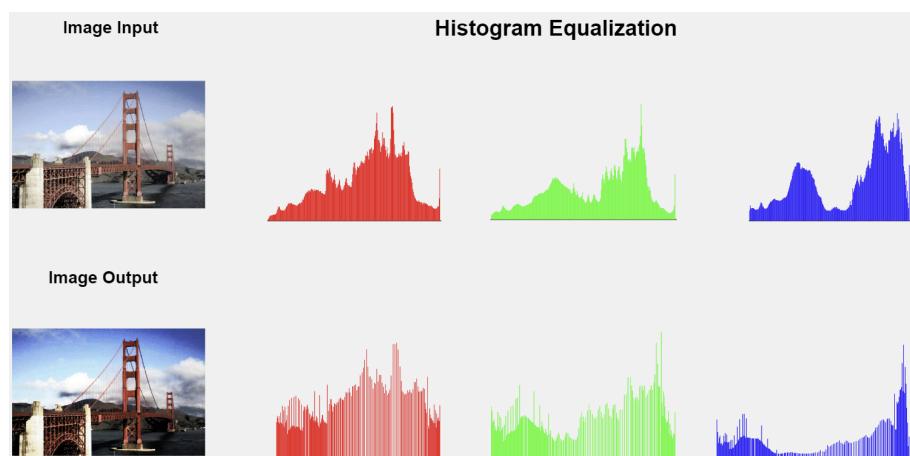
Berikut adalah contoh ketika program dijalankan:



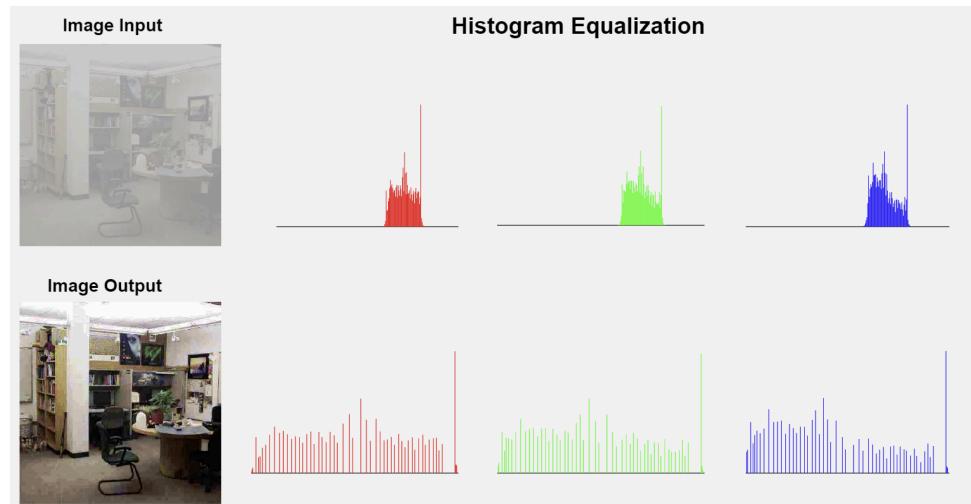
Gambar 2.3.1 Perataan histogram dari citra pollens_lowcontrast.jpg



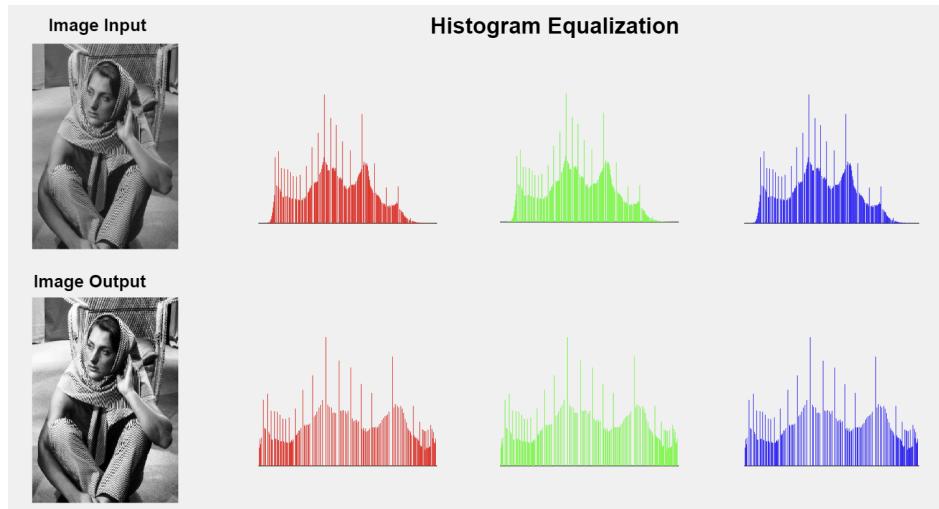
Gambar 2.3.2 Perataan histogram dari citra lofotenisland_lowcontrast.jpg



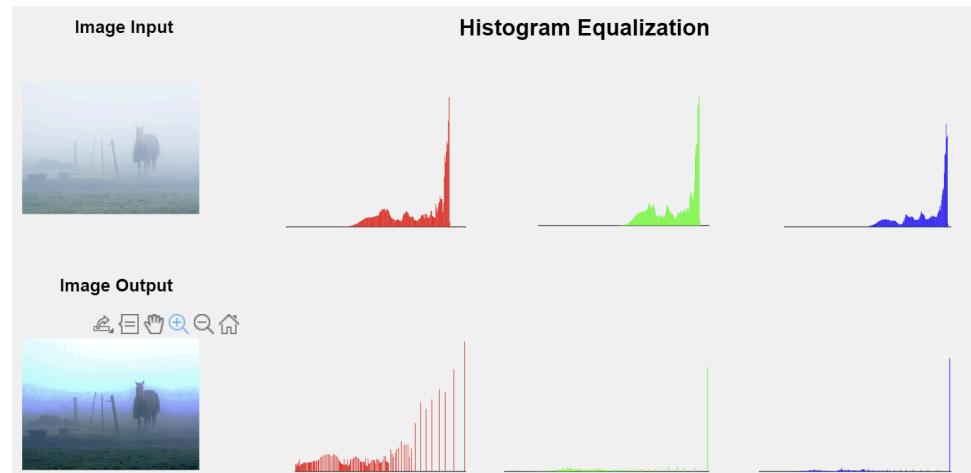
Gambar 2.3.3 Perataan histogram dari citra pollens_lowcontrast.jpg



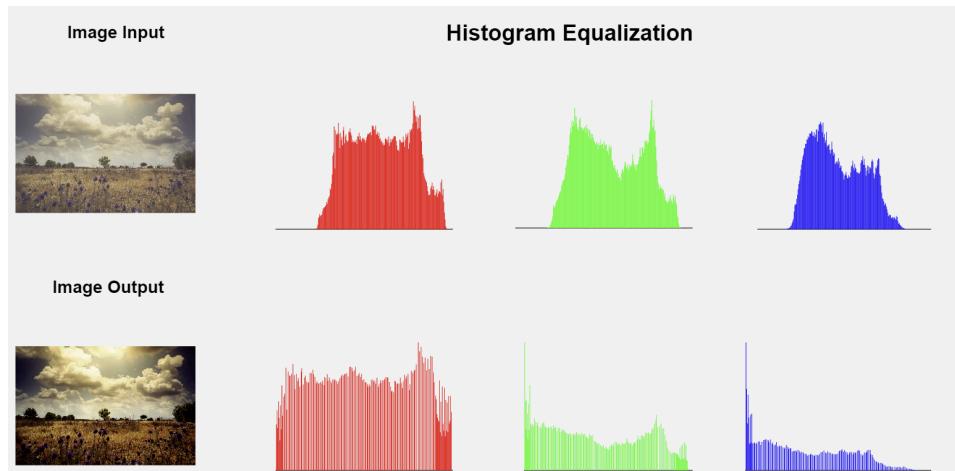
Gambar 2.3.4 Perataan histogram dari citra pollens_lowcontrast.jpg



Gambar 2.3.5 Perataan histogram dari citra barbara.bmp



Gambar 2.3.6 Perataan histogram dari citra horse_in_fog.png



Gambar 2.3.7 Perataan histogram dari citra field.png

Analisis:

Teknik perataan histogram pada citra bertujuan untuk meningkatkan distribusi intensitas piksel, sehingga menghasilkan citra yang memiliki kontras yang lebih baik. Teknik ini meratakan histogram citra, sehingga lebih banyak detail yang dapat terlihat, terutama pada area yang awalnya terlalu terang atau gelap.

Tahap pertama yang dilakukan adalah mengkomputasi histogram citra dengan fungsi `compute_histogram`. Kemudian, iterasi untuk 256 nilai derajat keabuan (sumbu-X histogram), jumlahkan nilai frekuensi kemunculan setiap nilai derajat keabuannya. Setelah itu, bagi dengan banyaknya piksel dalam citra, lalu dikalikan dengan banyaknya kanal warna (1 untuk citra *grayscale* atau 3 untuk citra berwarna). Terakhir, kalikan dengan banyaknya nilai derajat keabuan (255) dengan pembulatan ke bawah (*floor*).

Fungsi `histogram_equalization` sebagai fungsi utama untuk operasi perataan citra memanggil fungsi `compute_histogram`, lalu hasilnya ditambahkan dengan 1 karena struktur data MatLab memulai indeks dari 1, bukan 0.

Dengan menggunakan perataan histogram, citra yang dihasilkan akan memiliki distribusi nilai intensitas yang lebih merata. Pada citra yang awalnya memiliki rentang intensitas terbatas (terlalu gelap atau terlalu terang), perataan histogram dapat meningkatkan kontras di bagian yang cahayanya teredam. Teknik ini sangat berguna pada citra yang membutuhkan peningkatan kontras tanpa menambahkan *noise* atau menghilangkan detail penting. Misalnya pada kasus uji Gambar 2.3.1, citra yang semula memiliki histogram yang sempit di nilai besar diratakan sehingga citra menjadi lebih kontras dan jelas.

Namun, ada batasan yang perlu diperhatikan. Pada beberapa citra, terutama yang sudah memiliki distribusi intensitas yang baik, equalisasi histogram dapat memperkuat *noise* atau menyebabkan efek tidak alami pada citra, sehingga

penggunaannya perlu disesuaikan dengan karakteristik citra tersebut. Misalnya pada Gambar 2.3.5, histogram citra aslinya sudah memiliki persebaran yang cukup merata, sehingga perataan histogram menimbulkan *noise* pada citra.

d. Perbaikan Citra dengan *Histogram Specification (Histogram Matching)*

Tabel 2.8 Kode program perataan histogram

```
% Memeriksa citra grayscale atau berwarna
function I_new = histogram_matching(I,I2)
    if size(I, 3) == 3 && size(I2, 3) == 3
        % Gambar berwarna
        disp('Processing color images...');

        I_red = I(:,:,1); I_green = I(:,:,2); I_blue = I(:,:,3);
        I2_red = I2(:,:,1); I2_green = I2(:,:,2); I2_blue = I2(:,:,3);

        % histogram matching
        I_new_red = matching(I_red, I2_red);
        I_new_green = matching(I_green, I2_green);
        I_new_blue = matching(I_blue, I2_blue);

        % menggabungkan setiap komponen rgb
        I_new = cat(3, I_new_red, I_new_green, I_new_blue);
    else
        % Gambar grayscale
        disp('Processing grayscale images...');

        if size(I, 3) == 3
            I = rgb2gray(I);
        end
        if size(I2, 3) == 3
            I2 = rgb2gray(I2);
        end

        % histogram matching
        I_new = matching(I, I2);
    end
end

function output_image = matching(image1, image2)
    [row, col] = size(image1);
    output_image = zeros(row, col);
    hist1 = compute_equalization(image1); % Gambar input
    hist2 = compute_equalization(image2); % Gambar acuan
    invHist = zeros(256,1);
    for i = 1:256
        [~, minj] = min(abs(hist1(i) - hist2));
        invHist(i) = minj - 1; % Sesuaikan indeks
    end
    for i = 1:row
        for j = 1:col
            output_image(i,j) = invHist(image1(i,j) + 1);
        end
    end
```

```

    end

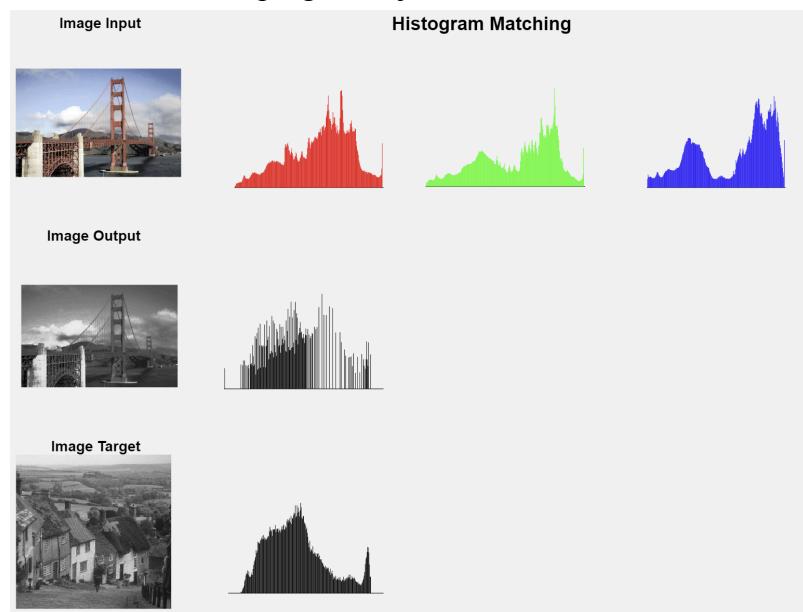
    output_image = uint8(output_image);
end

function hist_eq = compute_equalization(image)
hist_frequency = compute_histogram(image);

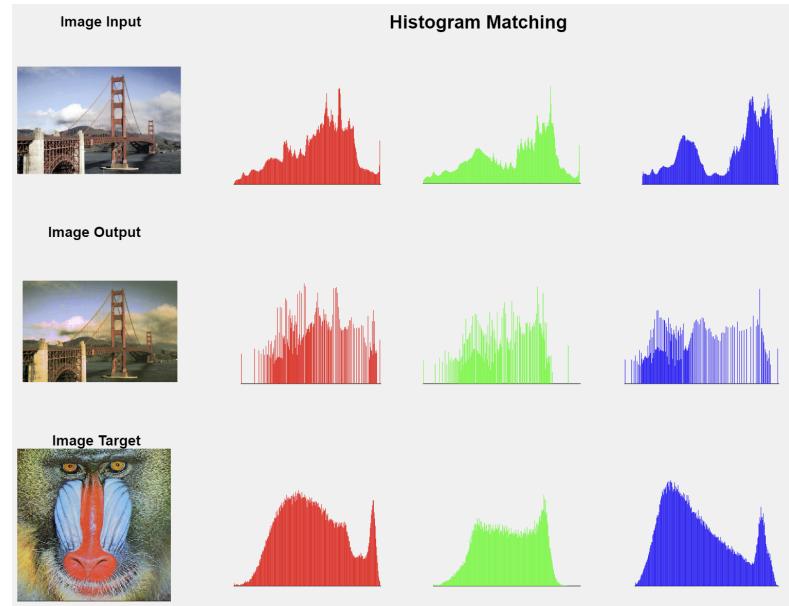
cdf = cumsum(hist_frequency) / numel(image) * size(image,3);
hist_eq = uint8(floor(cdf * 255));
end

```

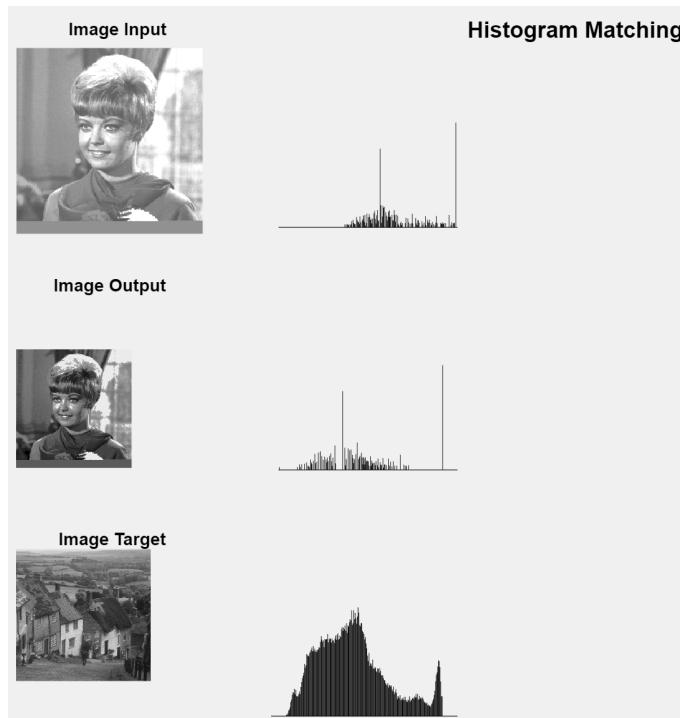
Berikut adalah contoh ketika program dijalankan:



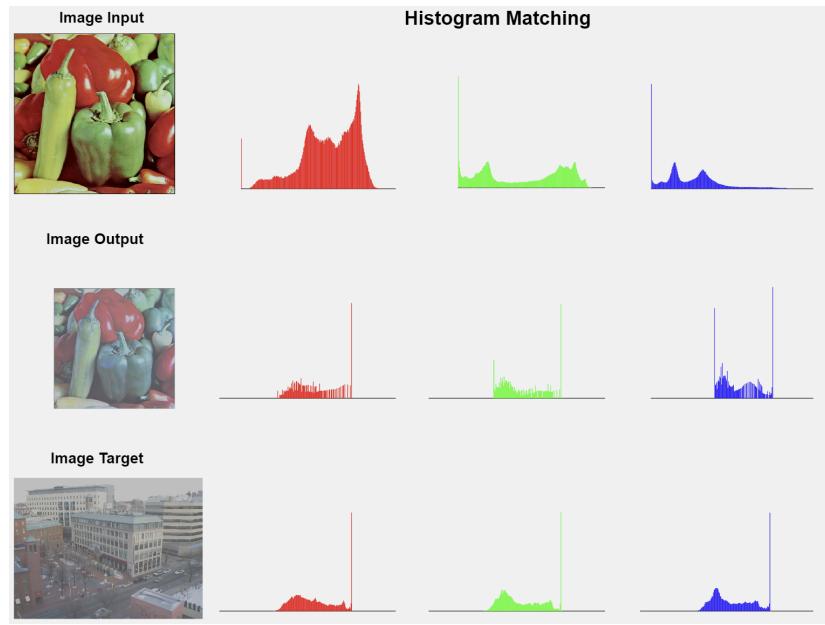
Gambar 2.4.1 Histogram matching citra bridge.png terhadap citra goldhill.bmp



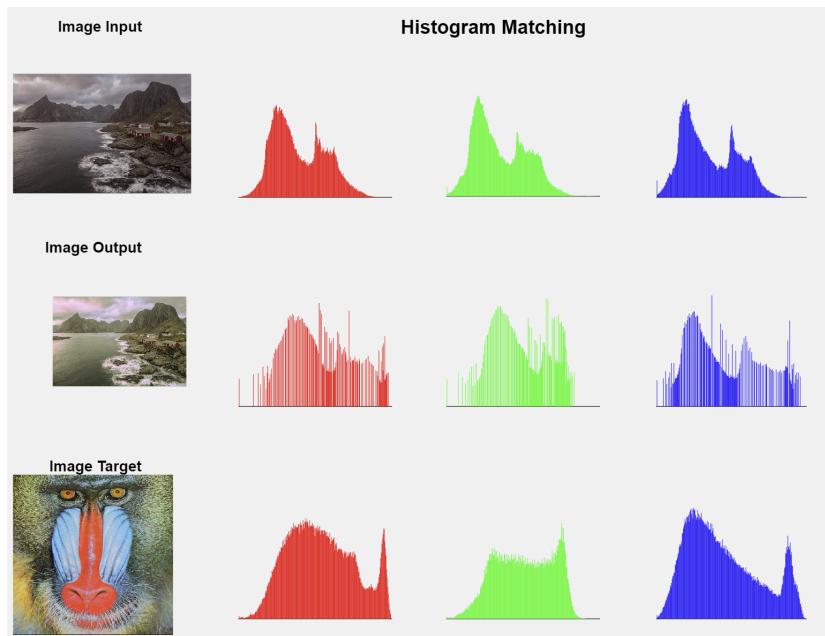
Gambar 2.4.2 *Histogram matching* citra bridge.png terhadap citra baboon24.bmp



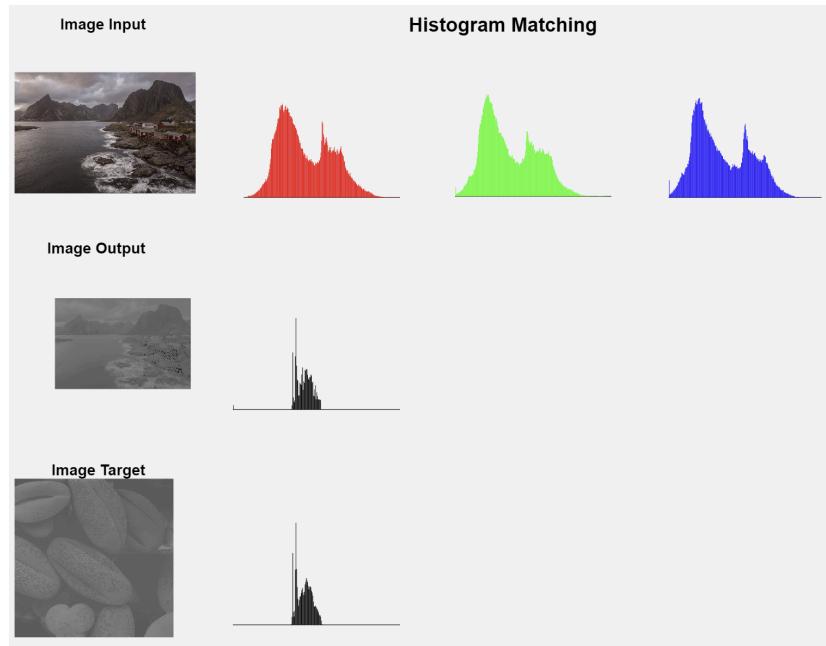
Gambar 2.4.3 *Histogram matching* citra girl.bmp terhadap citra goldhill.bmp



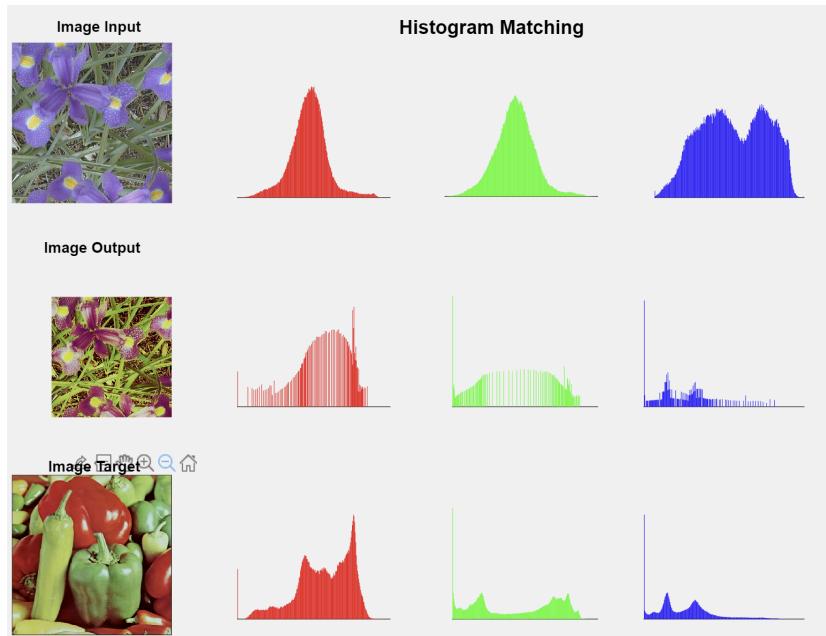
Gambar 2.4.4 *Histogram matching* citra peppers.bmp terhadap citra city_lowcontrast.jpg



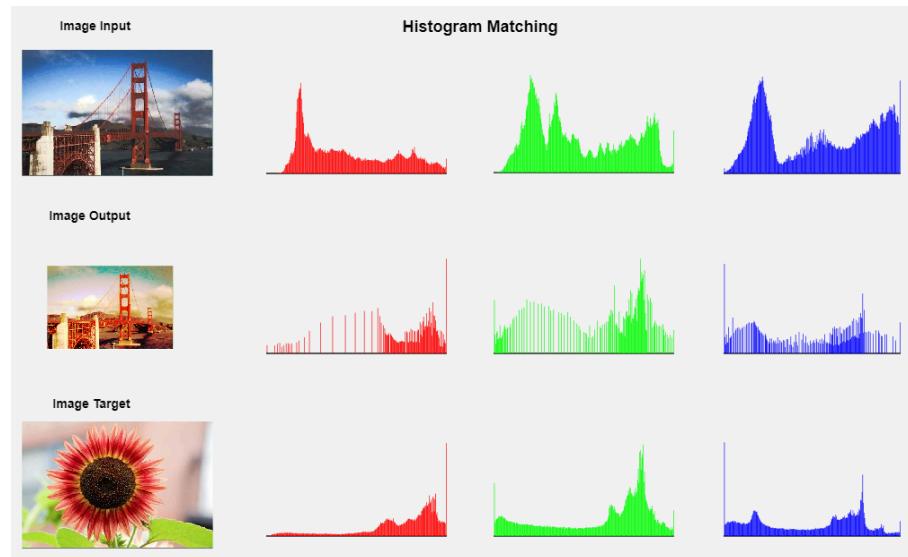
Gambar 2.4.5 *Histogram matching* citra lofotenisland_lowcontrast.jpg terhadap citra baboon24.bmp



Gambar 2.4.6 *Histogram matching* citra lofotenisland_lowcontrast.jpg terhadap citra pollens_lowcontrast.jpg



Gambar 2.4.7 *Histogram matching* citra flower.jpg terhadap citra peppers.bmp



Gambar 2.4.8 *Histogram matching* citra bridge.bmp terhadap citra sunflower.bmp

Analisis:

Operasi *histogram matching* ini berusaha untuk mengubah histogram dari citra input semirip mungkin dengan histogram dari citra acuan. Bisa dilihat bahwa citra hasil operasi ini memiliki *tone* warna yang mirip dengan citra acuan. Histogram dari citra hasil juga mirip dengan histogram dari target. Caranya adalah dengan melakukan perataan histogram pada citra input dan citra acuan. Kemudian pada hasil perataan, dilakukan transformasi dengan fungsi balikan, yang menyesuaikan nilai-nilai piksel pada citra input dengan citra acuan terdekat. Fungsi ini diiterasi untuk setiap piksel yang ada di citra input.

3. Alamat GitHub

https://github.com/arleenchr/IF4073_Tugas1