

Laporan Tugas 1 IF4073 Interpretasi dan Pengolahan Citra
Semester I Tahun 2023/2024



Disusun Oleh:

Fachry Dennis Heraldi 13520139

M Syahrul Surya Putra 13520161

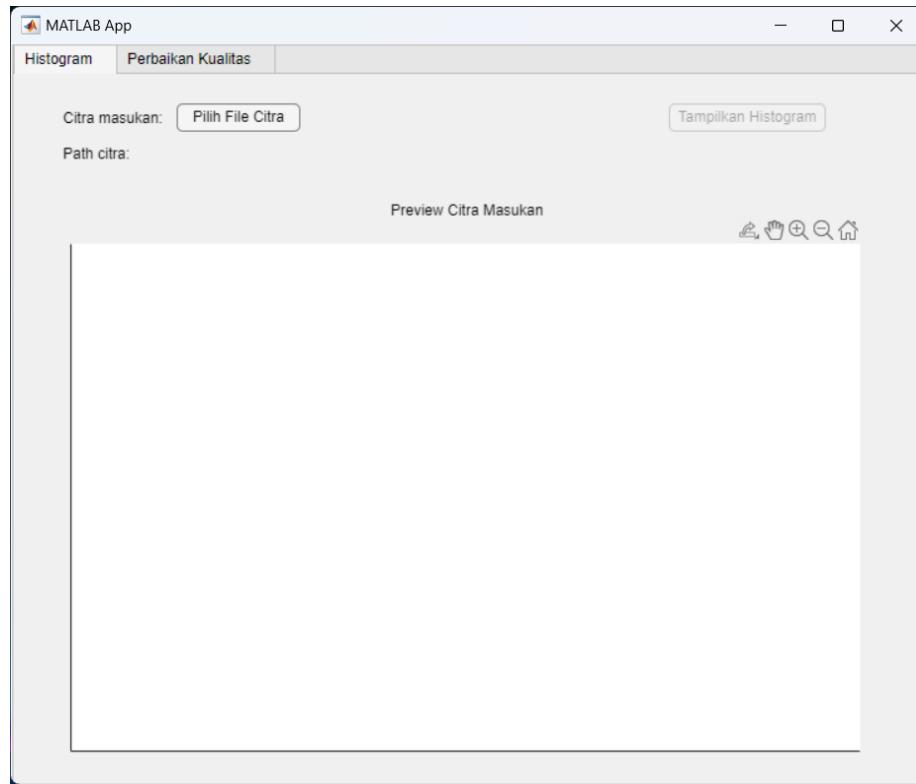
PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2023

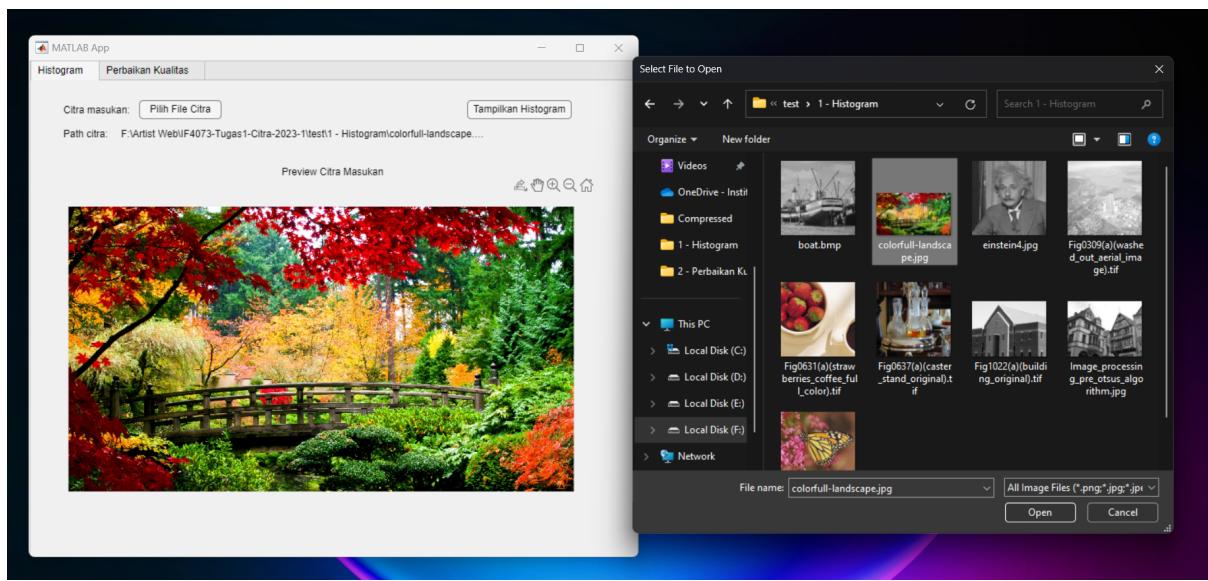
A. Screenshot GUI Program

1. Program Menghitung dan Menampilkan Histogram Citra

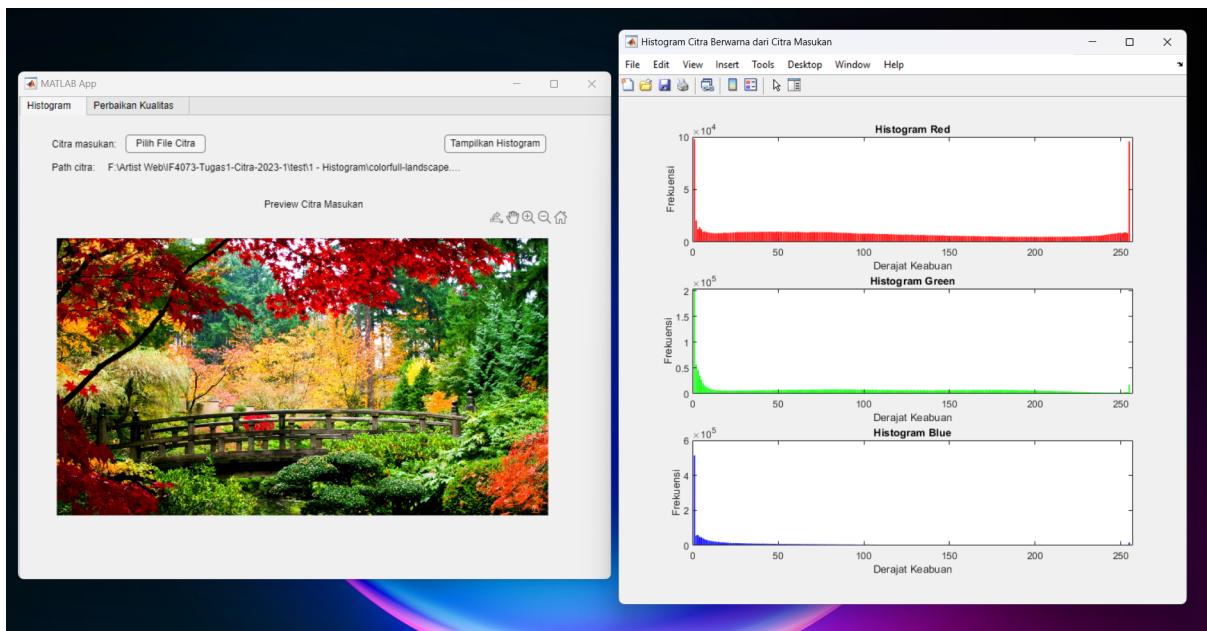
- Tampilan awal ketika program untuk menampilkan histogram dijalankan



- Tampilan ketika memilih citra masukan

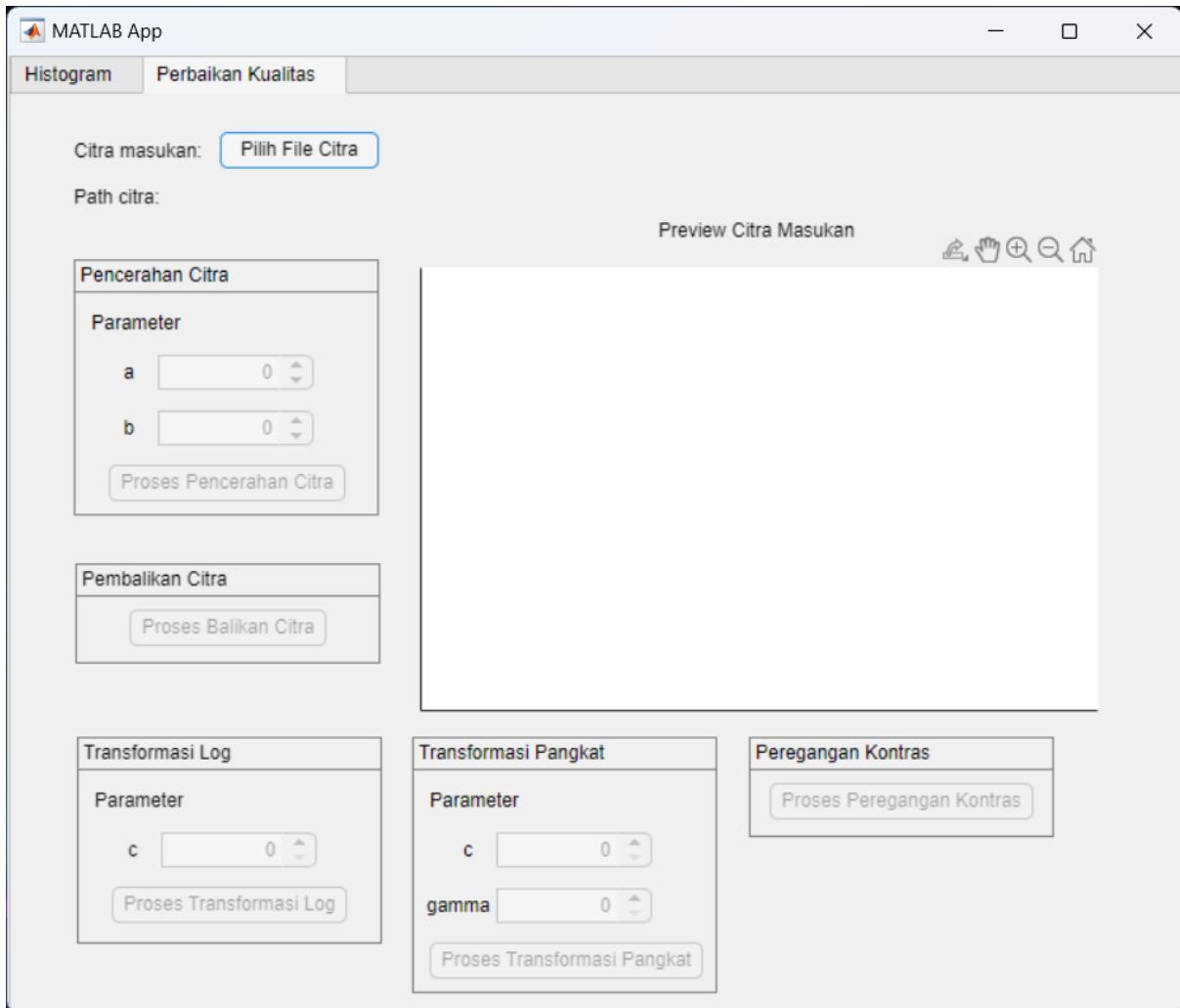


- Tampilan ketika tombol 'Tampilkan Histogram' ditekan

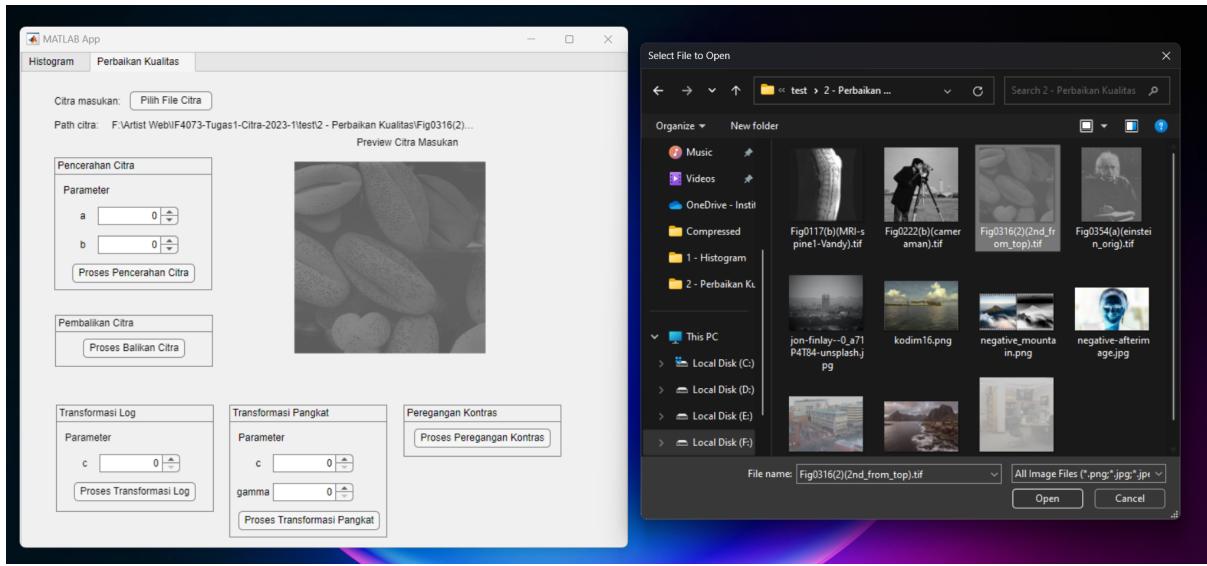


2. Program Perbaikan Kualitas Citra

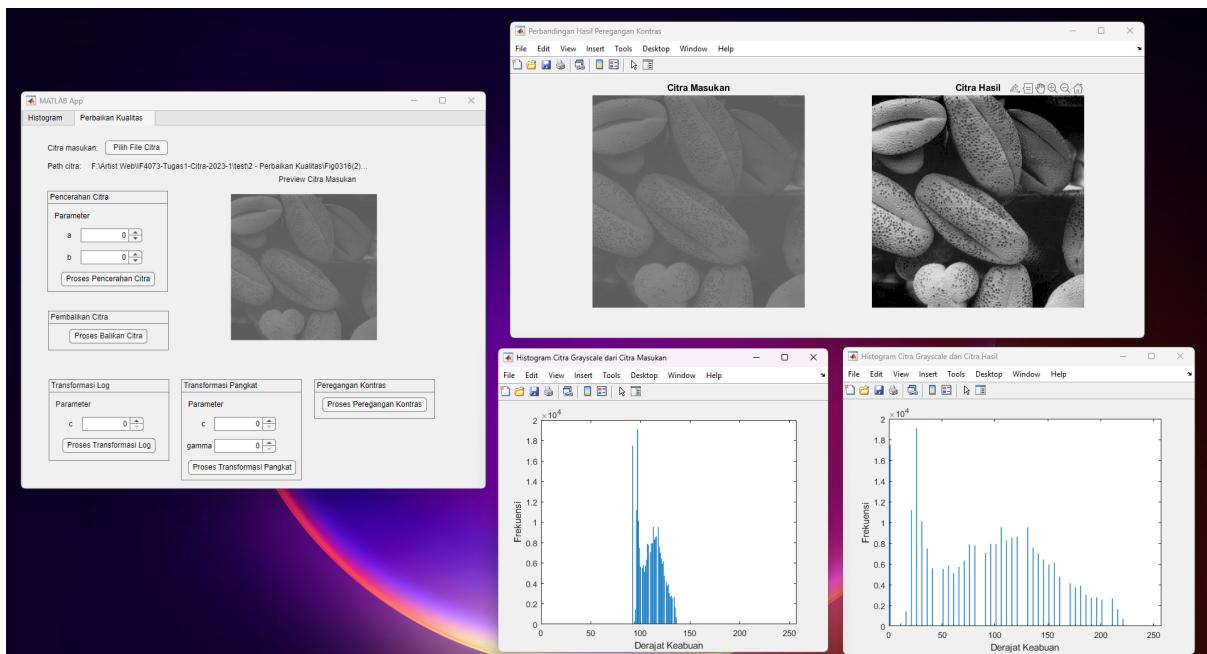
a. Tampilan awal ketika program dijalankan



b. Tampilan ketika memilih citra masukan

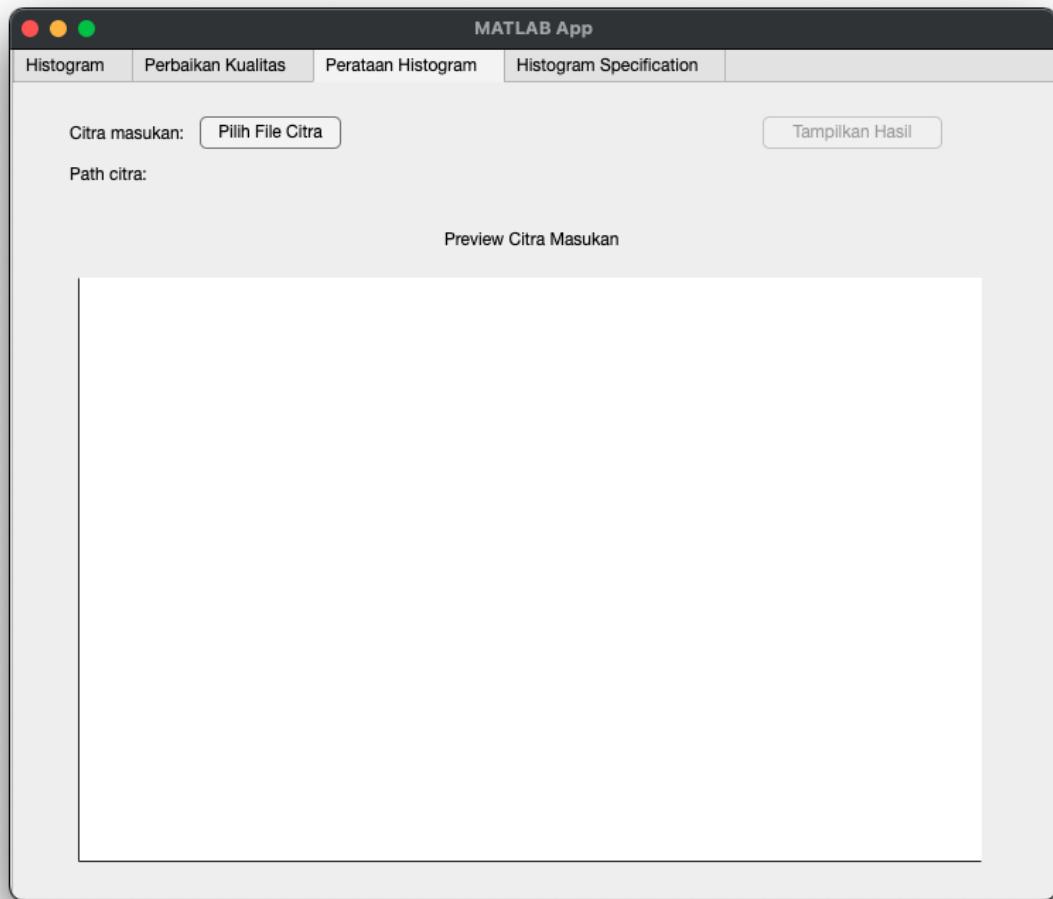


c. Tampilan ketika menekan tombol 'Proses Peregangan Kontras'

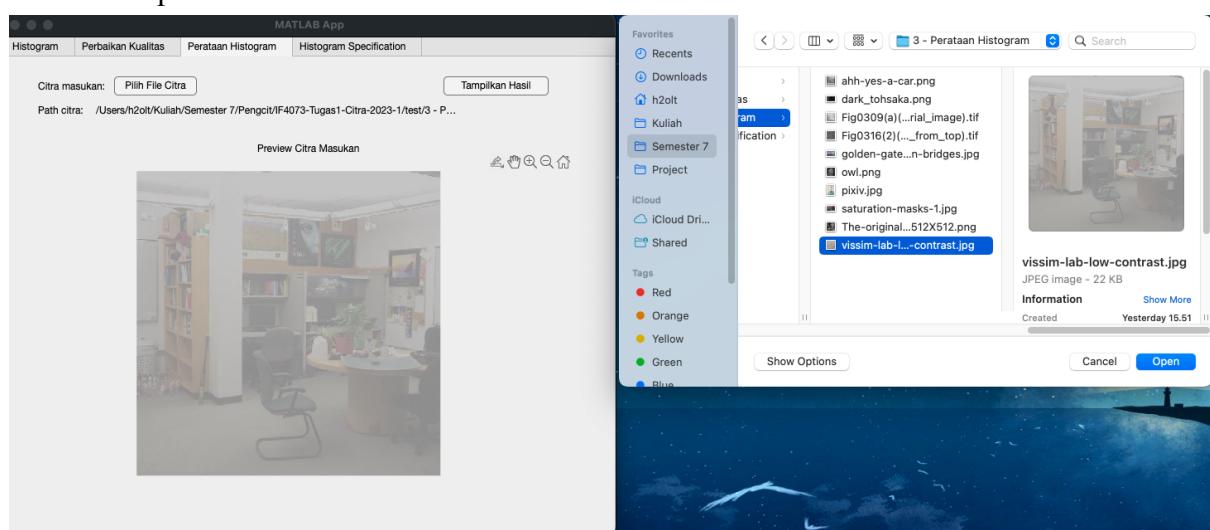


3. Program Perataan Histogram

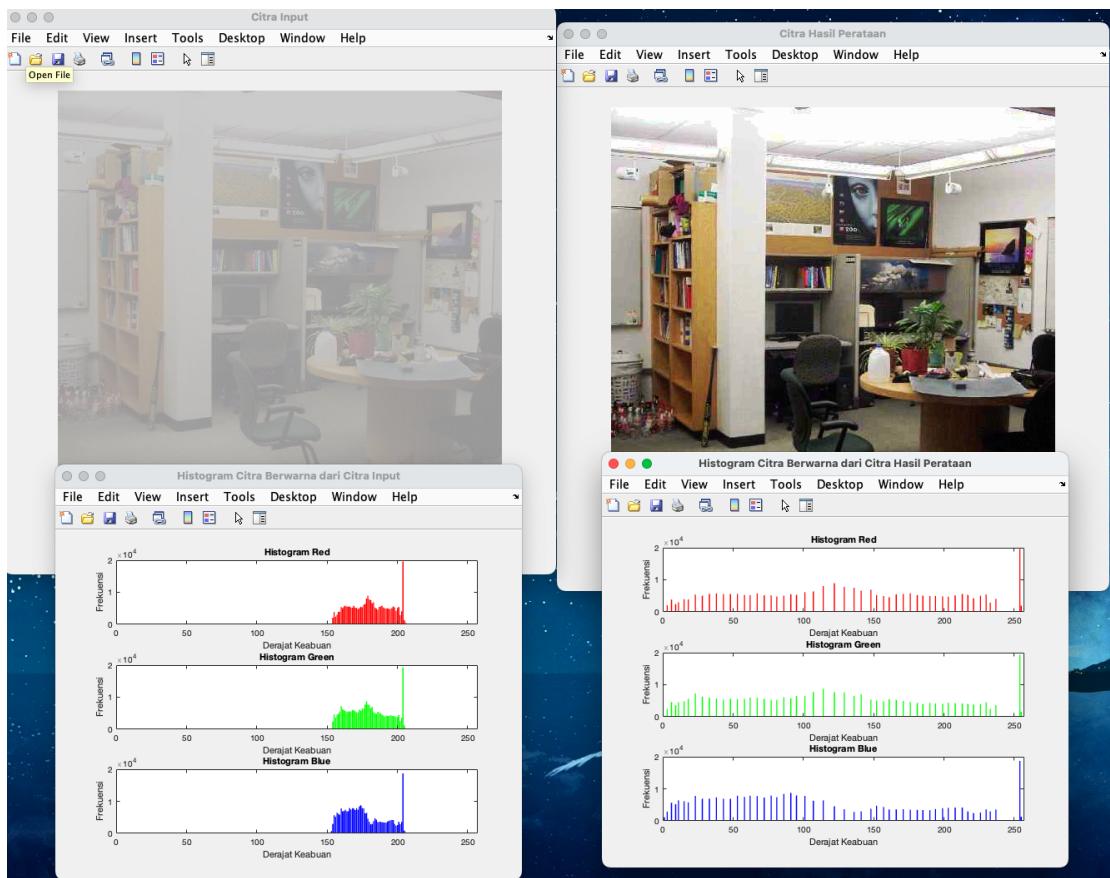
- Tampilan awal ketika program dijalankan



- Tampilan ketika memilih citra masukan

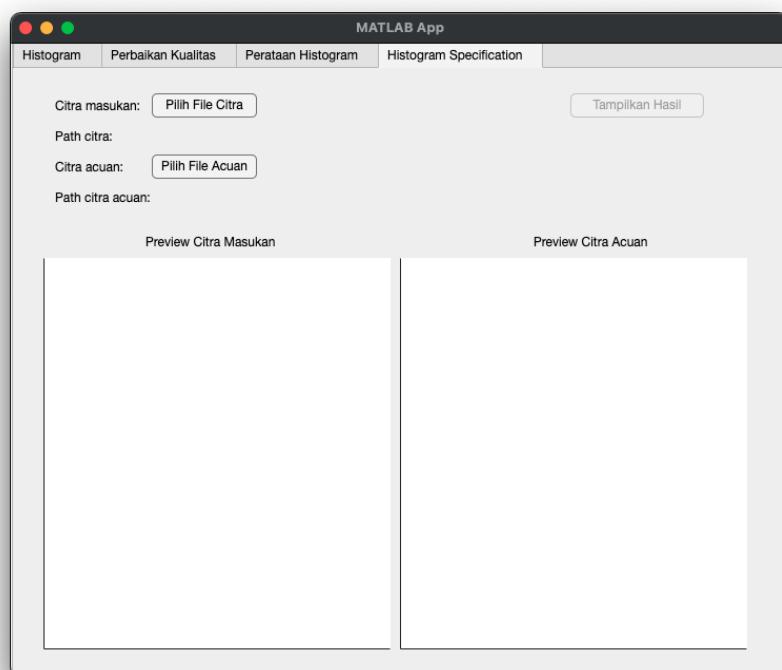


c. Tampilan ketika menekan tombol 'Tampilkan Hasil'

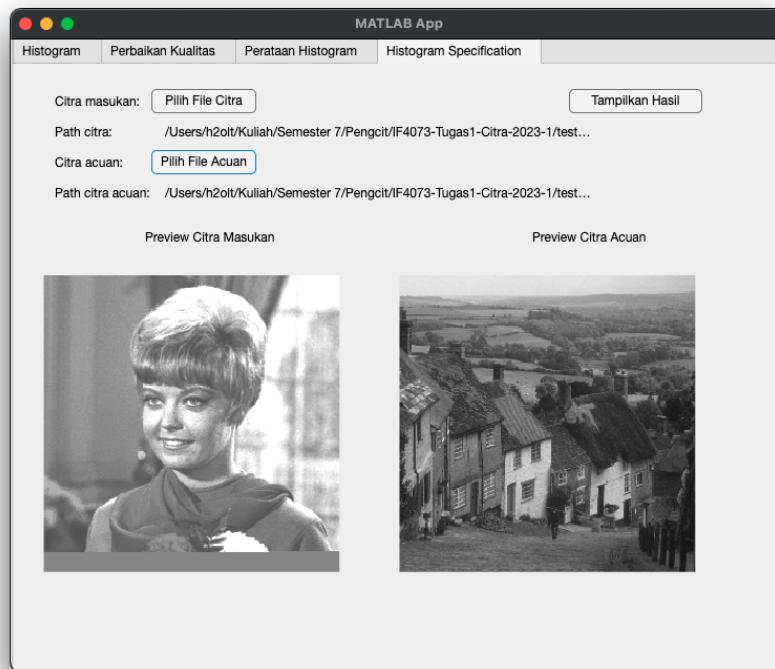


4. Program Perbaikan Citra dengan *Histogram Matching*

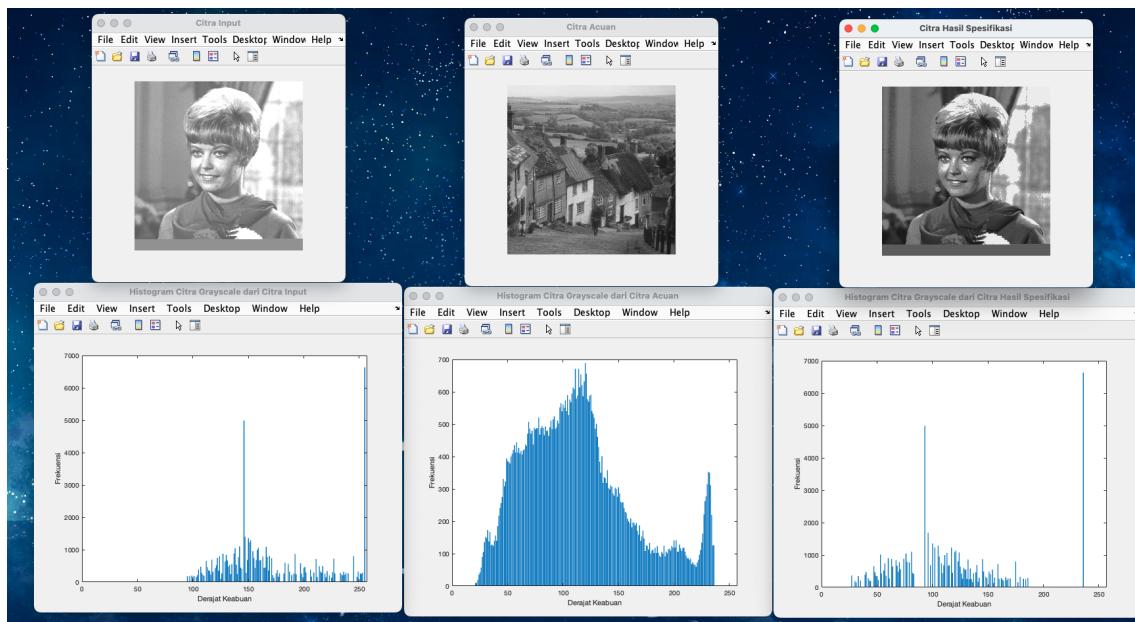
a. Tampilan awal ketika program dijalankan



b. Tampilan ketika memilih citra masukan



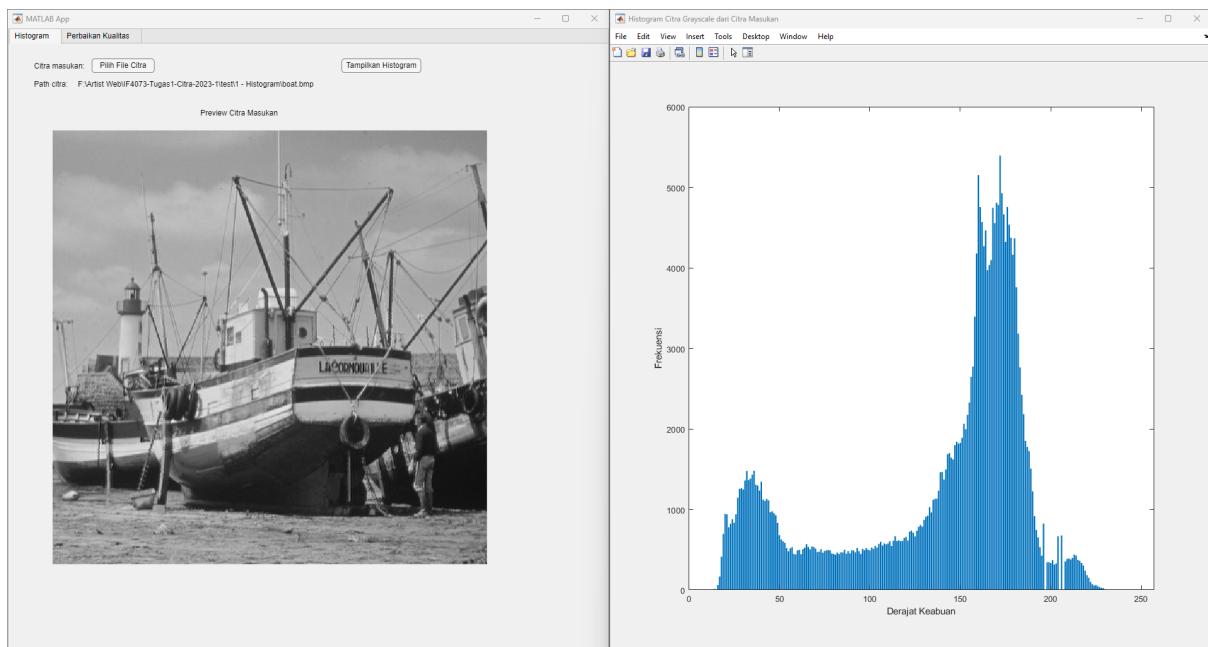
c. Tampilan ketika menekan tombol 'Tampilkan Hasil'



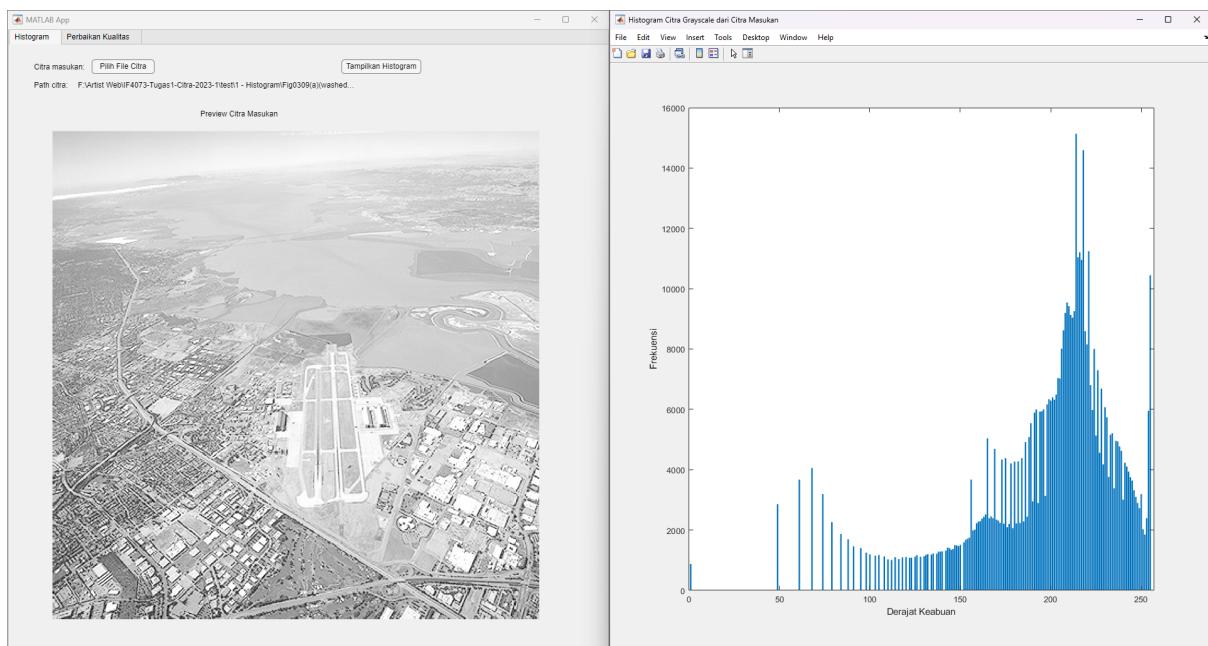
B. Contoh Hasil Eksekusi Program dengan Contoh-contoh Citra Input

1. Program Menghitung dan Menampilkan Histogram Citra

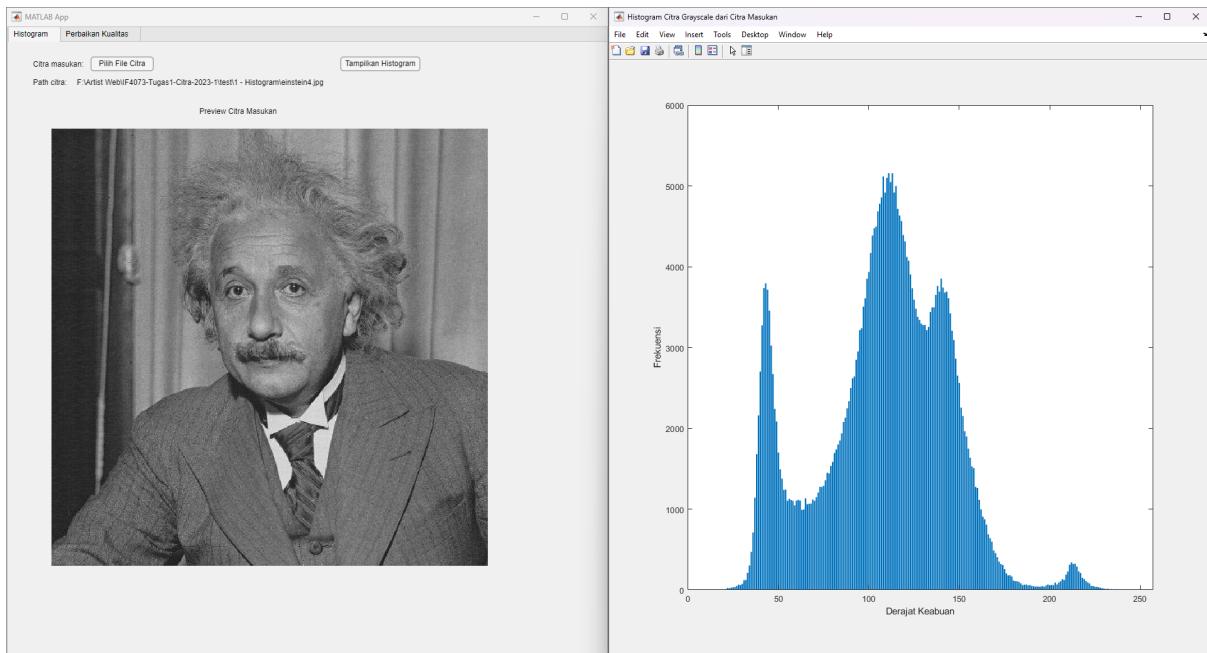
a. Hasil Eksekusi Citra 1 dari Contoh



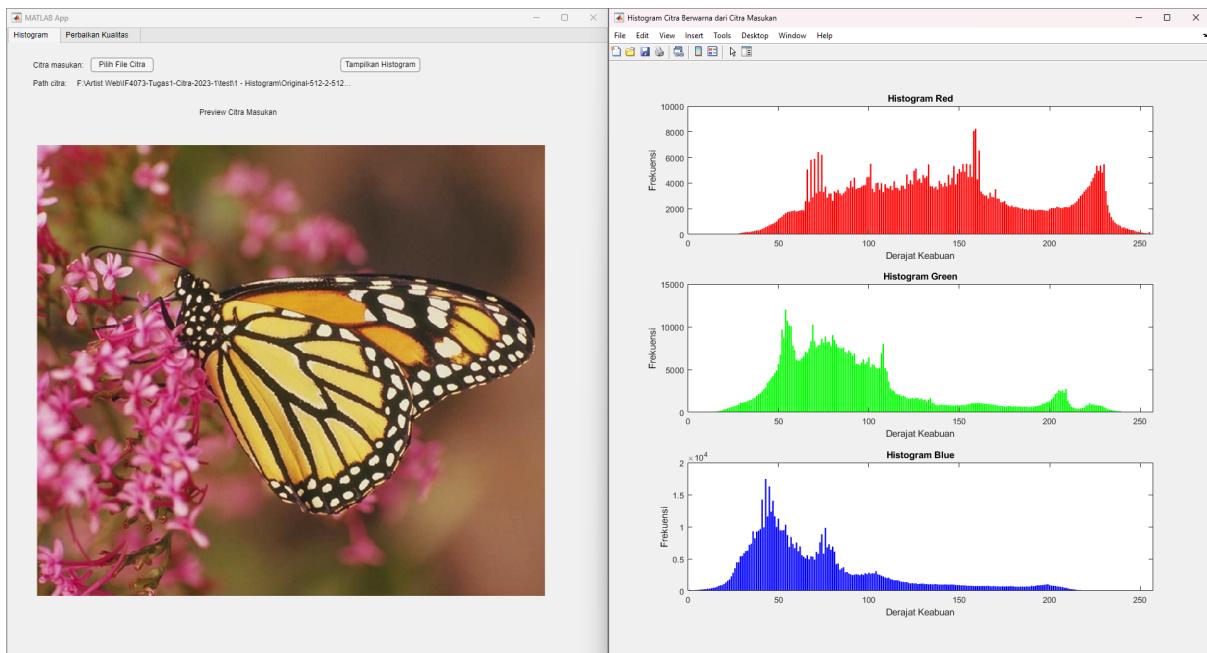
b. Hasil Eksekusi Citra 2 dari Contoh



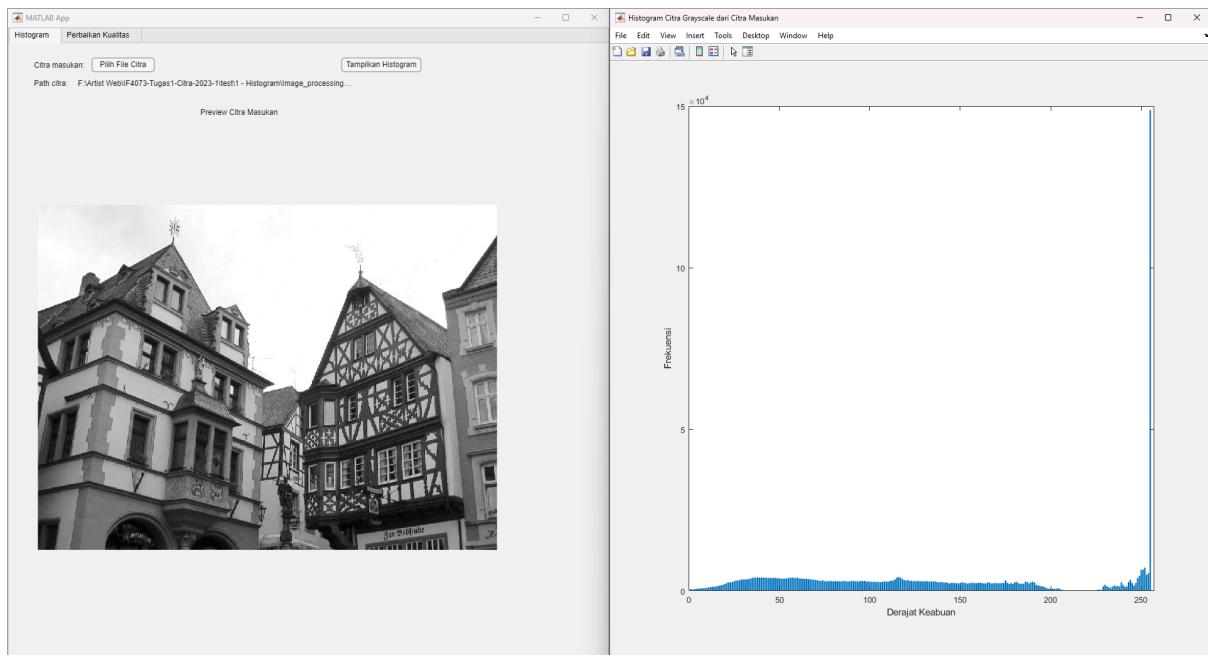
c. Hasil Eksekusi Citra 3 dari Contoh



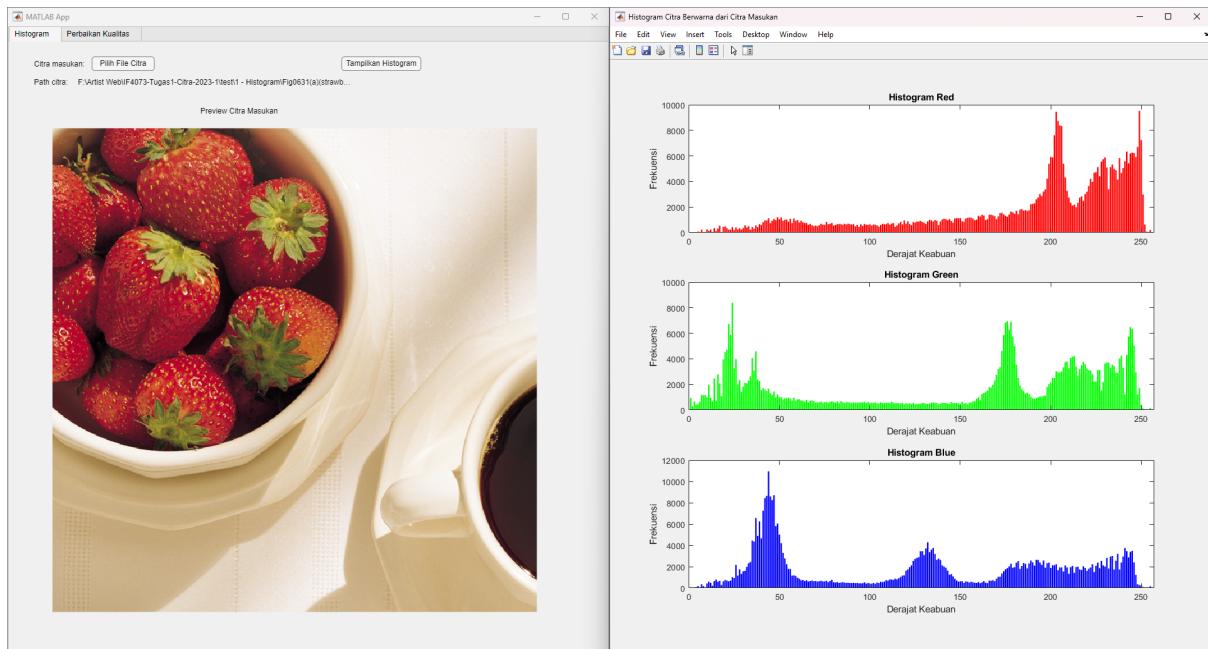
d. Hasil Eksekusi Citra 4 dari Contoh



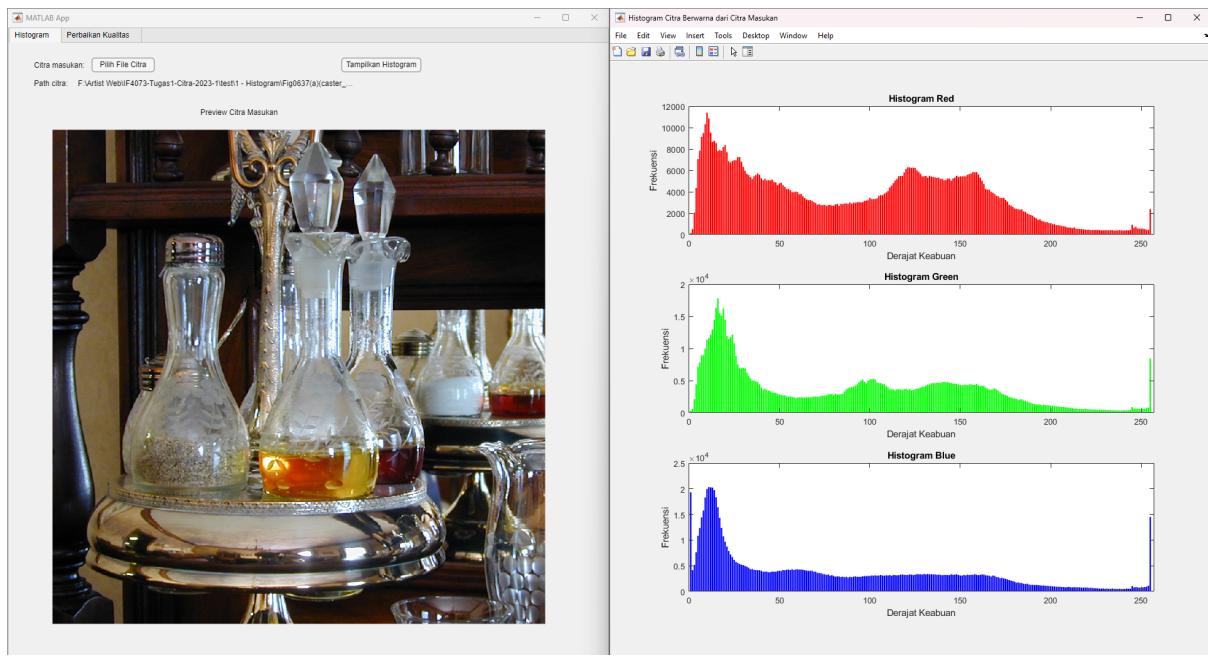
e. Hasil Eksekusi Citra 5 dari Contoh



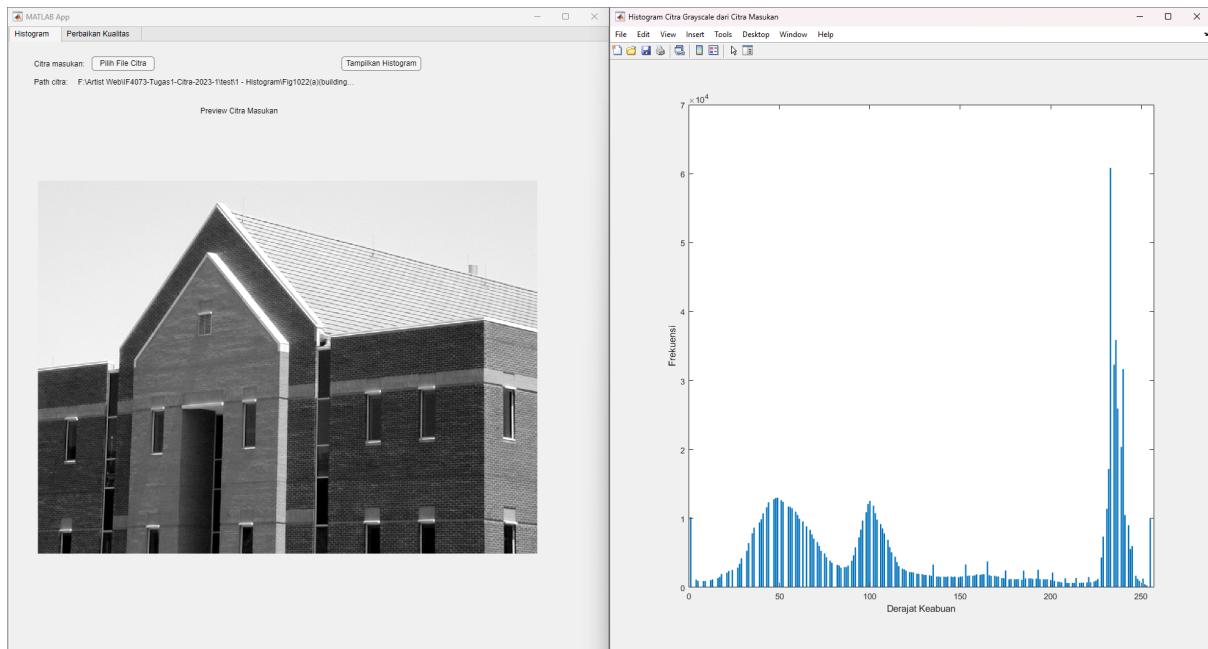
f. Hasil Eksekusi Citra 6 dari Contoh



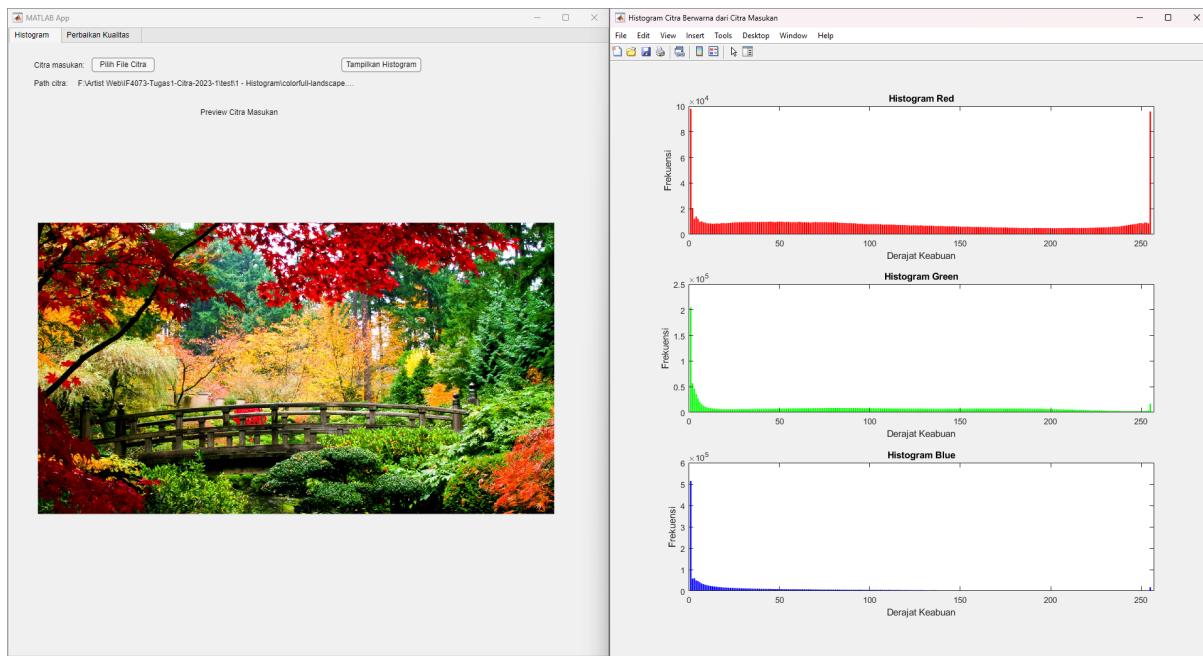
g. Hasil Eksekusi Citra 7 dari Contoh



h. Hasil Eksekusi Citra Grayscale Tambahan



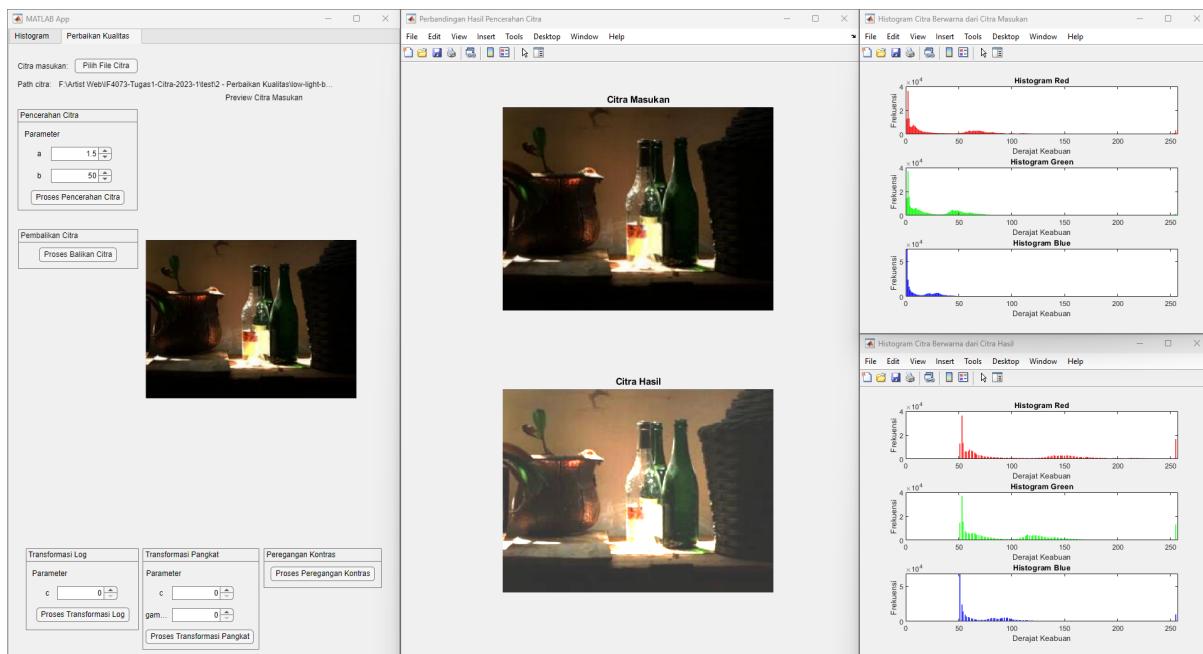
i. Hasil Eksekusi Citra Berwarna Tambahan



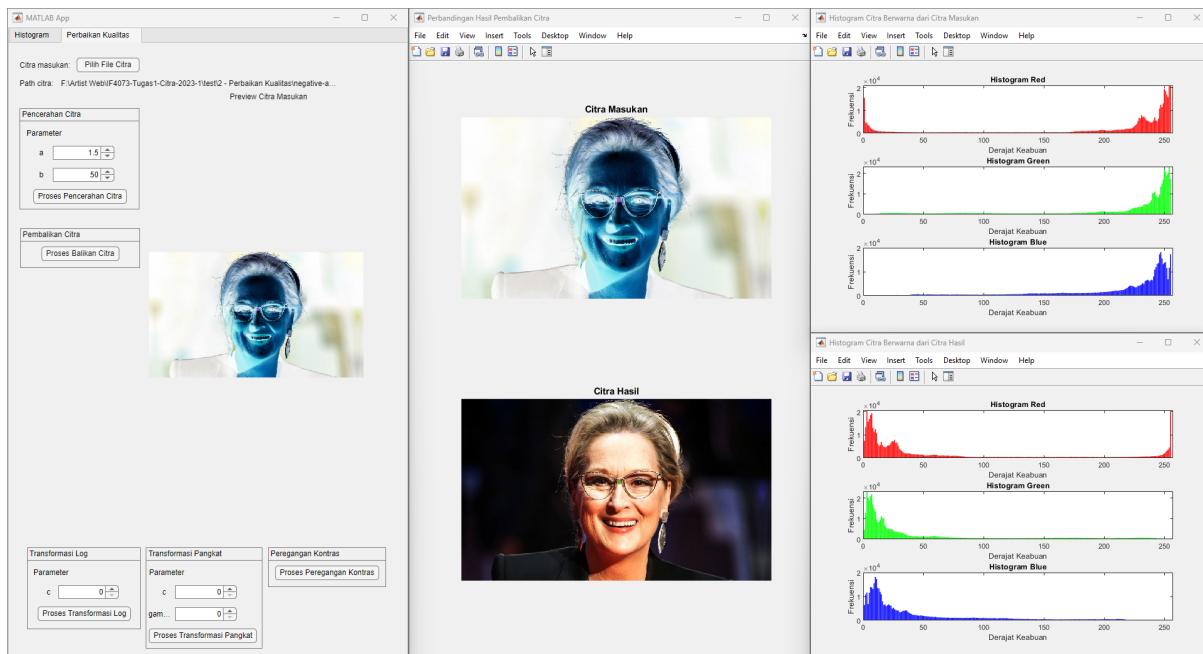
2. Program Perbaikan Kualitas Citra

a. Hasil Eksekusi Pencerahan Citra

Pencerahan pada citra gelap dengan parameter $a = 1.5$ dan $b = 50$

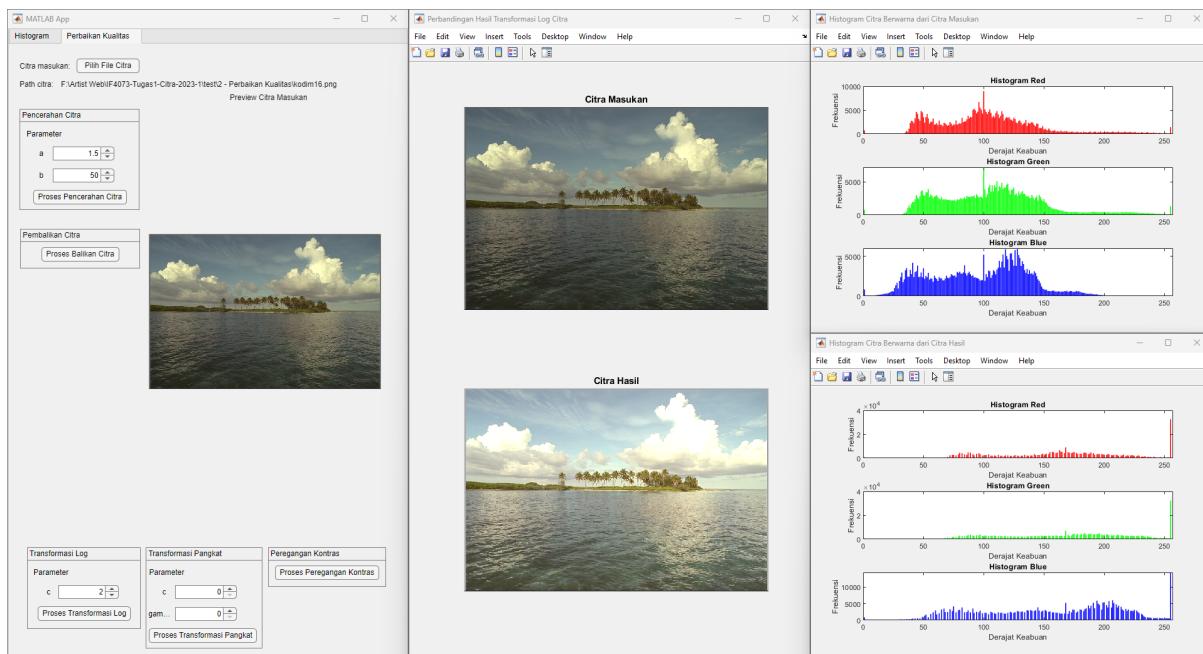


b. Hasil Eksekusi Pembalikan Citra



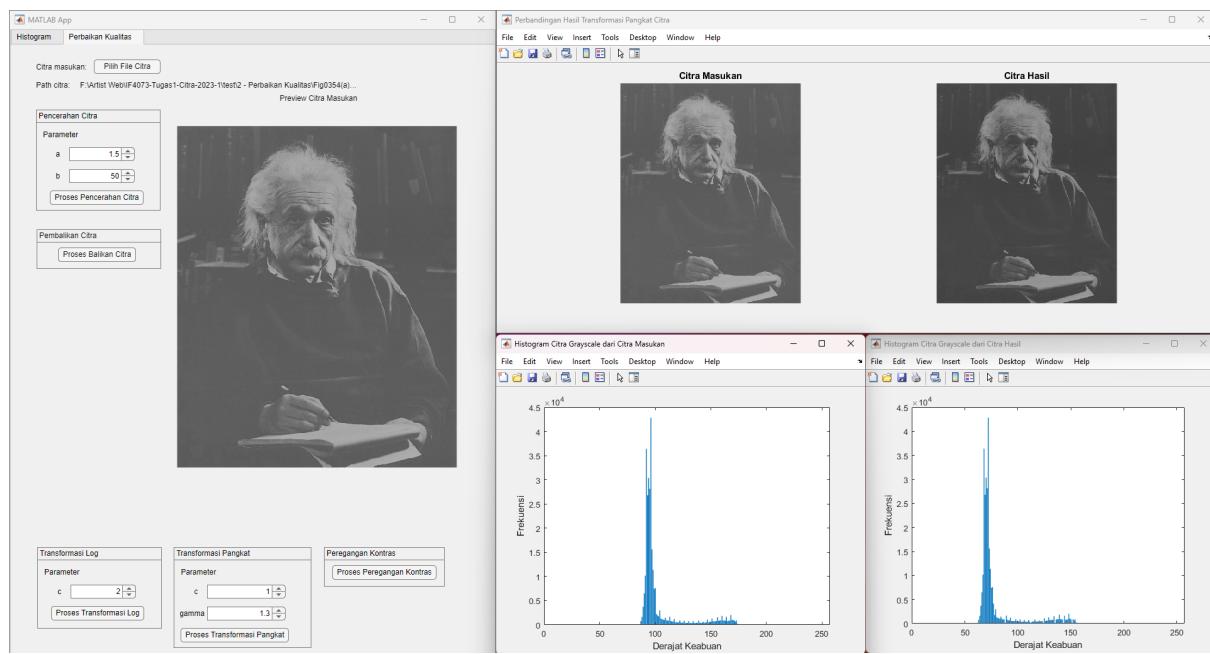
c. Hasil Eksekusi Transformasi Log pada Citra

Transformasi log pada citra dengan parameter $c = 2$

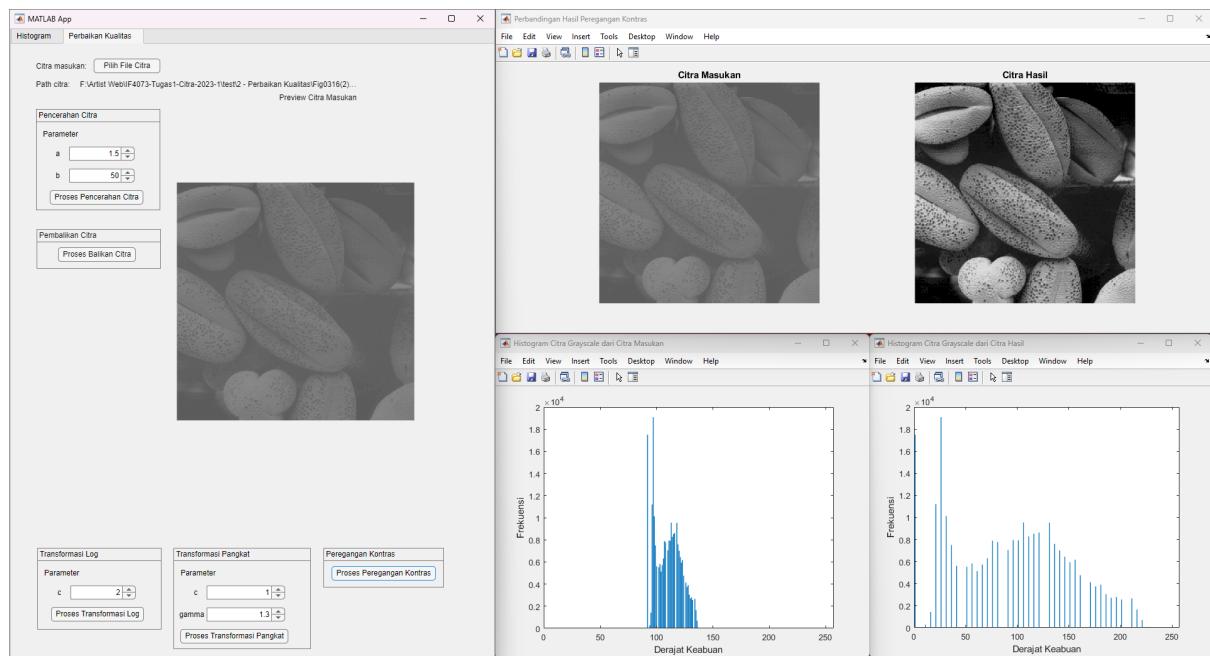


d. Hasil Eksekusi Transformasi Pangkat pada Citra

Transformasi pangkat pada citra dengan parameter $c = 1$ dan $\text{gamma} = 1.3$

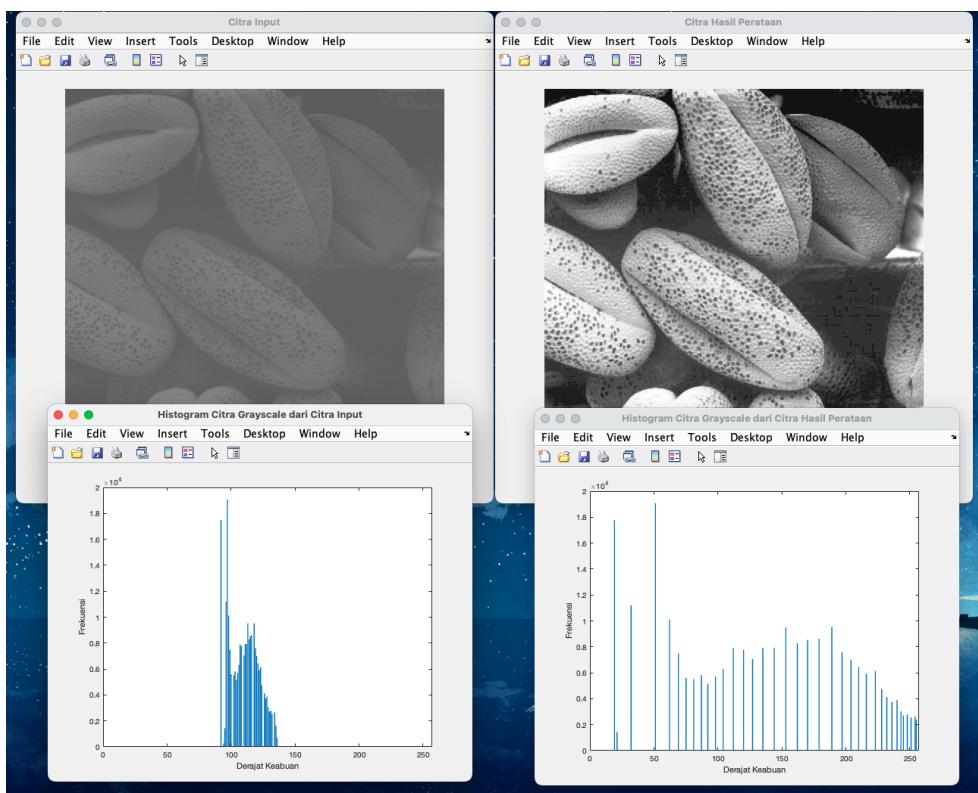


e. Hasil Eksekusi Peregangan Kontras pada Citra

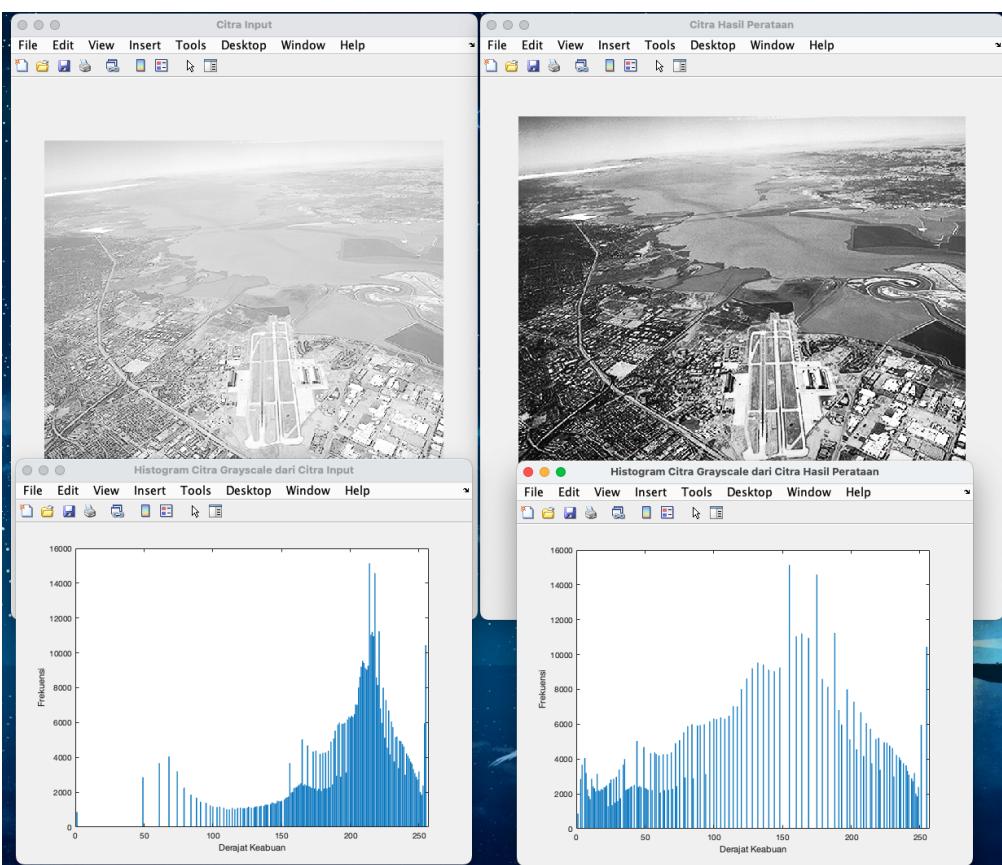


3. Program Perataan Histogram

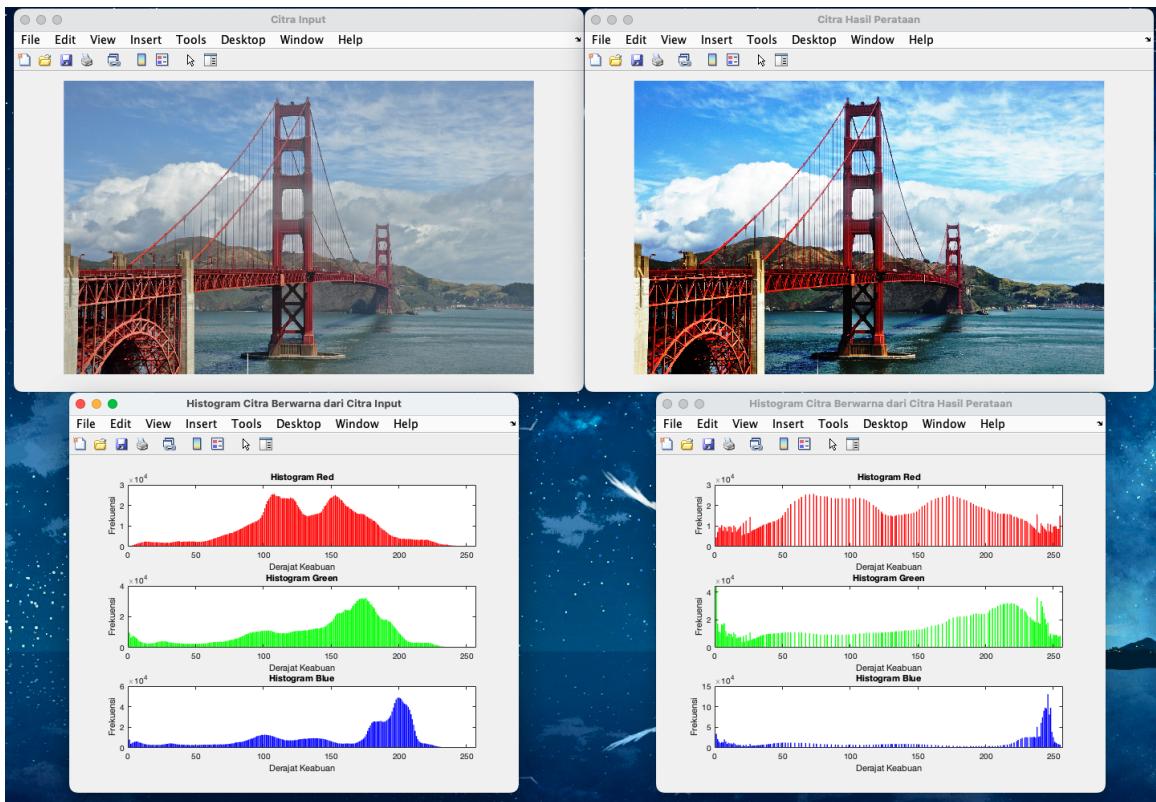
a. Hasil Eksekusi Citra 1 dari Contoh



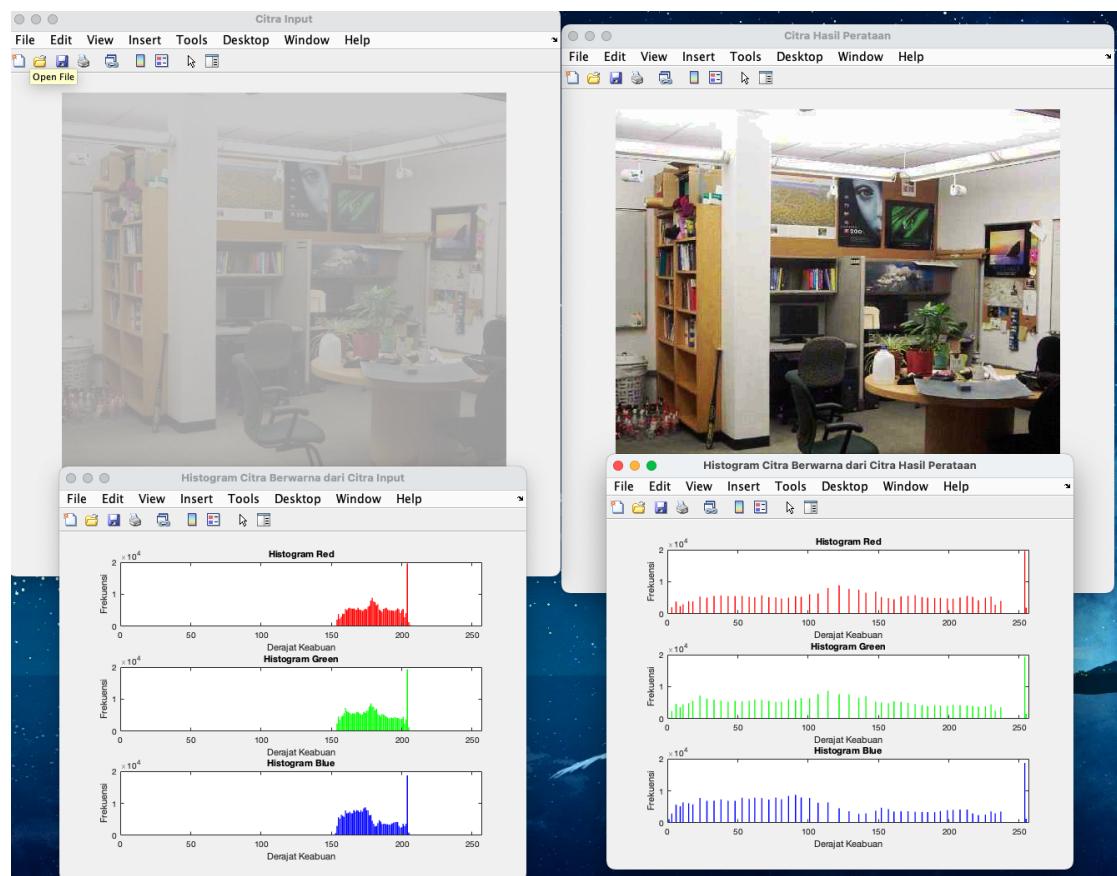
b. Hasil Eksekusi Citra 2 dari Contoh



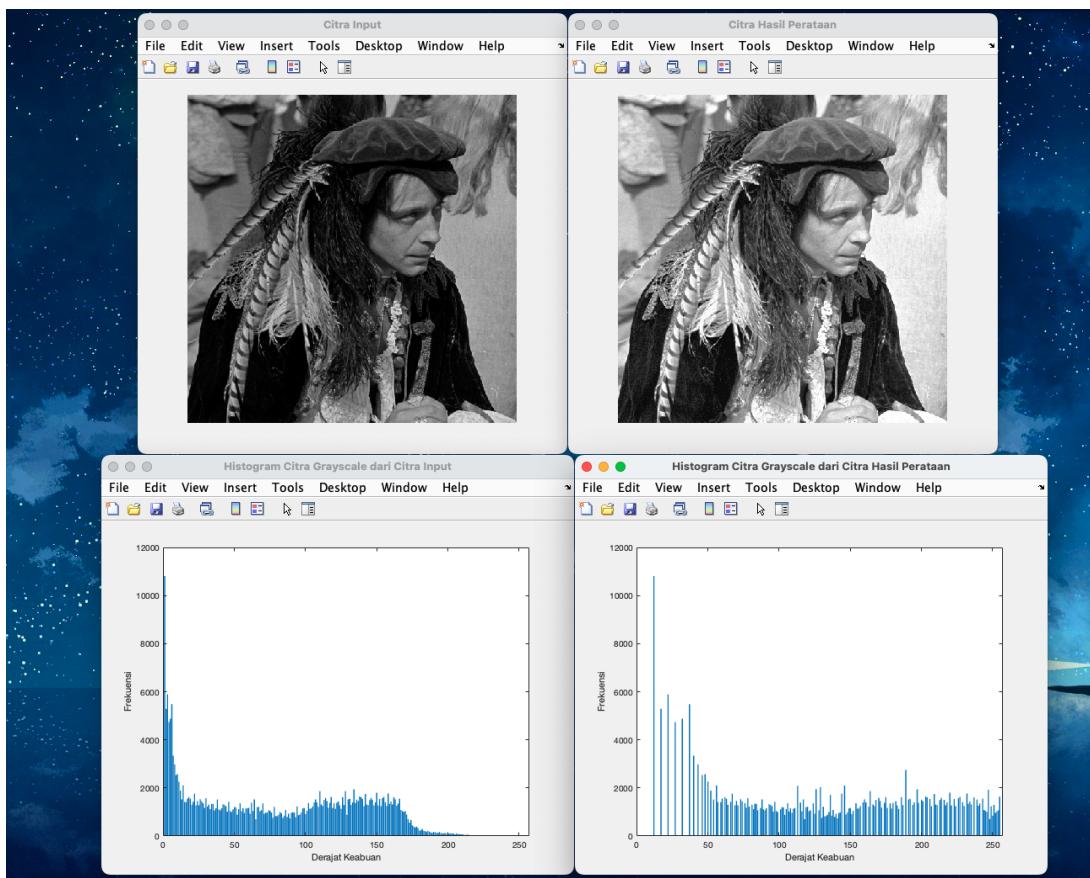
c. Hasil Eksekusi Citra 3 dari Contoh



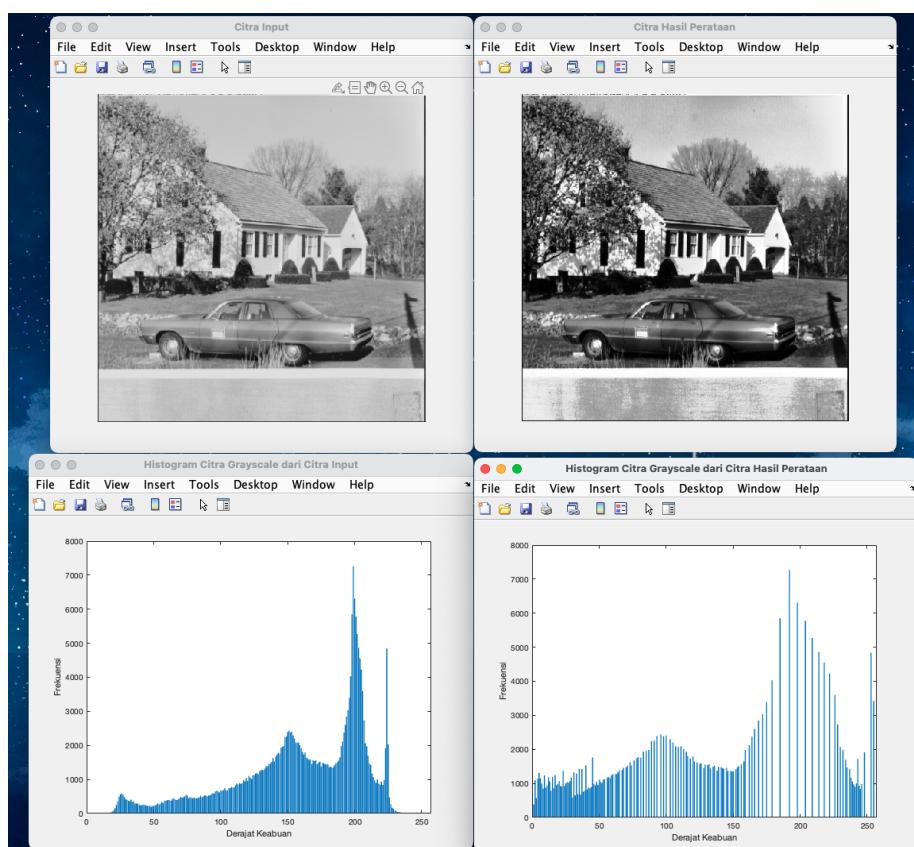
d. Hasil Eksekusi Citra 4 dari Contoh



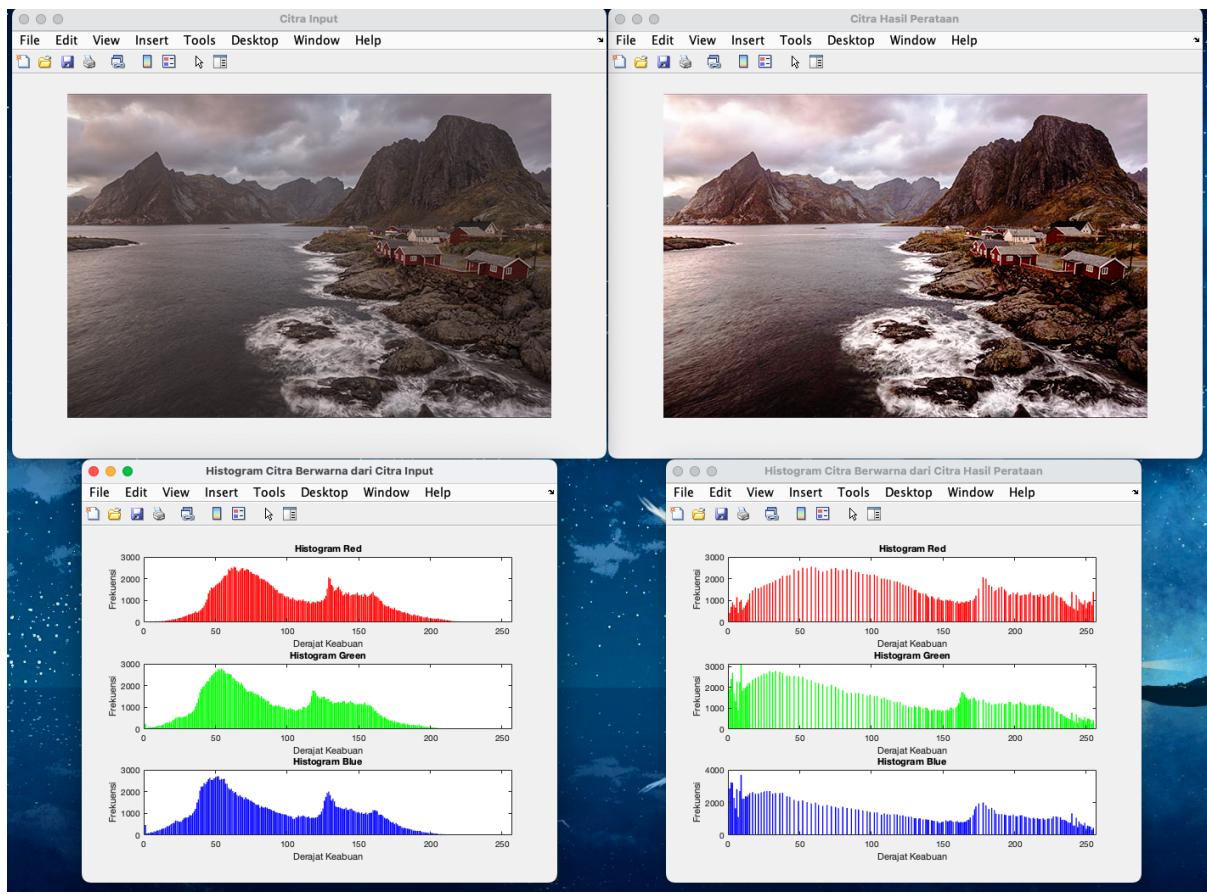
e. Hasil Eksekusi Citra 5 dari Contoh



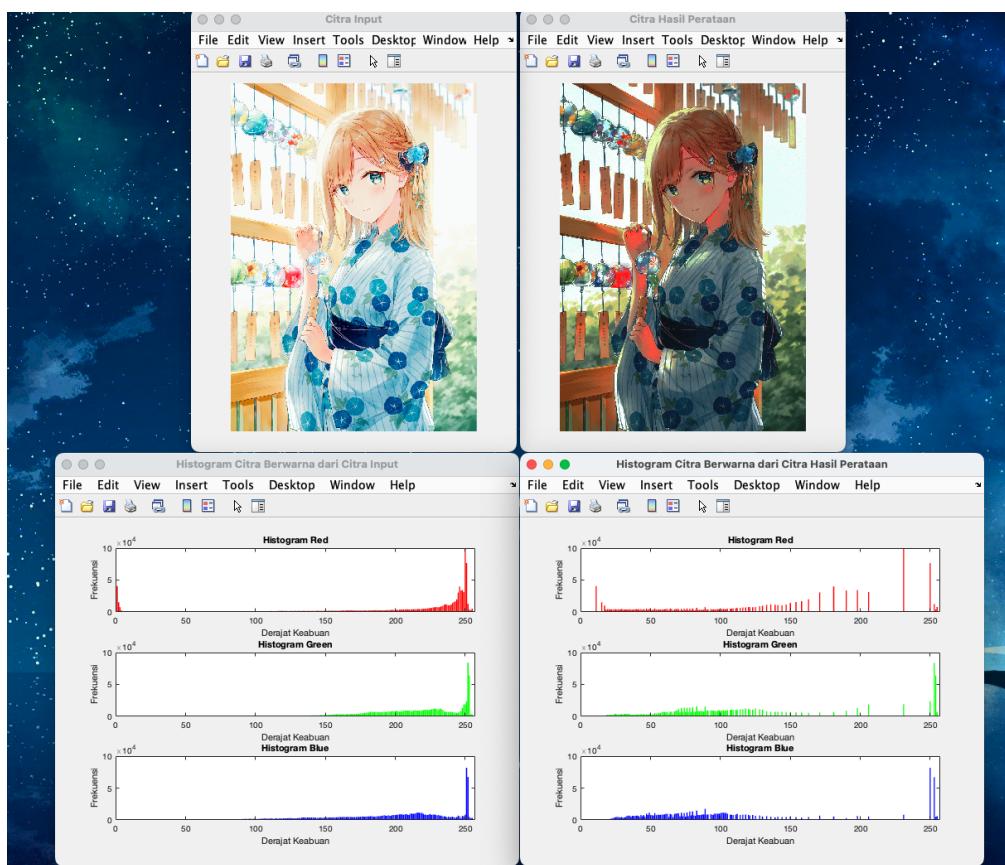
f. Hasil Eksekusi Citra 6 dari Contoh



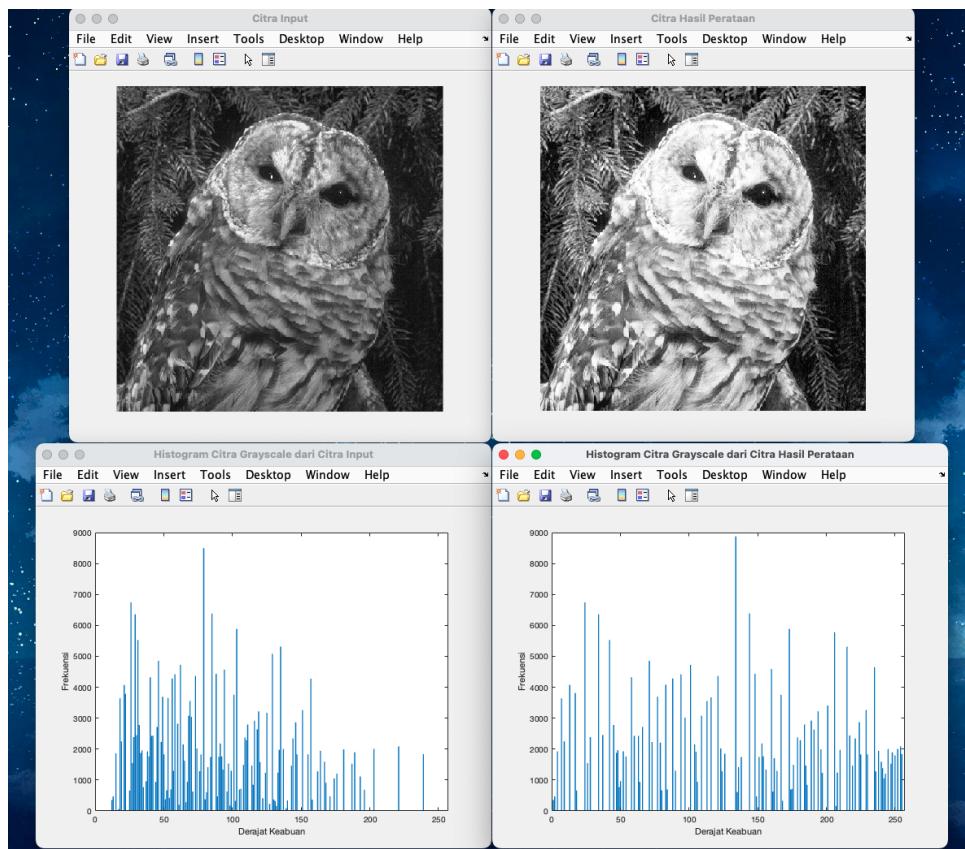
g. Hasil Eksekusi Citra 7 dari Contoh



h. Hasil Eksekusi Citra Tambahan 1

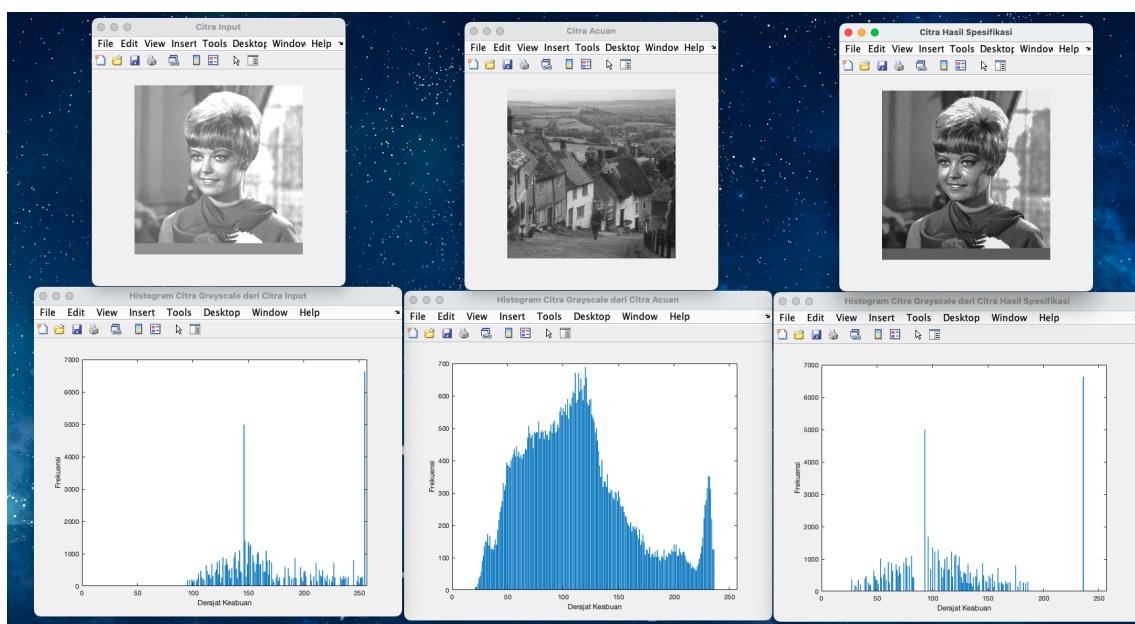


i. Hasil Eksekusi Citra Tambahan 2

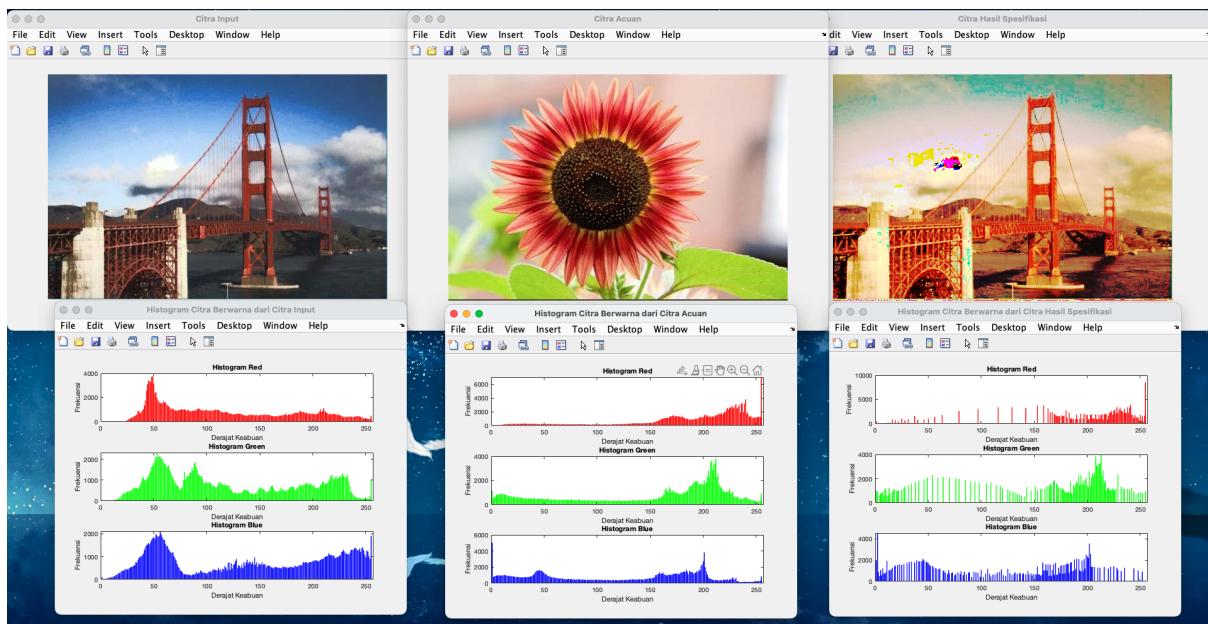


4. Program Perbaikan Citra dengan Histogram Matching

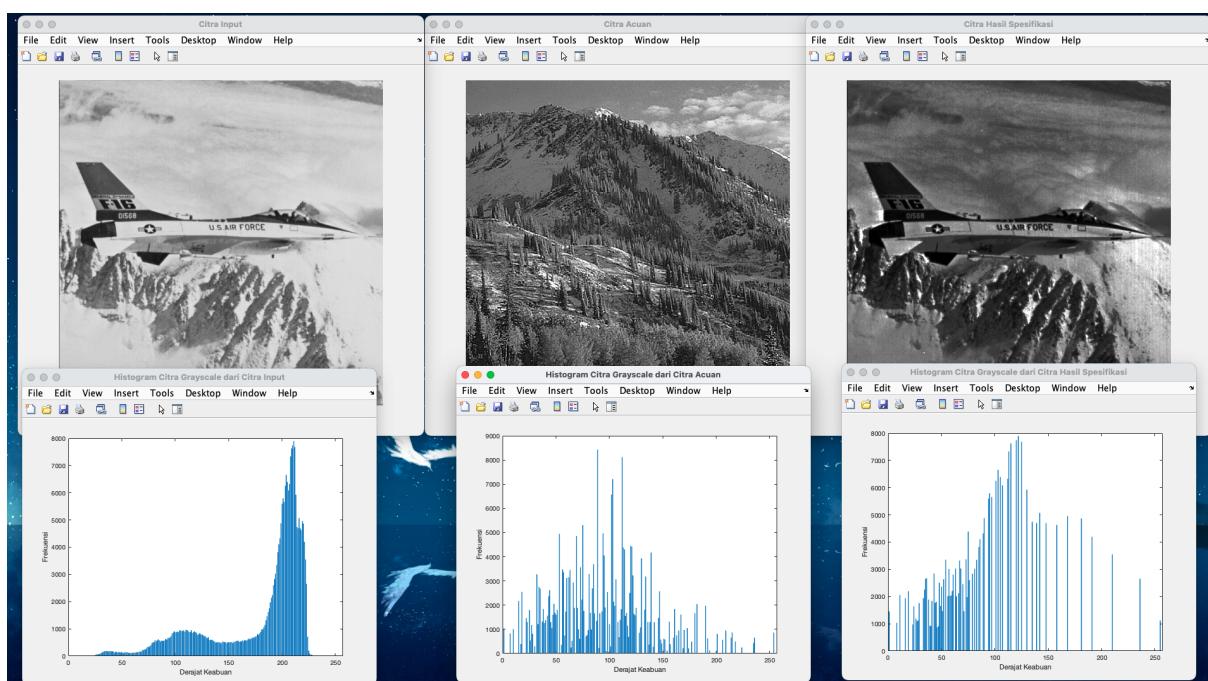
a. Hasil Eksekusi Citra 1 dari Contoh



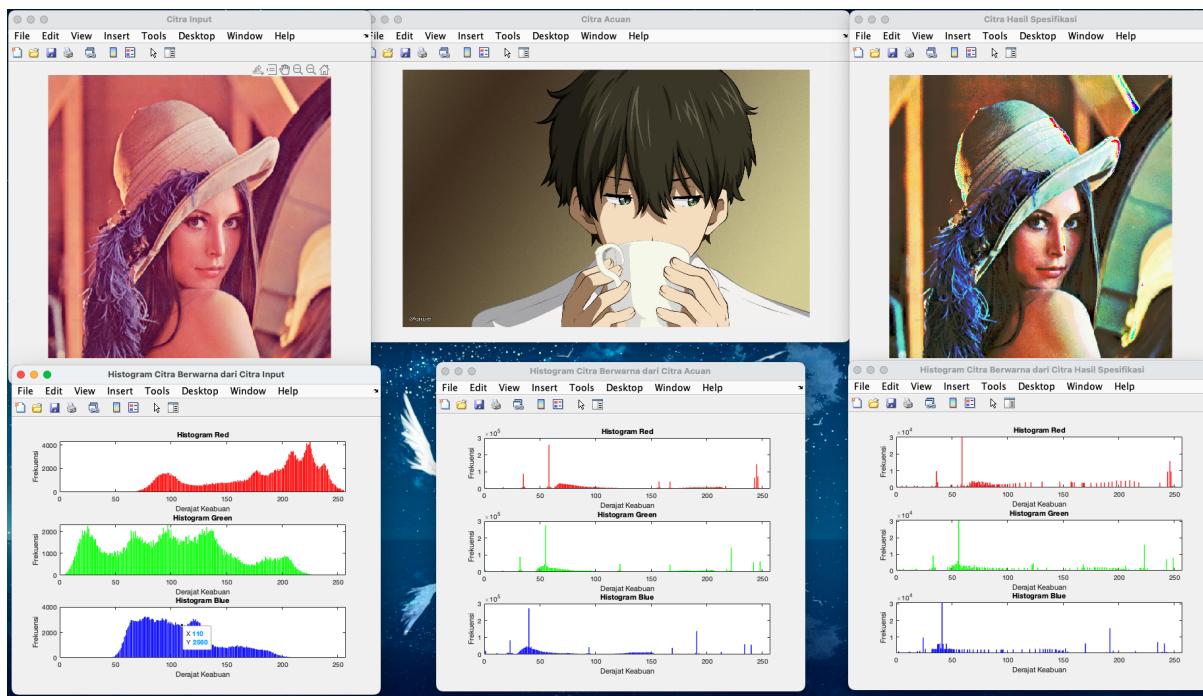
b. Hasil Eksekusi Citra 2 dari Contoh



c. Hasil Eksekusi Citra Tambahan 1



d. Hasil Eksekusi Citra Tambahan 2



C. Analisis Cara Kerja Fungsi Program dan Hasil

1. Program Menghitung dan Menampilkan Histogram Citra

Berikut adalah kode fungsi untuk menghitung dan menampilkan histogram citra.

```
function [resultGray, resultR, resultG, resultB] = getimagehistogram(~, img)
    % Menghitung histogram
    if size(img, 3) == 1
        % Citra Grayscale
        histValues = zeros(1, 256);
        for i = 1:size(img, 1)
            for j = 1:size(img, 2)
                intensity = img(i, j) + 1;
                histValues(intensity) = histValues(intensity) + 1;
            end
        end

        resultGray = histValues;
        resultR = 0;
        resultG = 0;
        resultB = 0;
    else
        % Citra Berwarna
        histValuesR = zeros(1, 256);
        histValuesG = zeros(1, 256);
        histValuesB = zeros(1, 256);
        for i = 1:size(img, 1)
            for j = 1:size(img, 2)
                intensityR = img(i, j, 1) + 1;
                intensityG = img(i, j, 2) + 1;
                intensityB = img(i, j, 3) + 1;
                histValuesR(intensityR) = histValuesR(intensityR) + 1;
                histValuesG(intensityG) = histValuesG(intensityG) + 1;
                histValuesB(intensityB) = histValuesB(intensityB) + 1;
            end
        end

        resultGray = 0;
        resultR = histValuesR;
        resultG = histValuesG;
        resultB = histValuesB;
    end
end

function imagehistogram(app,img,name)
    if size(img, 3) == 1
        [histValues, ~, ~, ~] = app.getimagehistogram(img);
        % Menampilkan histogram Grayscale dalam jendela popup
        figure('Name', ['Histogram Citra Grayscale dari', ' ', name], 'NumberTitle', 'off');
        bar(histValues);
        xlabel('Derajat Keabuan');
        ylabel('Frekuensi');
    else
        [~, histValuesR, histValuesG, histValuesB] = app.getimagehistogram(img);
        % Menampilkan histogram Citra Berwarna
        figure('Name', ['Histogram Citra Berwarna dari', ' ', name], 'NumberTitle', 'off');
        subplot(3, 1, 1);
        bar(histValuesR, 'r');
        title('Histogram Red');
        xlabel('Derajat Keabuan');
        ylabel('Frekuensi');
        subplot(3, 1, 2);
```

```

bar(histValuesG, 'g');
title('Histogram Green');
xlabel('Derajat Keabuan');
ylabel('Frekuensi');
subplot(3, 1, 3);
bar(histValuesB, 'b');
title('Histogram Blue');
xlabel('Derajat Keabuan');
ylabel('Frekuensi');
end
end

```

Fungsi untuk mendapatkan histogram terdiri dari 2, `getimagehistogram` dan `imagehistogram`. `getimagehistogram` memiliki satu parameter masukan: `img` (citra yang akan dihitung histogramnya) dan untuk `imagehistogram` memiliki tiga parameter masukan: `~` (tidak digunakan), `img` (citra yang akan dihitung histogramnya), dan `name` (nama citra yang akan ditampilkan dalam judul jendela histogram).

Fungsi mula-mula memeriksa apakah citra yang diberikan adalah citra grayscale atau citra berwarna dengan menggunakan `if size(img, 3) == 1`. Jika citra grayscale, maka histogram grayscale akan dihitung. Untuk menghitung histogram, fungsi menggunakan loop `for` untuk mengiterasi melalui setiap piksel dalam citra. Setiap nilai intensitas piksel ditambahkan dengan 1 dan digunakan sebagai indeks dalam array `histValues`. Array ini digunakan untuk menghitung frekuensi masing-masing nilai intensitas.

Jika citra adalah citra berwarna, fungsi akan menghitung histogram untuk masing-masing saluran warna (merah, hijau, dan biru). Ini dilakukan dalam tiga loop bersarang yang mengiterasi melalui piksel-piksel citra berwarna. Histogram masing-masing saluran warna dihitung dalam array terpisah (misalnya, `histValuesR` untuk merah, `histValuesG` untuk hijau, dan `histValuesB` untuk biru).

Setelah histogram-histogram dihitung, fungsi akan menampilkan histogram tersebut dalam tiga subplot jika citra berwarna. Setiap subplot akan menampilkan histogram untuk saluran warna merah, hijau, dan biru secara berurutan. Fungsi ini juga memberikan label sumbu x dan y serta judul untuk masing-masing subplot.

Hasil akhir dari fungsi ini adalah tampilan jendela popup yang menampilkan histogram citra sesuai dengan jenisnya (grayscale atau berwarna), dengan judul jendela berisi nama citra yang diberikan. Fungsi ini berguna untuk menganalisis distribusi intensitas piksel dalam citra dan dapat digunakan dalam berbagai aplikasi pemrosesan citra.

2. Program Perbaikan Kualitas Citra

Berikut ini adalah kode fungsi untuk melakukan pencerahan citra.

```

function results = pencerahancitra(~, img, a, b)
% Pencerahan citra
% Input:
% - img: Citra asli (dalam tipe data apapun)
% - a: Konstanta pengali
% - b: Konstanta penjumlahan
% Output:
% - results: Citra hasil pencerahan dalam tipe data uint8
if ~isa(img, 'double')

```

```

    img = im2double(img);
end
results = (a*img)+(b/255);
results = uint8(255*results);

```

Fungsi **pencerahancitra** digunakan untuk mencerahkan citra. Fungsi ini memiliki tiga parameter masukan: **img** (citra asli), **a** (konstanta pengali), dan **b** (konstanta penjumlahan). Pertama, fungsi memeriksa jenis data citra. Jika citra bukan tipe data double, maka konversi ke tipe data double dilakukan dengan `im2double`. Kemudian, citra asli dikalikan dengan **a** dan ditambah dengan **b/255** (untuk mengubah konstanta penjumlahan ke dalam rentang [0, 1]). Hasilnya dikonversi ke tipe data `uint8` dengan memetakan nilai-nilai ke rentang [0, 255] menggunakan faktor 255.

Berikut ini adalah kode fungsi untuk melakukan pembalikan citra.

```

function results = pembalikancitra(~,img)
% Pembalikan citra
% Input:
% - img: Citra asli (dalam tipe data apapun)
% Output:
% - results: Citra hasil balikan
results = 255 - img;

```

Fungsi **pembalikancitra** digunakan untuk membalik citra. Fungsi ini hanya memiliki satu parameter masukan, yaitu **img** (citra asli). Hasilnya adalah citra yang telah dibalik dengan mengurangkan setiap nilai piksel dari 255. Hasilnya adalah citra negatif dari citra asli.

Berikut ini adalah kode fungsi untuk melakukan transformasi log pada citra.

```

function results = transformasilog(~,img,c)
% Transformasi log pada citra
% Input:
% - img: Citra asli (dalam tipe data apapun)
% - c: Konstanta transformasi
% Output:
% - results: Citra hasil transformasi dalam tipe data uint8
if ~isa(img, 'double')
    img = im2double(img);
end
results = c*log(img+1);
results = uint8(255*results);

```

Fungsi **transformasilog** digunakan untuk melakukan transformasi log pada citra. Fungsi ini memiliki dua parameter masukan: **img** (citra asli) dan **c** (konstanta transformasi). Pertama, fungsi memeriksa jenis data citra dan mengonversi jika diperlukan. Kemudian, citra asli ditransformasi menggunakan rumus $c \cdot \log(img + 1)$. Hasilnya kemudian dikonversi ke tipe data `uint8`.

Berikut ini adalah kode fungsi untuk melakukan transformasi pangkat pada citra.

```

function results = transformasipangkat(~,img,c,gamma)
% Transformasi pangkat pada citra

```

```
% Input:
%   - img: Citra asli (dalam tipe data apapun)
%   - c: Konstanta positif transformasi
%   - gamma: Konstanta positif pangkat transformasi
% Output:
%   - results: Citra hasil transformasi dalam tipe data uint8
if ~isa(img, 'double')
    img = im2double(img);
end
results = c*(img.^gamma);
results = uint8(255*results);
end
```

Fungsi **transformasipangkat** digunakan untuk melakukan transformasi pangkat pada citra. Fungsi ini memiliki tiga parameter masukan: **img** (citra asli), **c** (konstanta positif transformasi), dan **gamma** (konstanta positif pangkat transformasi). Seperti fungsi sebelumnya, citra asli diperiksa jenis datanya dan dikonversi jika diperlukan. Kemudian, citra asli dinaikkan ke pangkat **gamma** dan dikalikan dengan **c**. Hasilnya dikonversi ke tipe data uint8.

Berikut ini adalah kode fungsi untuk melakukan peregangan kontras pada citra.

```
function results = peregangankontras(~,img)
% Peregangan kontras pada citra
% Input:
%   - img: Citra asli (dalam tipe data apapun)
% Output:
%   - results: Citra hasil peregangan dalam tipe data uint8

% Inisiasi rmax dan rmin
rmax = max(img, [], [1, 2]);
rmin = min(img, [], [1, 2]);

% Menghitung citra hasil
results = (img - rmin) .* (255/(rmax - rmin));
end
```

Fungsi **peregangankontras** digunakan untuk melakukan peregangan kontras pada citra. Fungsi ini memiliki satu parameter masukan, yaitu **img** (citra asli). Fungsi ini pertama-tama menghitung nilai maksimum dan minimum dari citra menggunakan max dan min untuk setiap saluran warna jika citra berwarna. Kemudian, citra hasil peregangan kontras dihitung dengan mengikuti rumus $(img - rmin) * (255 / (rmax - rmin))$. Hasilnya adalah citra dengan kontras yang diperluas.

3. Program Perataan Histogram

Berikut ini adalah kode fungsi untuk meratakan histogram citra

```
function [result, histRataGray, histRataR, histRataG, histRataB] =
perataanhistogram(app,img)
    newimage = uint8(zeros(size(img, 1), size(img, 2)));
    totalpixel = (size(img, 1) * size(img, 2));
    if size(img, 3) == 1
        [histGray, ~, ~, ~] = app.getimagehistogram(img);
        histEqGray = zeros(1, 256);
```

```

sum = 0.0;
for i = 1:256
    sum = sum + histGray(i);
    prob = sum / totalpixel;
    histEqGray(i) = round(255 * prob);
end
for i = 1:size(img, 1)
    for j = 1:size(img, 2)
        newimage(i,j) = histEqGray(img(i,j)+1);
    end
end
histRataGray = histEqGray;
histRataR = 0;
histRataG = 0;
histRataB = 0;
else
    [~, histR, histG, histB] = app.getimagehistogram(img);
    histEqR = zeros(1, 256);
    histEqG = zeros(1, 256);
    histEqB = zeros(1, 256);
    sumR = 0.0;
    sumG = 0.0;
    sumB = 0.0;
    for i = 1:256
        sumR = sumR + histR(i);
        sumG = sumG + histG(i);
        sumB = sumB + histB(i);
        probR = sumR / totalpixel;
        probG = sumG / totalpixel;
        probB = sumB / totalpixel;
        histEqR(i) = round(255 * probR);
        histEqG(i) = round(255 * probG);
        histEqB(i) = round(255 * probB);
    end
    for i = 1:size(img, 1)
        for j = 1:size(img, 2)
            newimage(i, j, 1) = histEqR(img(i, j, 1) + 1);
            newimage(i, j, 2) = histEqR(img(i, j, 2) + 1);
            newimage(i, j, 3) = histEqR(img(i, j, 3) + 1);
        end
    end
    histRataGray = 0;
    histRataR = histEqR;
    histRataG = histEqG;
    histRataB = histEqB;
end
result = newimage;
end

```

Fungsi `perataanhistogram` digunakan untuk meratakan histogram dari image yang dijadikan parameter. Fungsi ini hanya menerima 1 parameter: `img` (citra asli) dan bisa memberikan total 5 output: `result` (citra hasil), `histRataGray`, `histRataR`, `histRataG`, dan juga `histRataB`. Untuk menyelesaikan algoritma yang ingin diimplementasikan, fungsi ini mengambil fungsi `getimagehistogram` yang sudah didefinisikan sebelumnya.

Cara kerja fungsi ini adalah, mengecek terlebih dahulu apakah masukan citra merupakan citra grayscale atau citra berwarna. Tetapi, cara kerja selanjutnya tetaplah sama. Selanjutnya, dicarilah histogram dari citra asli tersebut. Setelah didapatkan histogram dari citranya, perlu dicari kemungkinan kumulatif dari intensitas warna (0 sampai 255). Setelah didapatkan semua kemungkinan kumulatif, hasil yang didapatkan digunakan untuk membuat histogram citra asal “tersebar” secara rata dan ditransformasikan untuk didapatkan citra baru.

Dan yang terakhir, meng-*assign* nilai-nilai yang didapatkan dalam parameter output yang didapatkan. Untuk hasil citra akan di-*assign* pada **result**. Untuk hasil histogram perataan untuk citra grayscale, di-*assign* pada **histRataGray**. Dan untuk hasil histogram perataan citra berwarna, pada **histRataR**, **histRataG**, dan **histRataB** untuk masing-masing channel warna. Variabel histogram ini nantinya akan digunakan untuk fungsi **spesifikasihistogram**.

4. Program Perbaikan Citra dengan *Histogram Matching*

Berikut ini adalah kode fungsi untuk menspesifikasikan histogram citra dengan histogram citra acuan

```
function result = spesifikasihistogram(app,img,imgacuan)
    newimage = uint8(zeros(size(img, 1), size(img, 2)));
    % Pemisahan antara gambar grayscale dan berwarna
    if size(img, 3) == 1
        % Mengambil histogram yang sudah diratakan untuk grayscale
        [~, hist, ~, ~, ~] = perataanhistogram(app, img);
        [~, histAcuan, ~, ~, ~] = perataanhistogram(app, imgacuan);
        histRes = zeros(1, 256);
        % Melakukan transformasi pada image input
        for i = 1:256
            minval = abs(hist(i) - histAcuan(1));
            minj = 0;
            for j = 1:256
                if abs(hist(i) - histAcuan(j)) < minval
                    minval = abs(hist(i) - histAcuan(j));
                    minj = j;
                end
            end
            histRes(i) = minj;
        end
        for i = 1:size(img, 1)
            for j = 1:size(img, 2)
                newimage(i,j) = histRes(img(i,j)+1);
            end
        end
    else
        % Mengambil histogram yang sudah diratakan untuk image
        % berwarna
        [~,~, histR, histG, histB] = perataanhistogram(app, img);
        [~,~, histAcuanR, histAcuanG, histAcuanB] = perataanhistogram(app, imgacuan);
        histResR = zeros(1, 256);
        histResG = zeros(1, 256);
        histResB = zeros(1, 256);
        % Melakukan transformasi pada image input
```

```

for i = 1:256
    doneR = false;
    doneG = false;
    doneB = false;
    for j = 1:256
        if histR(i) < histAcuanR(j) & ~doneR
            histResR(i) = j;
            doneR = true;
        end
        if histG(i) < histAcuanG(j) & ~doneG
            histResG(i) = j;
            doneG = true;
        end
        if histB(i) < histAcuanB(j) & ~doneB
            histResB(i) = j;
            doneB = true;
        end
        if doneR & doneG & doneB
            break;
        end
    end
end
% Mentransformasi hasil spesifikasi histogram pada gambar
for i = 1:size(img, 1)
    for j = 1:size(img, 2)
        newimage(i, j, 1) = histResR(img(i, j, 1) + 1);
        newimage(i, j, 2) = histResG(img(i, j, 2) + 1);
        newimage(i, j, 3) = histResB(img(i, j, 3) + 1);
    end
end
end
result = newimage;

```

Fungsi **spesifikasihistogram** digunakan untuk meratakan histogram dari image yang dijadikan parameter. Fungsi ini hanya menerima 2 parameter: **img** (citra asli) dan **imgacuan** (citra acuan) dan juga memberikan 1 output: **result** (citra hasil). Untuk menyelesaikan algoritma yang ingin diimplementasikan, fungsi ini mengambil fungsi **perataanhistogram** yang sudah didefinisikan sebelumnya.

Cara kerja fungsi ini adalah, mengecek terlebih dahulu apakah masukan citra merupakan citra grayscale atau citra berwarna. Tetapi, cara kerja selanjutnya tetaplah sama. Langkah selanjutnya adalah mendapatkan histogram rataan baik untuk citra asli maupun imgacuan. Setelah didapatkan histogram hasil rataannya, kemudian akan dilakukan transformasi agar didapatkan histogram hasil yang sesuai dengan histogram citra acuan. Setelah itu, hasilnya akan ditransformasi untuk dijadikan citra baru.

Output dari fungsi ini adalah sebuah citra baru yang telah ditransformasikan dari histogram citra input dan histogram citra acuan.

LAMPIRAN

Alamat GitHub: <https://github.com/dennisheraldi/IF4073-Tugas1-Citra-2023-1>