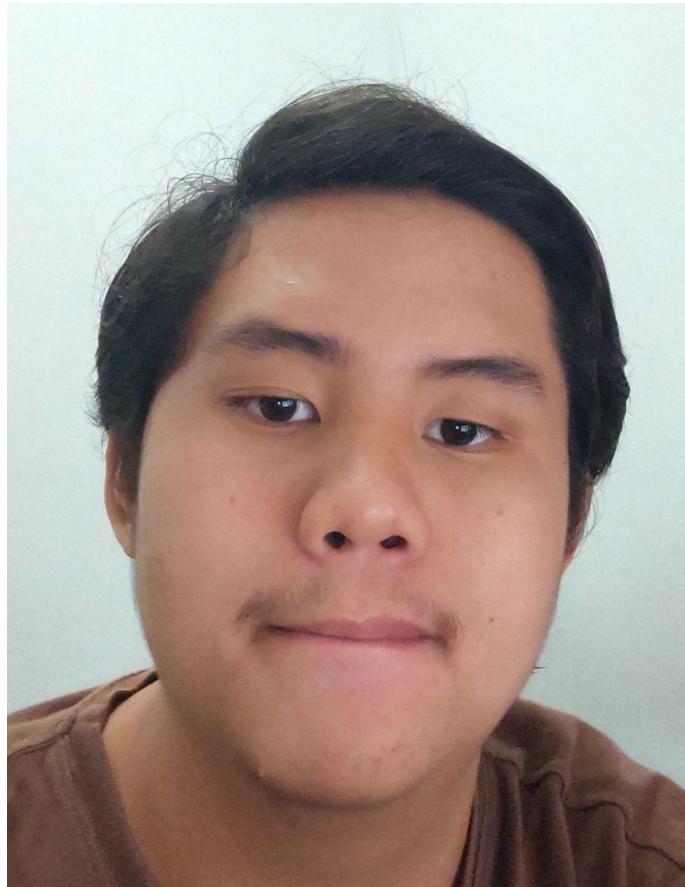


Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra

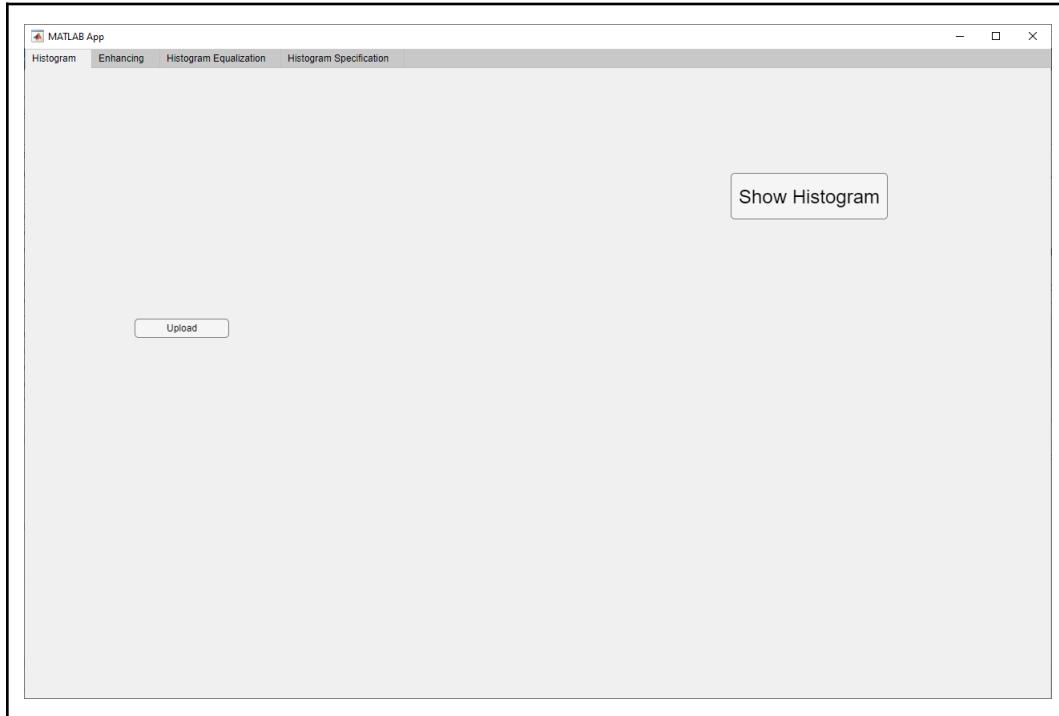
Laporan Tugas 1 Interpretasi dan Pengolahan Citra



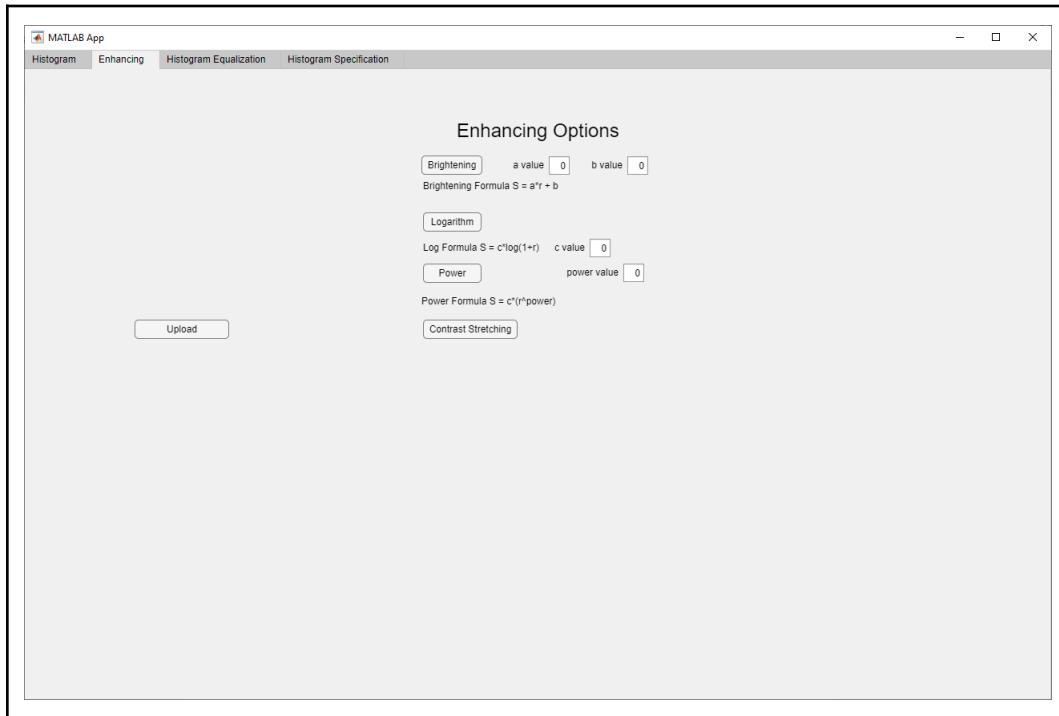
Naufal Yahya Kurnianto
13519141

1. Screenshot GUI Program

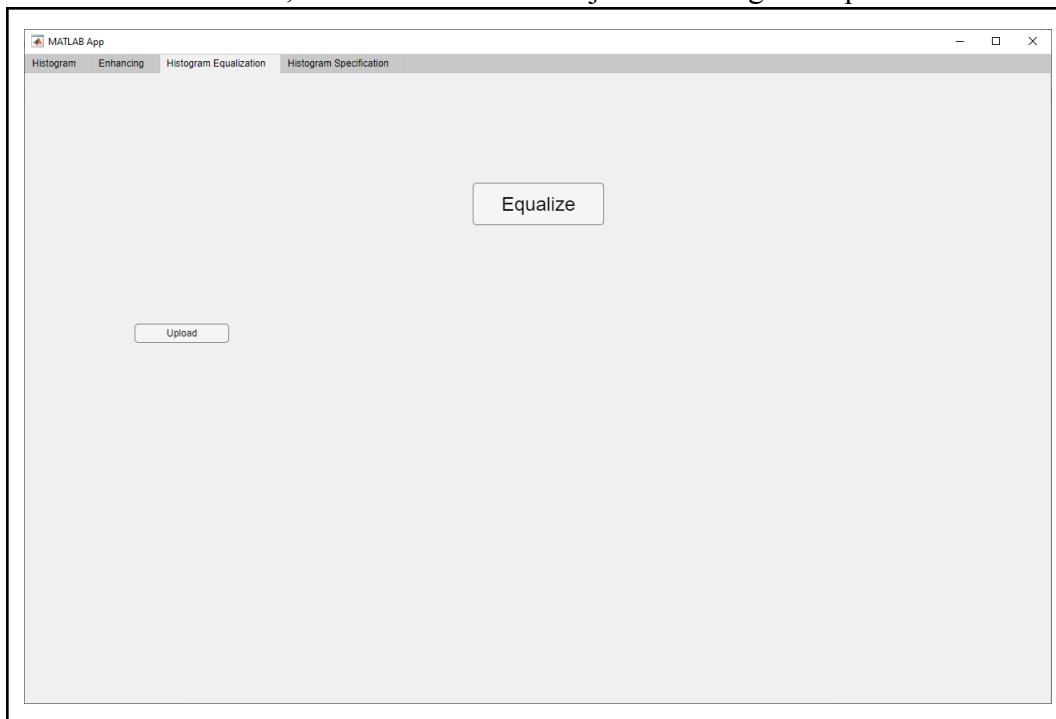
- a. Halaman untuk soal 1, menunjukkan histogram gambar



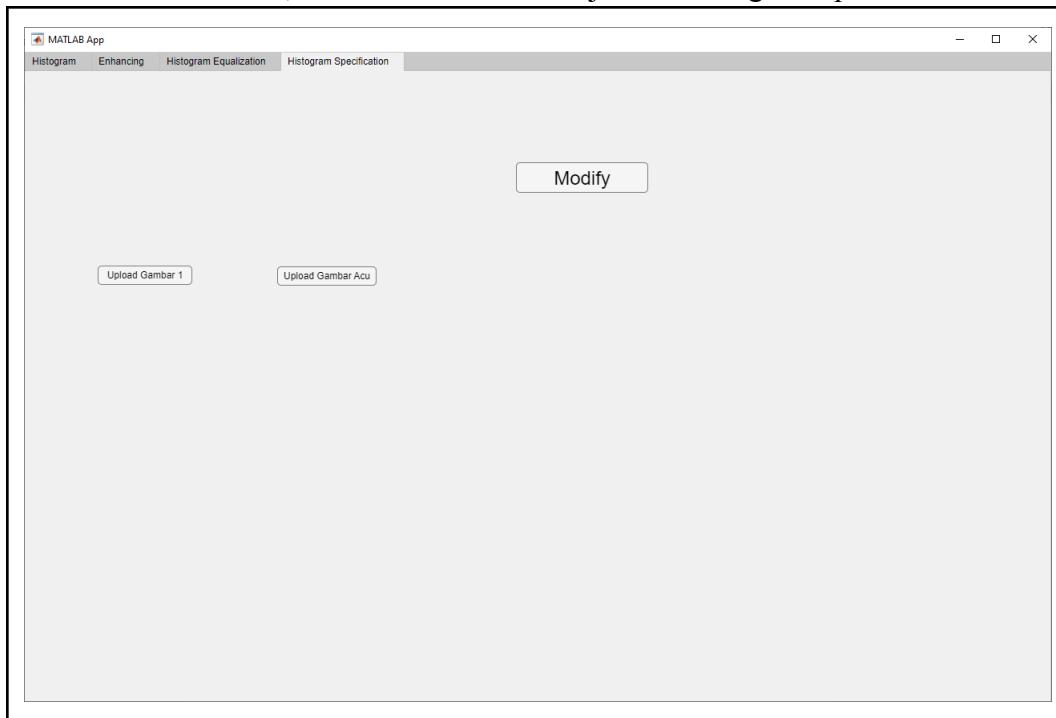
- b. Halaman untuk soal 2, melakukan dan menunjukkan hasil image enhancing



- c. Halaman untuk soal 3, melakukan dan menunjukkan histogram equalization



- d. Halaman untuk soal 4, melakukan dan menunjukkan histogram specification



Pada beberapa potongan gambar di atas, hanya terlihat tombol-tombol yang nantinya akan digunakan, akan diperlihatkan tampilan GUI yang lebih jelas nantinya pada contoh penggunaan kode program pada GUI. Sebagai perhatian, nantinya akan ada beberapa histogram yang tidak cukup untuk ditampilkan di jendela GUI ini sehingga akan

menggunakan jendela (figure di matlab) baru untuk beberapa histogram tersebut. Untuk penggunaannya sendiri menggunakan alur yang sama untuk seluruh halaman. Upload gambar terlebih dahulu (untuk halaman 3 dan 4 histogram gambar awal akan langsung ditampilkan pada jendela yang berbeda serta ada 2 gambar yang harus diupload pada halaman 4). Kemudian, tekan tombol show, equalize, atau modify untuk halaman 1, 3, atau 4 serta mengisi field terlebih dahulu sebelum menekan tombol dari opsi enhancing pada halaman 2. Setelah itu, citra hasil operasi kode akan ditampilkan beserta histogramnya

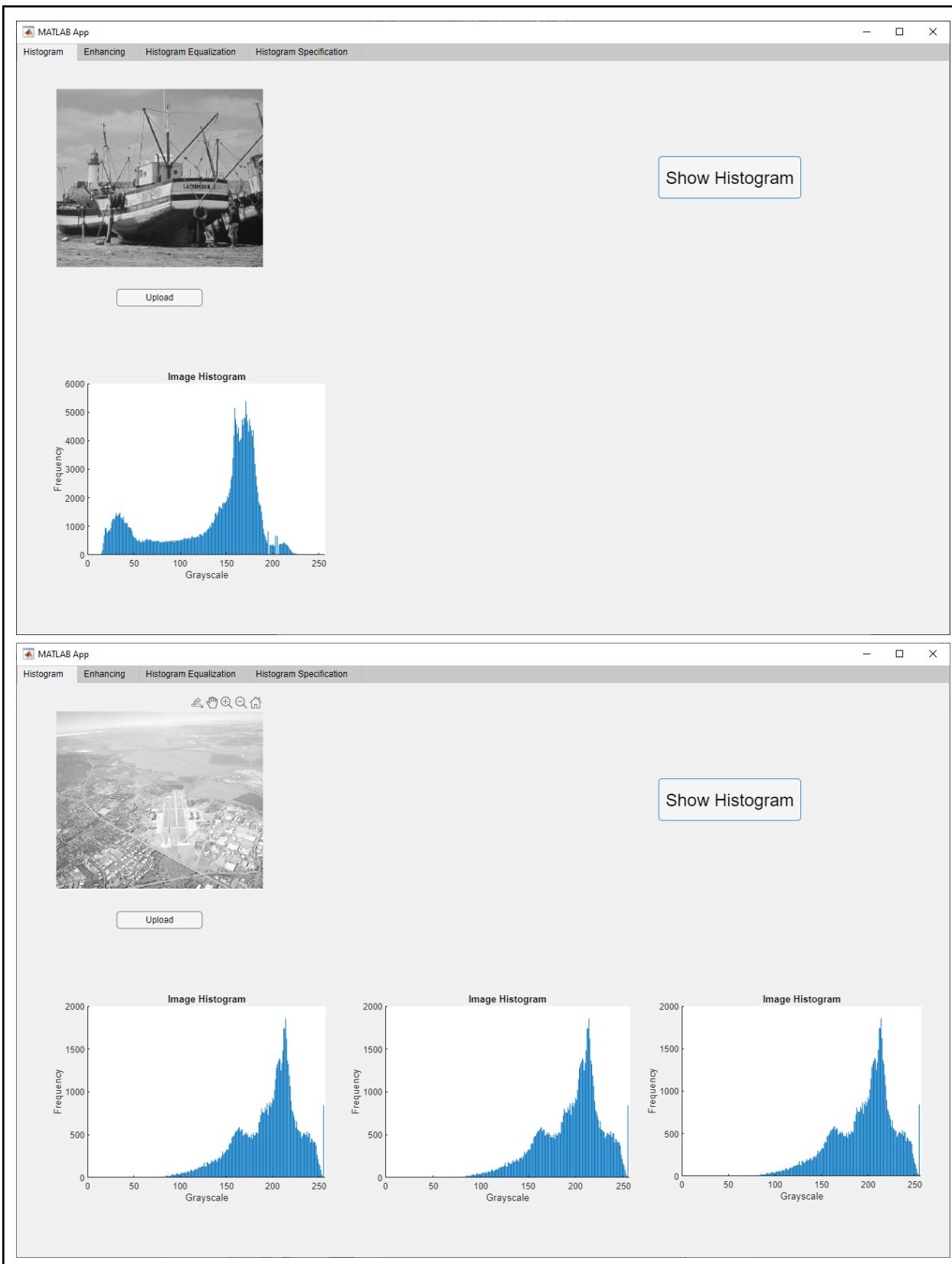
2. Kode, contoh, dan analisis program

a. Soal 1, menunjukkan histogram gambar

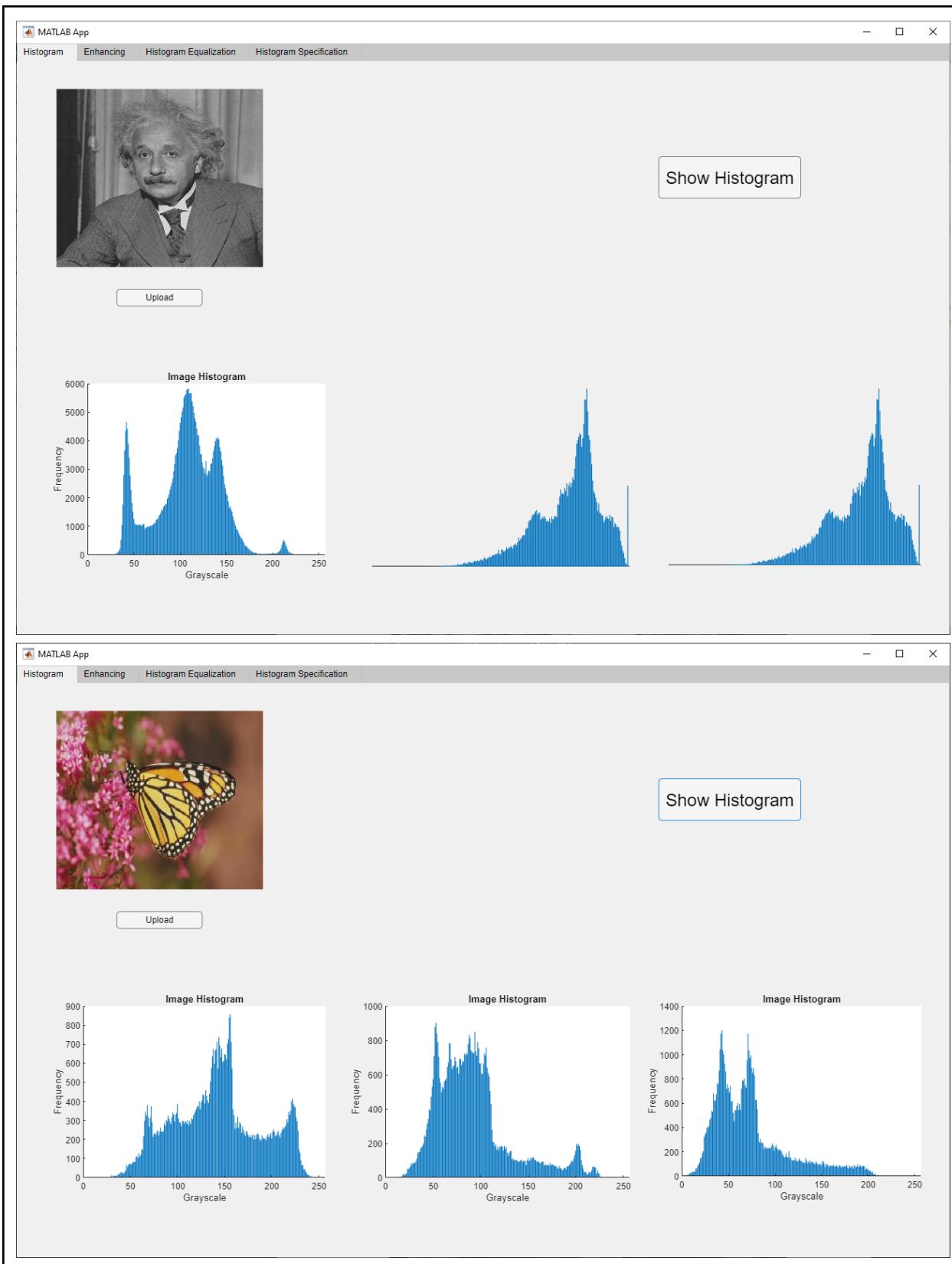
Pada soal pertama ini, hitung jumlah anggota dari tiap-tiap nilai keabuan kemudian dipetakan menjadi bentuk histogram. Untuk citra grayscale, histogram yang dihasilkan hanya 1 karena komponennya hanya 1 layer sedangkan untuk citra berwarna terdapat 3 layer yaitu red, green, dan blue. Berikut potongan kode program, contoh pengaplikasian pada GUI, dan analisis tambahan untuk soal pertama ini.

```
function [xax,imghist] = procHistogram(app,img,rows,cols,rgb)
    if isa(img,"double") % ada hasil image enhancing yang isi imgnya range [0..1] sehingga harus diubah dahulu
        img = img*256;
    end
    imghist = zeros(1,256); % inisialisasi histogram
    xax = 1:256; % x axis dari histogram
    if (rgb>1) % jika merupakan gambar berwarna, histogram ada 3 layer
        imghist(:,:,2) = 0;
        imghist(:,:,3) = 0;
    end
    for i = 1:rgb % looping dilakukan per layer
        for c = 1:cols
            for r = 1:rows
                pixval = img(r,c,i); % pixel value
                if pixval > 256
                    pixval = 256;
                elseif pixval < 1
                    pixval = 1;
                end
                % tiap index(yang menggambarkan nilai pixel) bertambah jumlah(elemen)nya
                imghist(:,int64(pixval),i) = imghist(:,int64(pixval),i) + 1;
            end
        end
    end
end
```

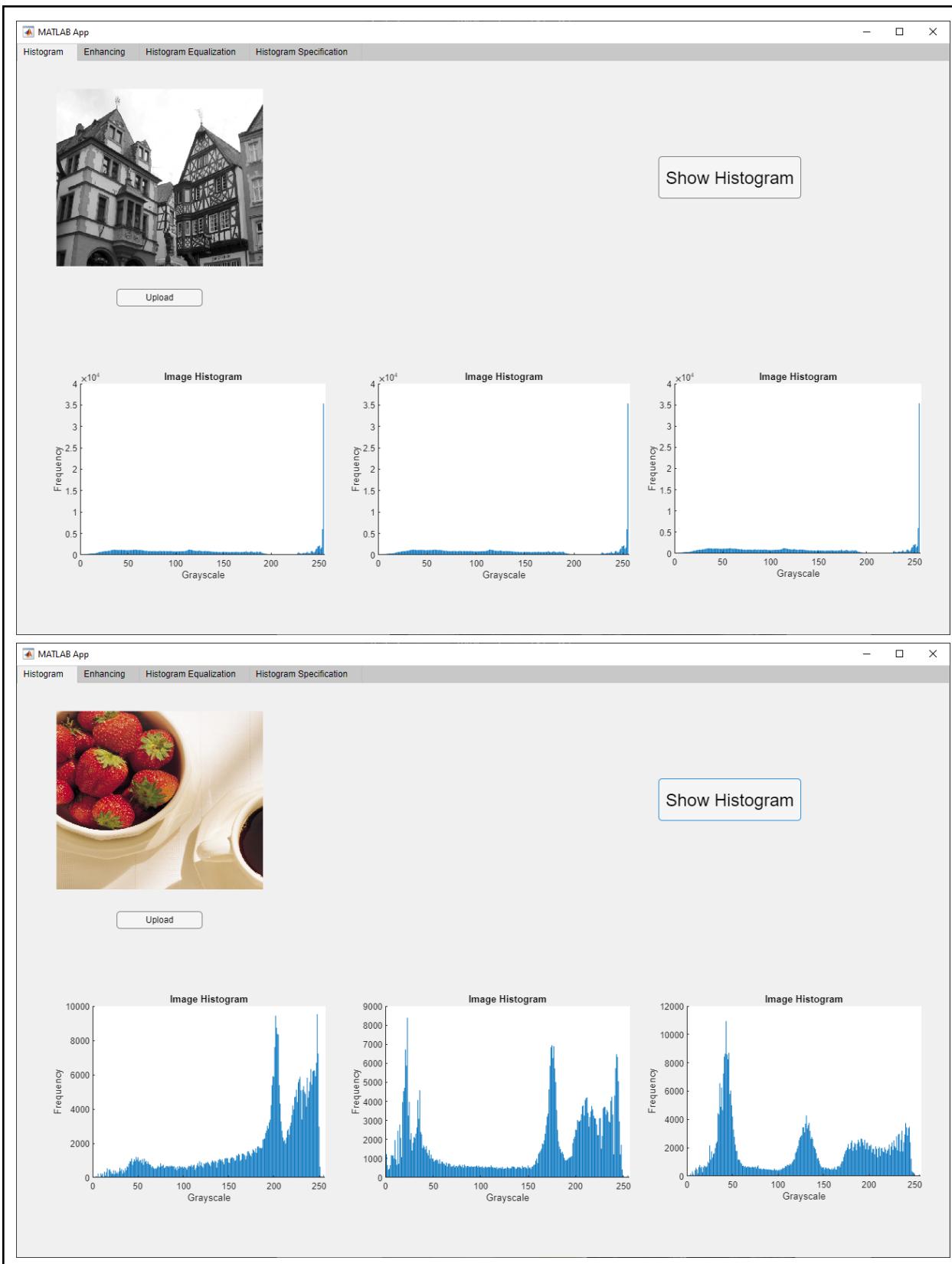
Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra



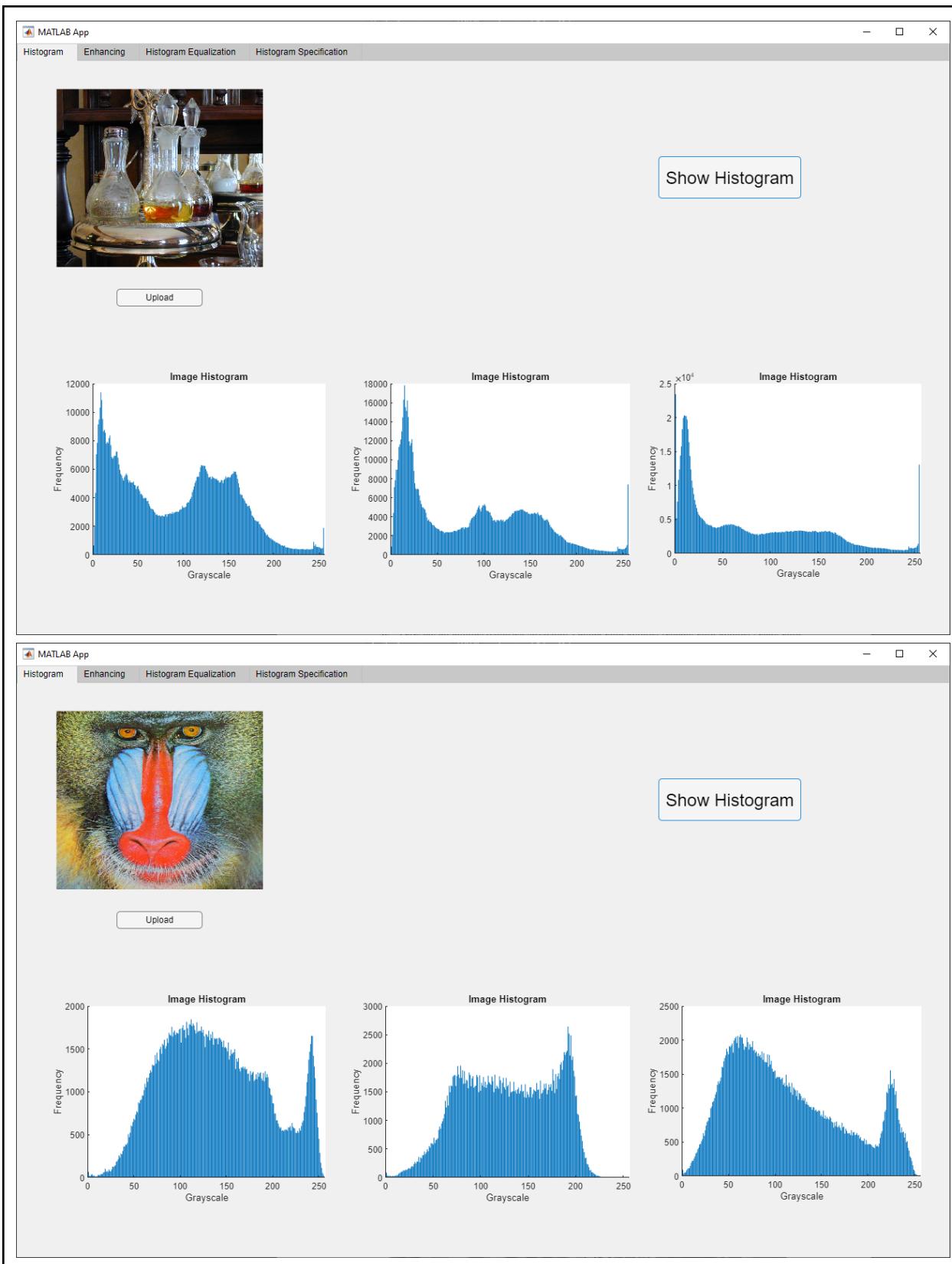
Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra



Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra



Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra





Pada berbagai contoh eksekusi kode program di GUI tersebut ada beberapa kecacatan yang dapat diidentifikasi. Pertama adalah adanya citra grayscale yang memiliki 3 layer pewarnaan. Hal tersebut disebabkan oleh spek gambar yang diperoleh dengan melakukan cropping pada dokumen spek kemudian menyimpannya dalam format PNG. Menggunakan metode tersebut, walaupun citranya merupakan citra grayscale, format citranya sendiri adalah citra berwarna atau RGB. Kecacatan kedua adalah, pada GUI ada citra grayscale (citra einstein) yang menunjukkan 3 histogram dimana 2 dari histogram tersebut merupakan histogram citra bekas pengoperasian citra sebelumnya. Dapat dilihat perbandingannya pada citra ke-2(runway) dengan citra ke-3(einstein) bahwa 2 histogram pada posisi tengah dan kanan sama. Namun demikian, histogram citra ke-3(einstein) yang merupakan histogram grayscale sudah benar isinya dan terletak pada bagian kiri.

Sebagai tambahan, urutan histogram adalah red - green - blue.

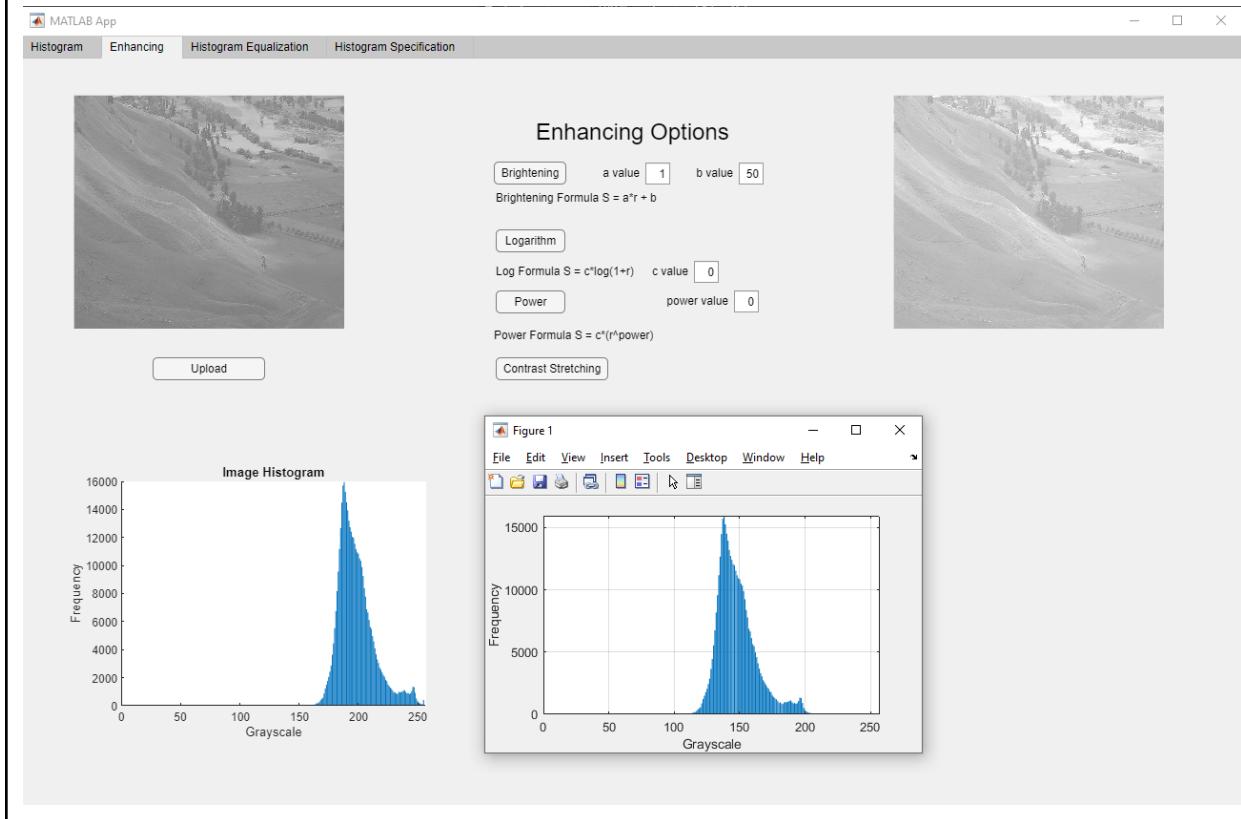
- b. Soal 2, melakukan perbaikan atau penguatan citra

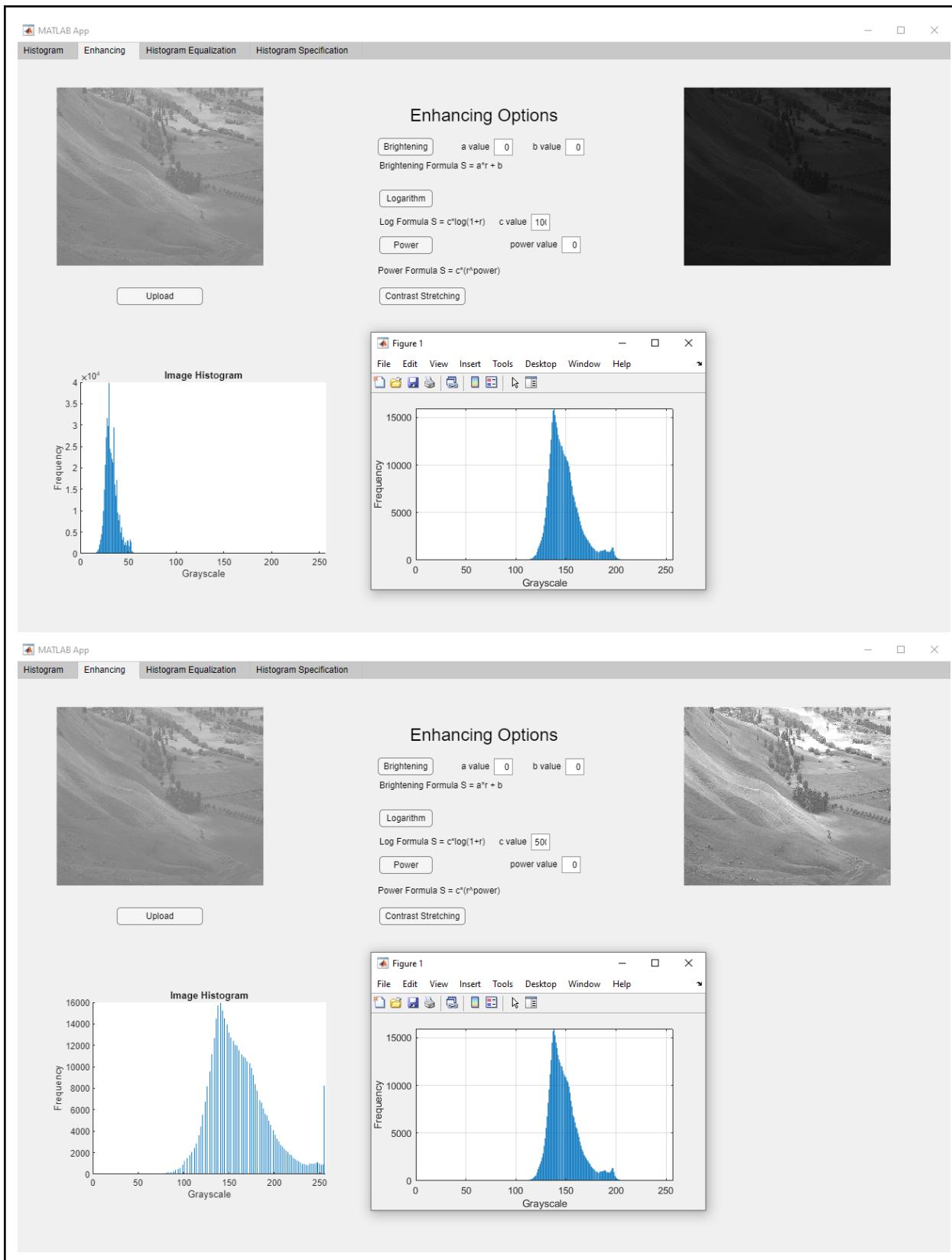
Pada soal kedua ini, terdapat empat opsi yang masing-masing harus diaplikasikan pada kode program. Keempat operasi perbaikan atau penguatan citra di antaranya adalah: Image brightening; Logarithmic transformation; Power transformation; dan Contrast stretching. Sama seperti sebelumnya untuk citra grayscale, histogram yang dihasilkan hanya 1 karena komponennya hanya 1 layer sedangkan untuk citra berwarna terdapat 3 layer yaitu red, green, dan blue. Berikut potongan

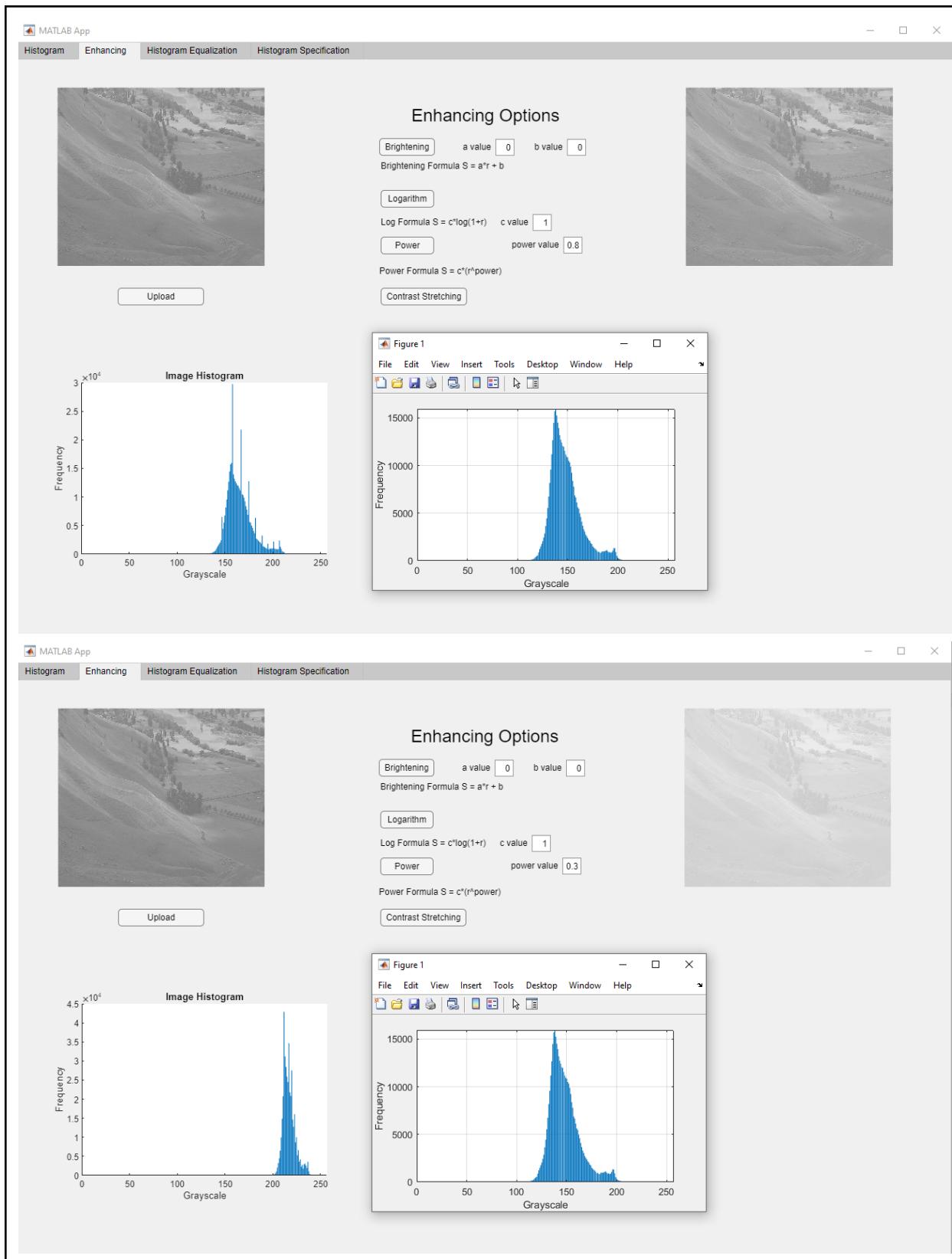
kode program, contoh pengaplikasian pada GUI, dan analisis tambahan untuk soal kedua ini.

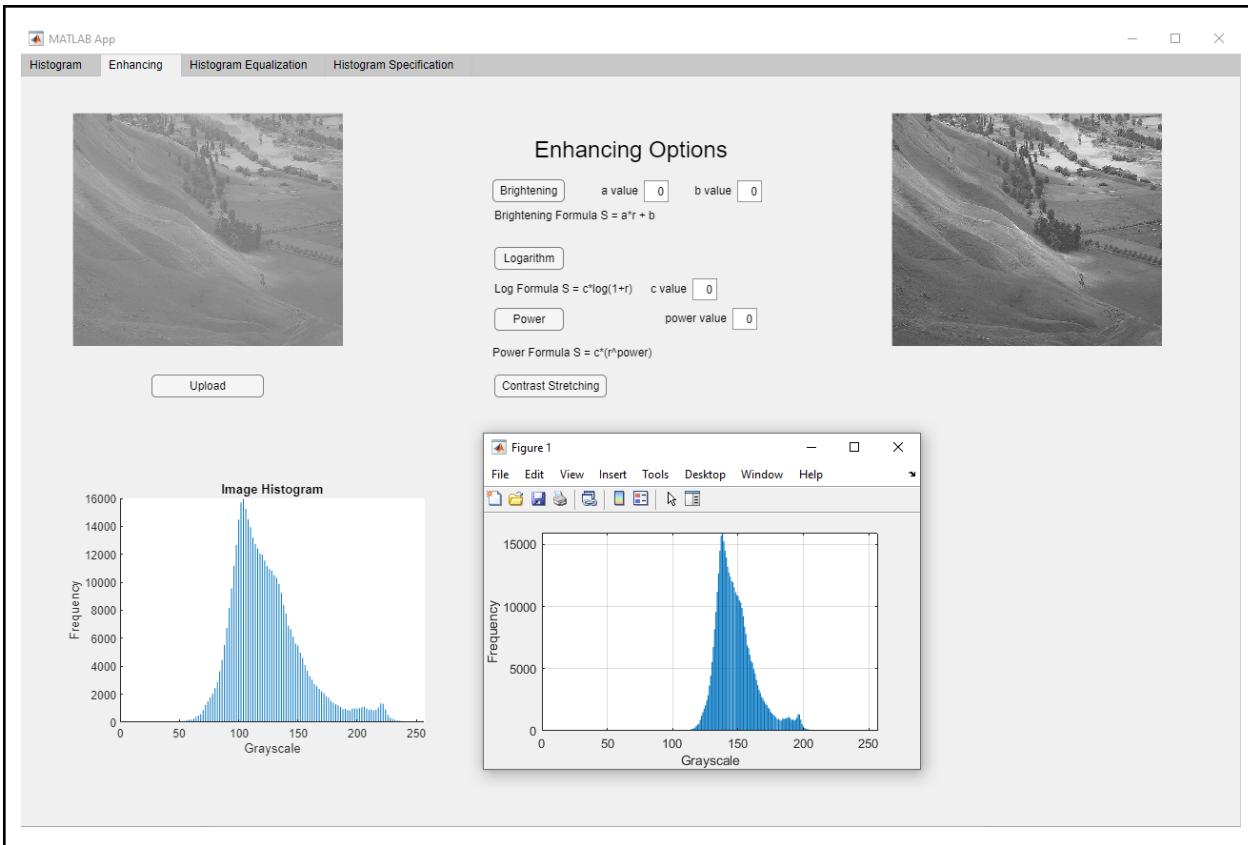
```
function newimg = enhImage(app,img,rows,cols,rgb,type)
    if type == 1 % image brightening
        a = app aValue Value; % nilai a dari GUI
        b = app bValue Value; % nilai b dari GUI
        newimg = img; % placeholder citra baru, speknya sama dengan citra lama
        for i = 1:rgb % rumus brightening diaplikasikan pada tiap-tiap pixel
            for c = 1:cols
                for r = 1:rows
                    newimg(r,c,i) = img(r,c,i)*a+b; % pengaplikasian rumus a*r+b
                    if newimg(r,c,i) > 255
                        newimg(r,c,i) = 255;
                    elseif newimg(r,c,i) < 0
                        newimg(r,c,i) = 0;
                    end
                end
            end
        end
    elseif type == 2 % logarithmic transformation
        constant = app cValue Value; % nilai c atau konstanta dari GUI
        newimg = img; % placeholder citra baru, speknya sama dengan citra lama
        var = log1p(mat2gray(img)); % operasi logaritma, log(1+r)
        for i = 1:rgb % operasi logaritma dilakukan pada tiap-tiap pixel
            for c = 1:cols
                for r = 1:rows
                    newimg(r,c,i) = constant*var(r,c,i); % pengaplikasian rumus c*log(1+r)
                    if newimg(r,c,i) > 255
                        newimg(r,c,i) = 255;
                    elseif newimg(r,c,i) < 0
                        newimg(r,c,i) = 0;
                    end
                end
            end
        end
    elseif type == 3 % power transformation
        constant = app cValue Value; % nilai c atau konstanta dari GUI
        pow = app pValue Value; % nilai power atau pangkat dari GUI (power atau gamma)
        var = im2double(img);
        newimg = constant * (var.^pow); % pengaplikasian rumus c*r^p
    elseif type == 4 % contrast stretching
        min = 257; % nilai grayscale terkecil
        max = 0; % nilai grayscale terbesar
        for i = 1:rgb
            for c = 1:cols
```

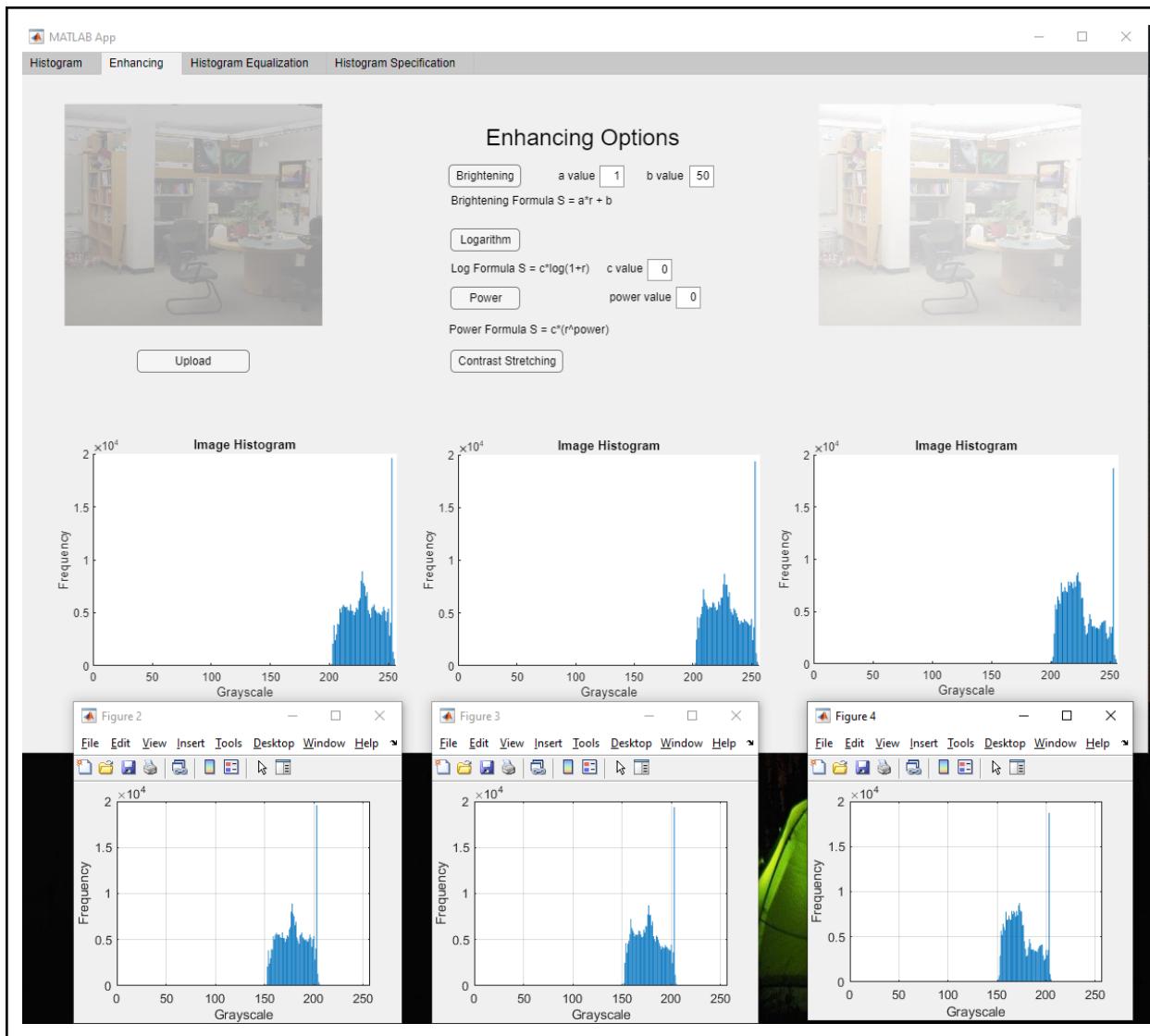
```
for r = 1:rows
    if min > img(r,c,i)
        min = img(r,c,i);
    elseif max < img(r,c,i)
        max = img(r,c,i);
    end
end
newimg(:,:,:,i) = (img(:,:,:,:) - min)*(256/(max - min)); % pengaplikasian rumus
jarak antar titik (dari PPT)
end
end
end
```

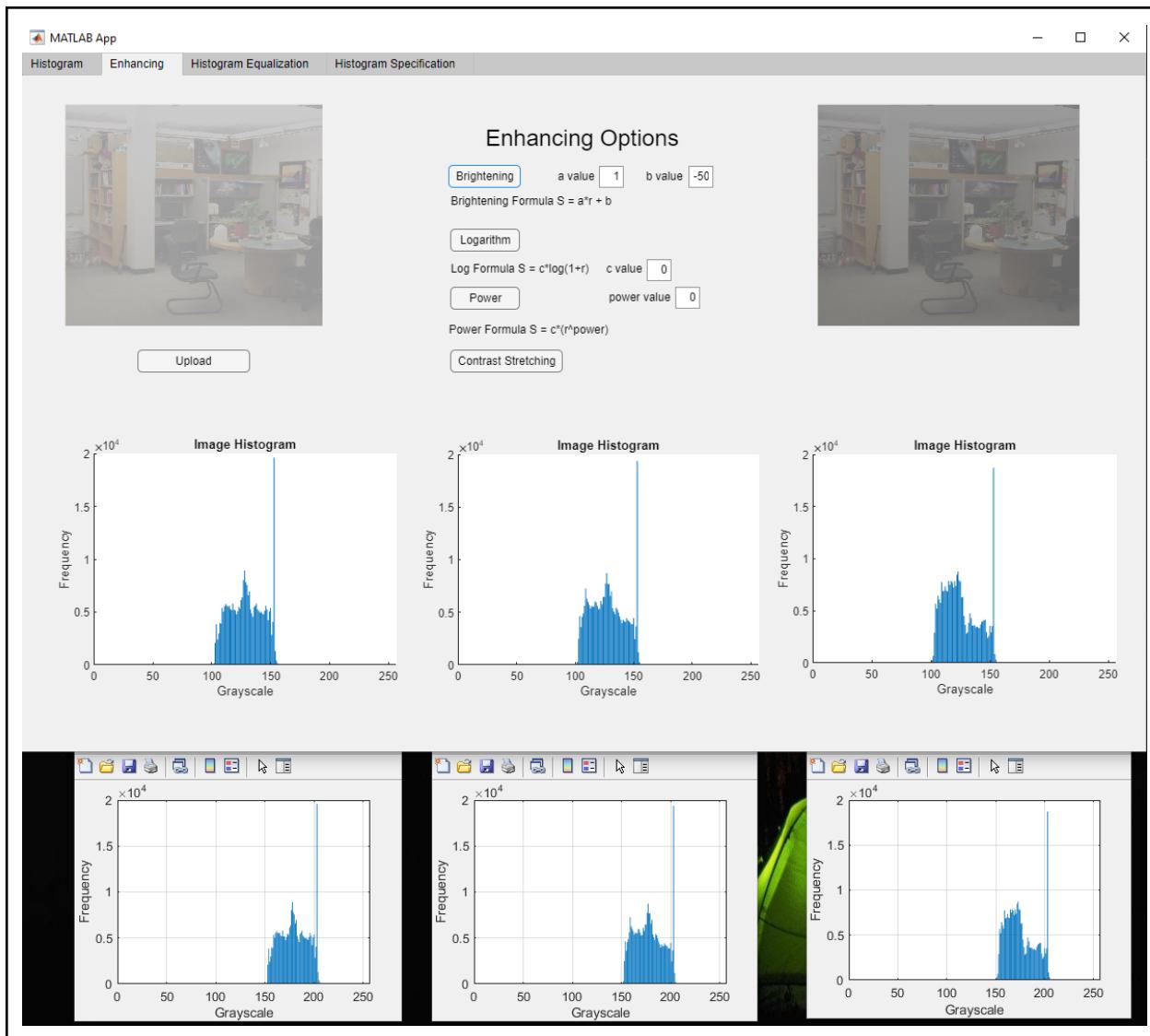


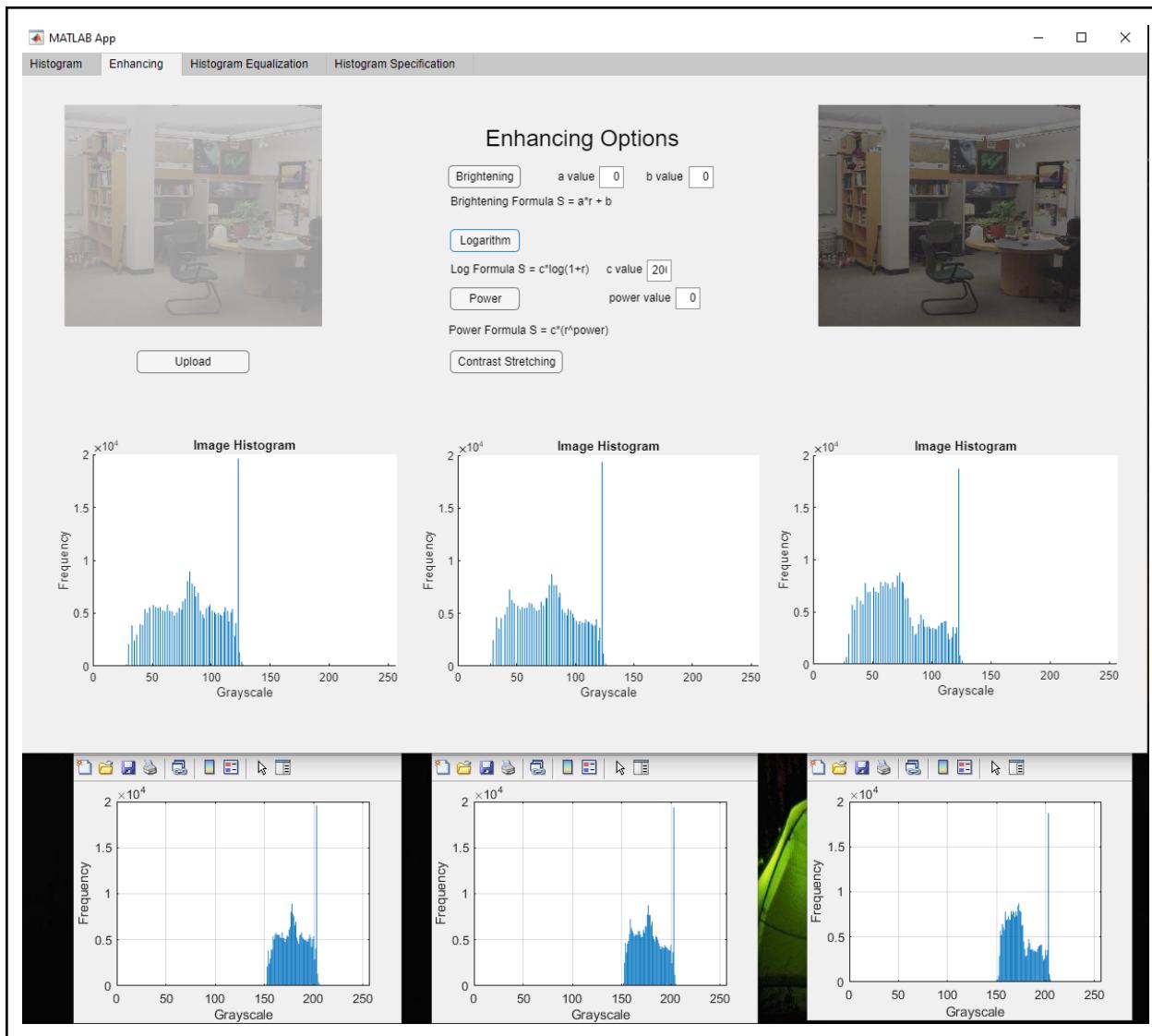


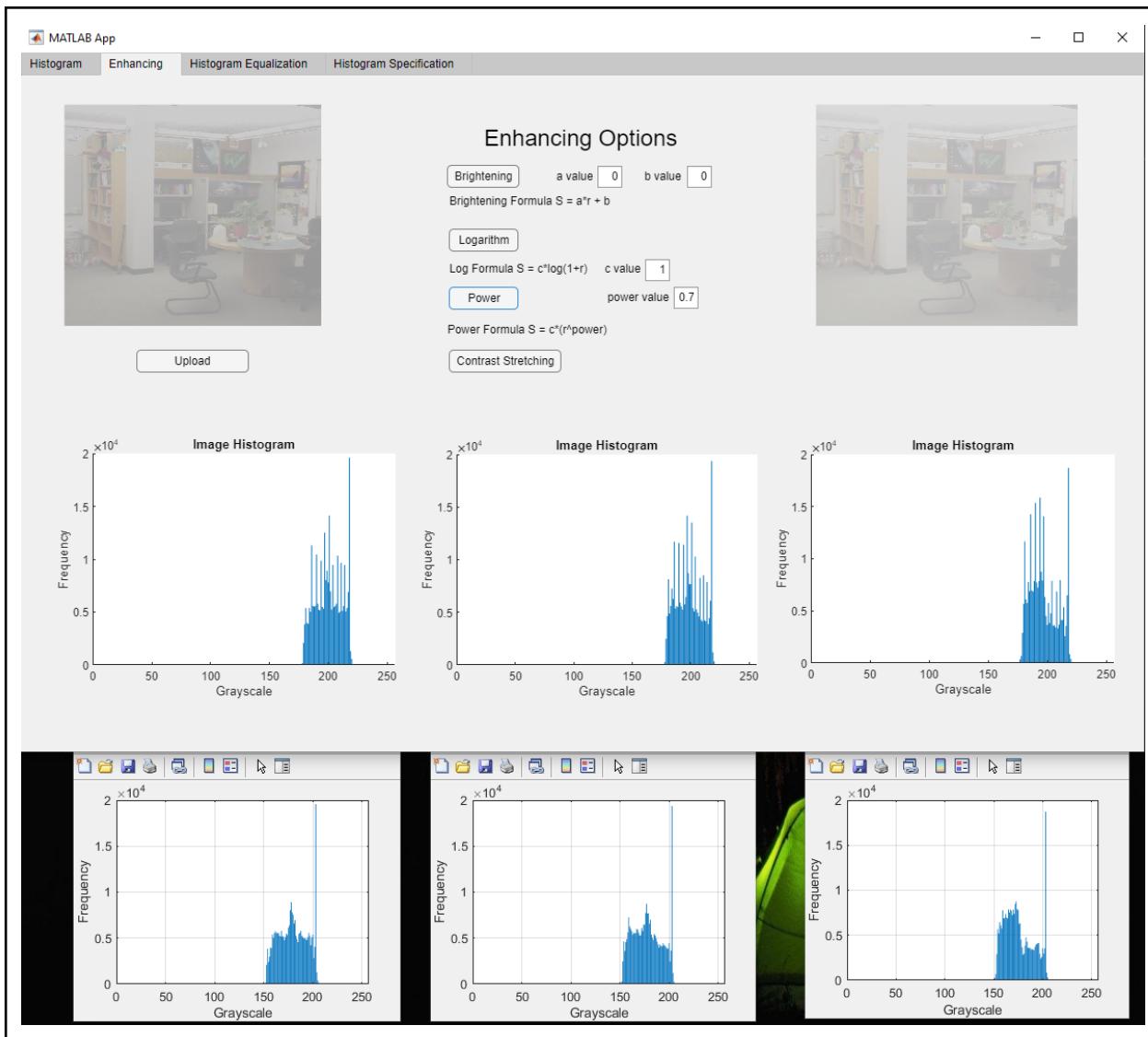


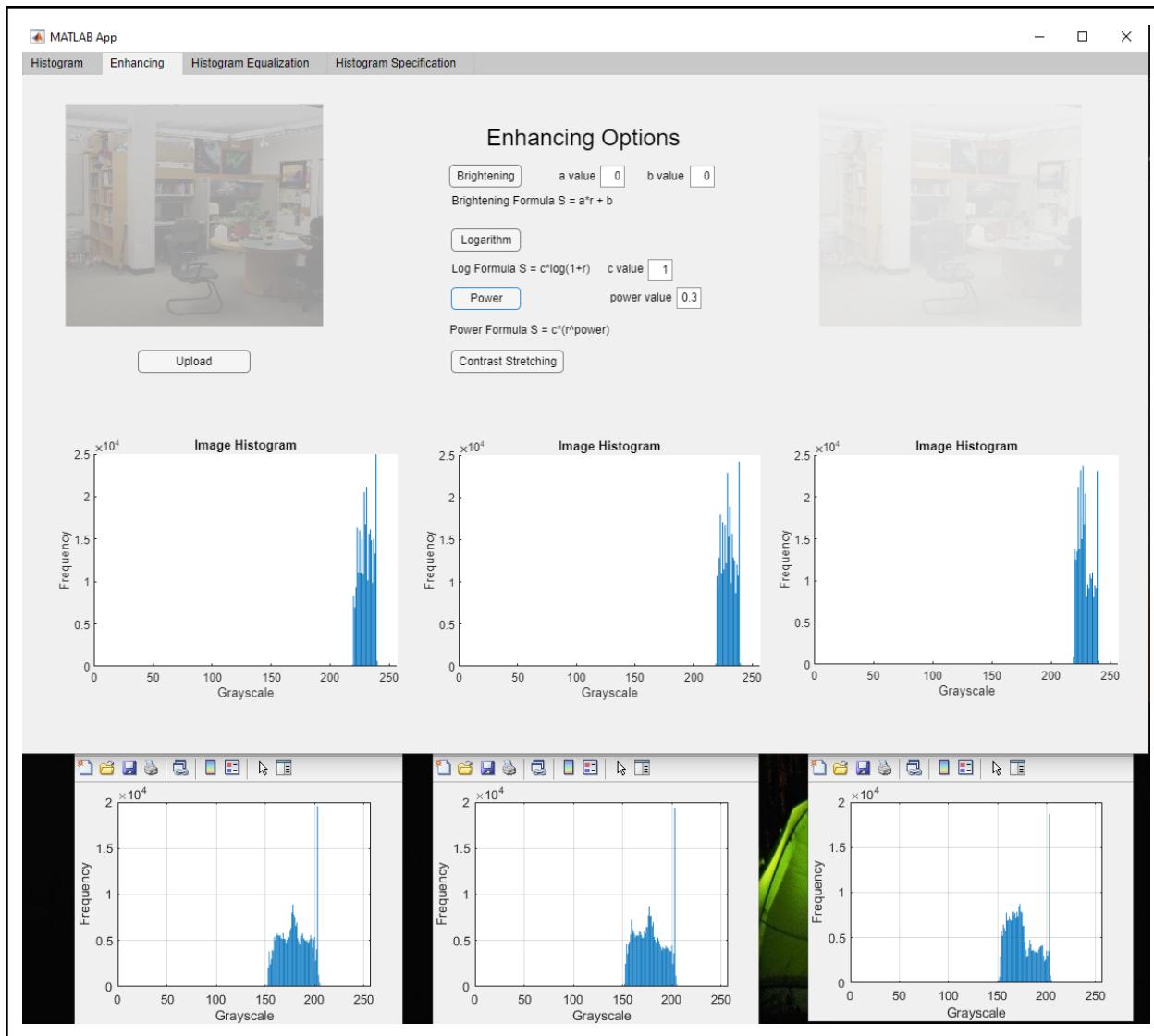


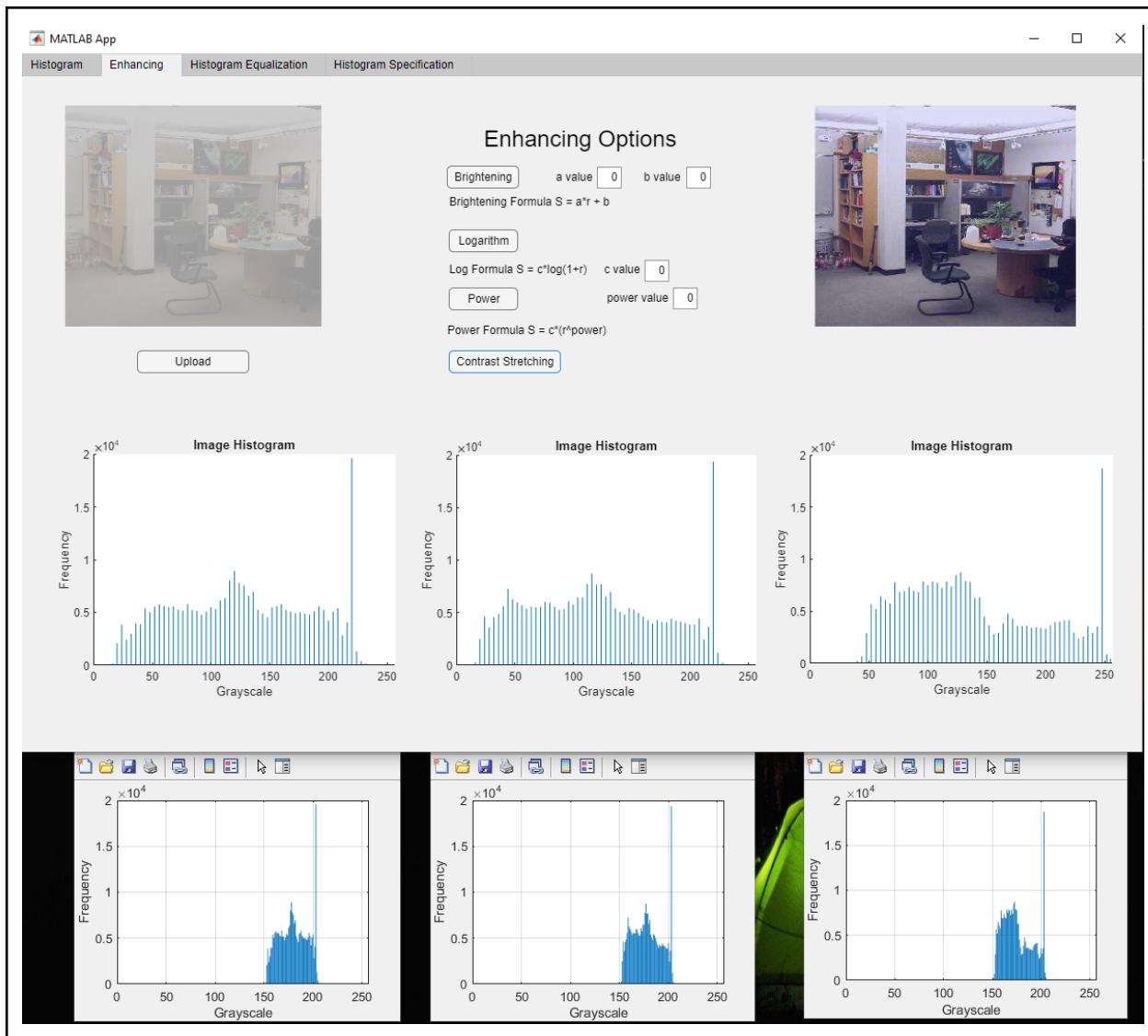


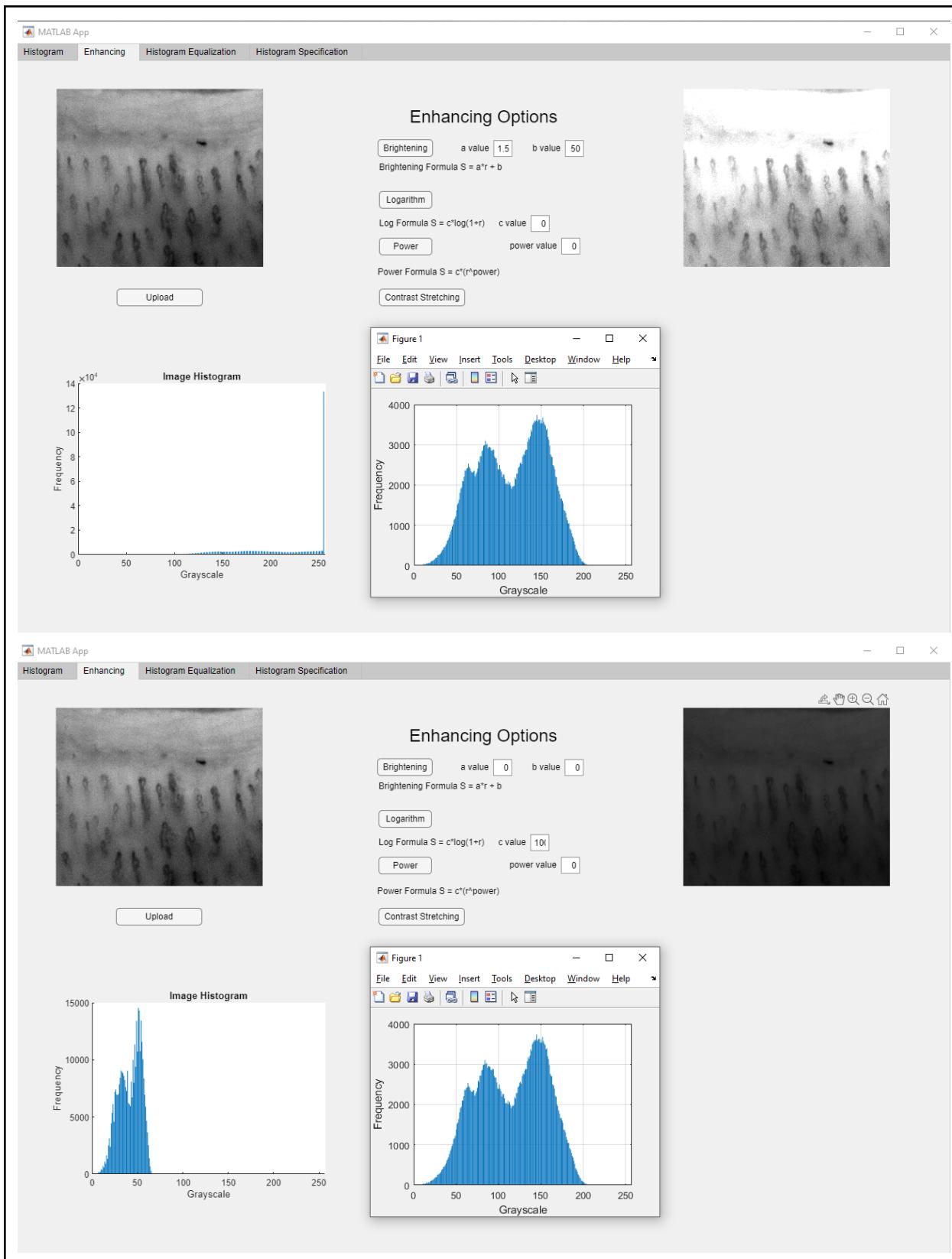


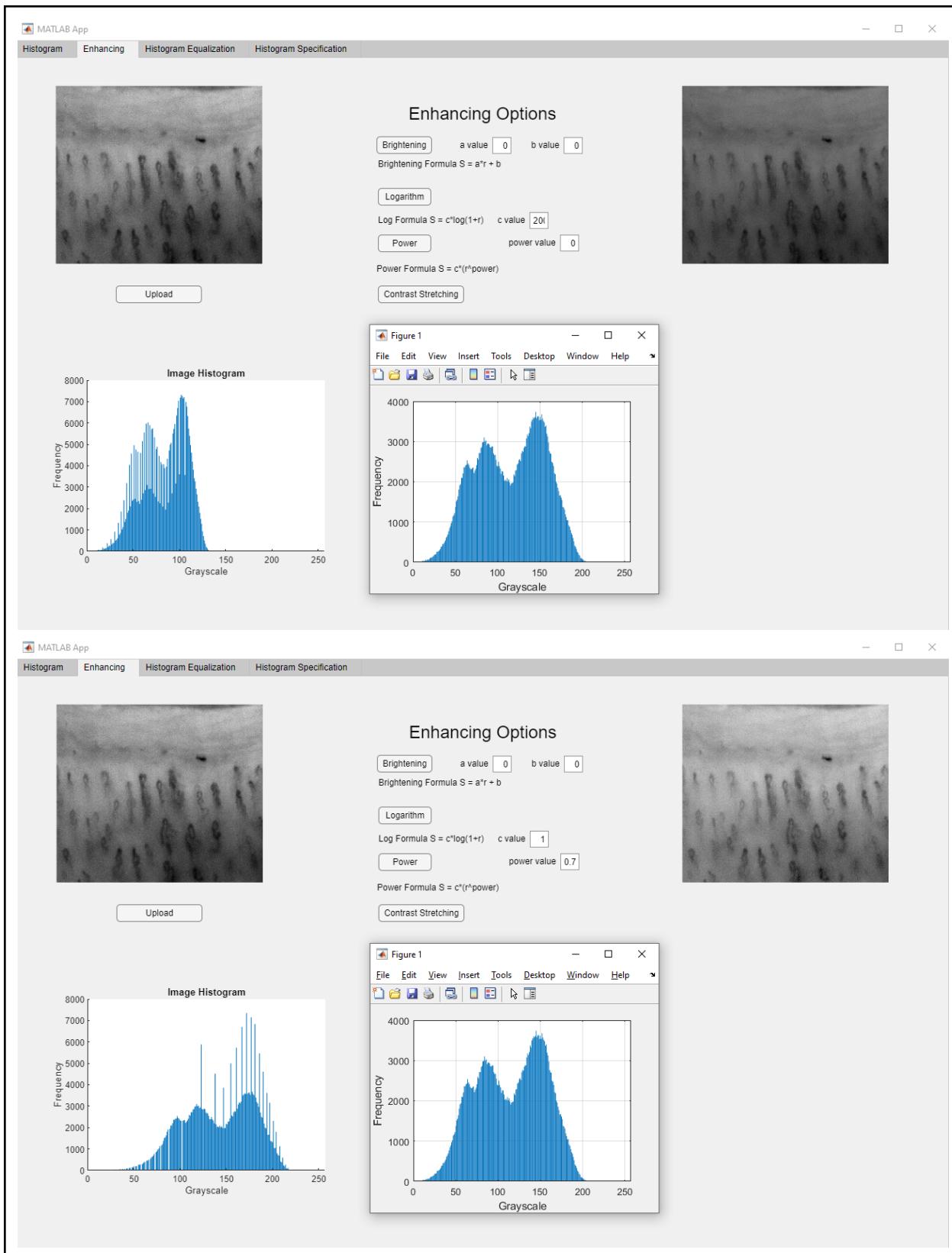


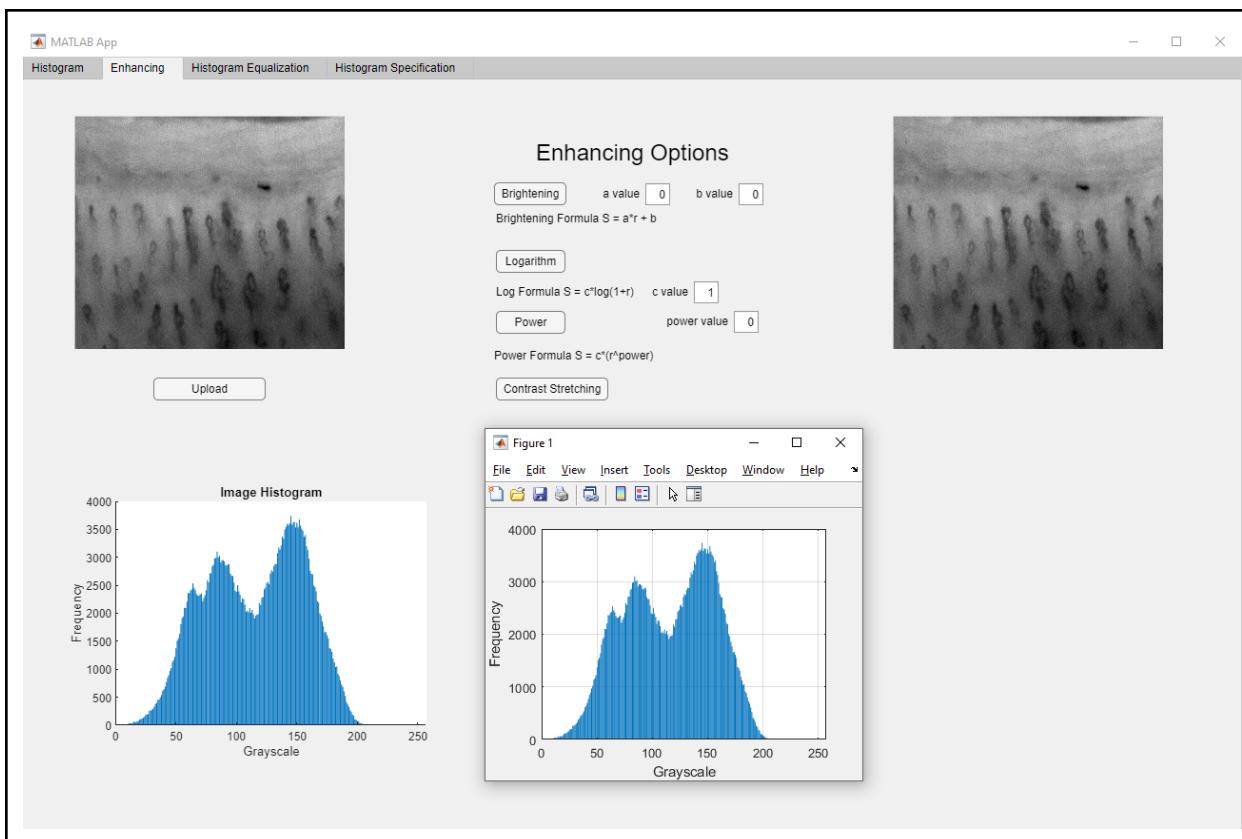












Pada berbagai contoh eksekusi kode program di GUI tersebut ada kecacatan yang dapat diidentifikasi dan dianalisis. Namun sebelum itu, perhatikan bahwa pada image brightening, transformasi pangkat, dan contrast stretching program dapat menghasilkan citra yang dapat dibilang memuaskan dan sudah cocok dengan sebagaimana seharusnya. Pada transformasi logaritma, dapat dilihat bahwa histogram yang diperoleh tidak benar-benar membentuk transformasi logaritma yang benar, setidaknya dari sisi aplikasi pengalian konstanta. Dari kode program dan eksekusinya menurut saya tingkat keterangan dari gambar sangat bergantung pada nilai konstanta yang tinggi adalah karena hasil aplikasi rumus logaritma pada tiap-tiap pixel menghasilkan nilai di range yang lumayan kecil. Sayangnya walau seperti itu, aplikasi berbagai solusi yang telah saya pikirkan seperti mengubah format penggunaan rumus atau mengalikan hasil operasi logaritma pada suatu angka terlebih dahulu sebelum dikalikan konstanta tidak membuat hasil. Maka dari itu, hasil akhir transformasi logaritma adalah nilai pixel-pixel individual yang lumayan kecil dan kemudian akan dikalikan dengan konstanta sehingga nilainya membesar hingga membentuk gambar yang terlihat.

Sebagai tambahan, urutan histogram adalah red - green - blue.

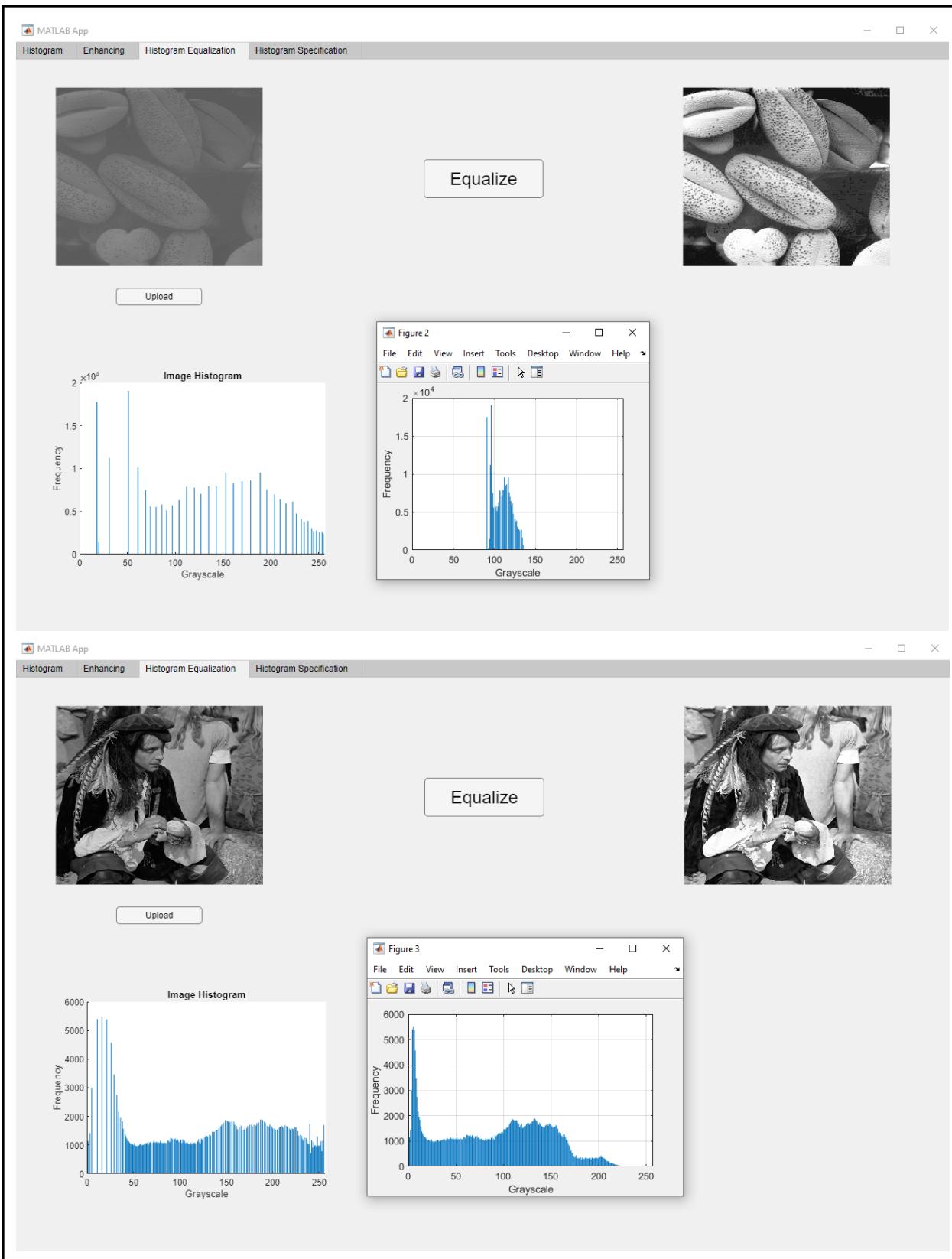
- c. Soal 3, melakukan histogram equalization dan menunjukkan hasilnya

Pada soal ketiga ini, dilakukan perataan histogram untuk memperoleh penyebaran histogram yang merata, sedemikian sehingga setiap derajat keabuan memiliki jumlah pixel yang relatif sama. Dalam kata lain, diusahakan frekuensi dari tiap-tiap derajat keabuan terlihat seimbang. Oleh karena itu, operasi ini dapat menambah kontras karena yang awalnya frekuensi derajat keabuan misalnya

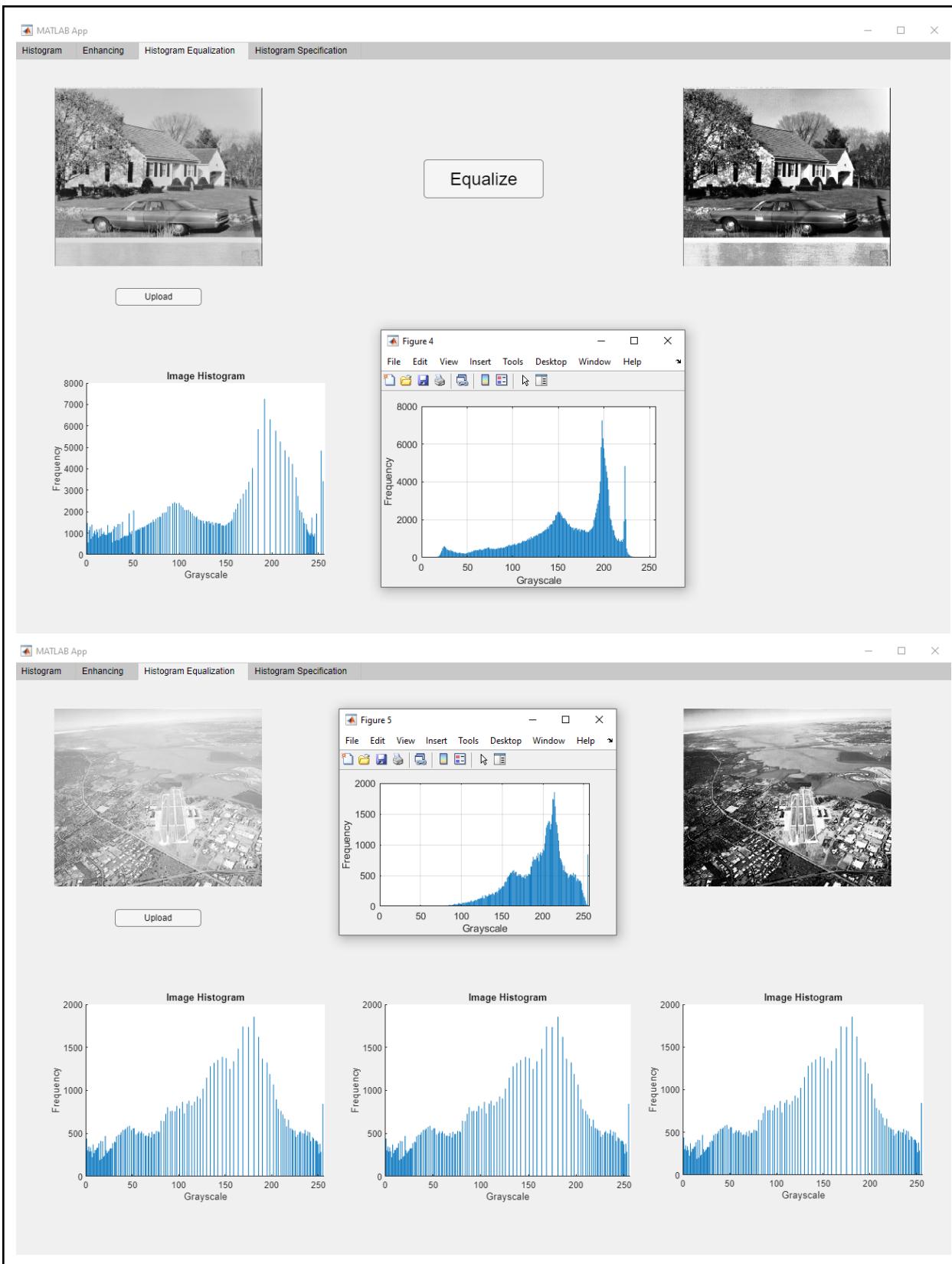
terlalu condong ke nilai yang kecil atau nilai yang besar akan diusahakan berubah menjadi setara antara jumlah anggota dengan derajat keabuan kecil dan jumlah anggota dengan derajat keabuan besar. Sama seperti sebelumnya untuk citra grayscale, histogram yang dihasilkan hanya 1 karena komponennya hanya 1 layer sedangkan untuk citra berwarna terdapat 3 layer yaitu red, green, dan blue. Berikut potongan kode program, contoh pengaplikasian pada GUI, dan analisis tambahan untuk soal kedua ini.

```
function newimg = equalizeHistogram(app, img, imghist, rows, cols, rgb)
    % secara umum modifikasi yang ada di PPT dengan variabel dan informasi yang dimiliki
    newxax = 1:256; % placeholder nilai derajat keabuan (x axis pada histogram) yang baru
    if (rgb>1)
        newxax(:,:,2) = 1:256;
        newxax(:,:,3) = 1:256;
    end
    for i = 1:rgb
        inc = 0;
        for j=1:256
            inc = inc + imghist(1,j,i);
        newxax(1,j,i) = (inc/(rows*cols))*256;
            % nilai derajat keabuan yang baru diperbarui dengan nilai perbandingan relatif
            % antara nilai derajat keabuan sekarang dengan jumlah seluruh pixel
        end
    end
    newxax = int64(newxax); % nilai derajat keabuan (x axis pada histogram) yang baru harus
    bulat
    newimg = img;
    for i = 1:rgb
        for c = 1:cols
            for r = 1:rows
                if(img(r,c,i) == 0)
                    continue
                end
                newimg(r,c,i) = newxax(1,img(r,c,i),i);
                % nilai derajat keabuan per pixelnya diganti dengan yang baru
                % ingat placeholder nilai derajat keabuan yang baru juga memiliki indeks
                sebanyak 256
                    % maka dari itu, dapat dibayangkan bahwa [indeks] merupakan derajat keabuan
                lama
                    % dan [elemen dari indeks] merupakan derajat keabuan yang baru
                end
            end
        end
    end
end
```

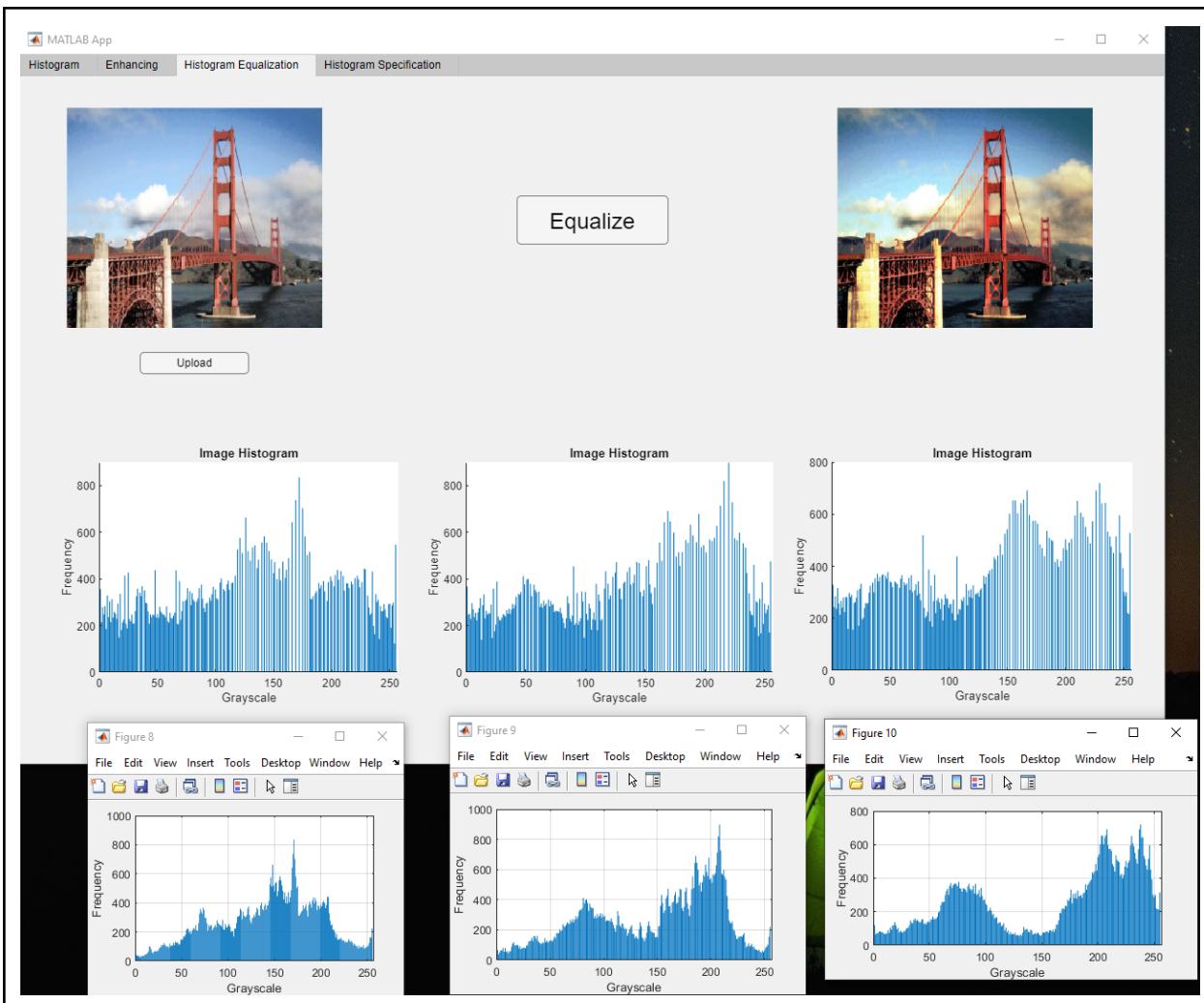
Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra

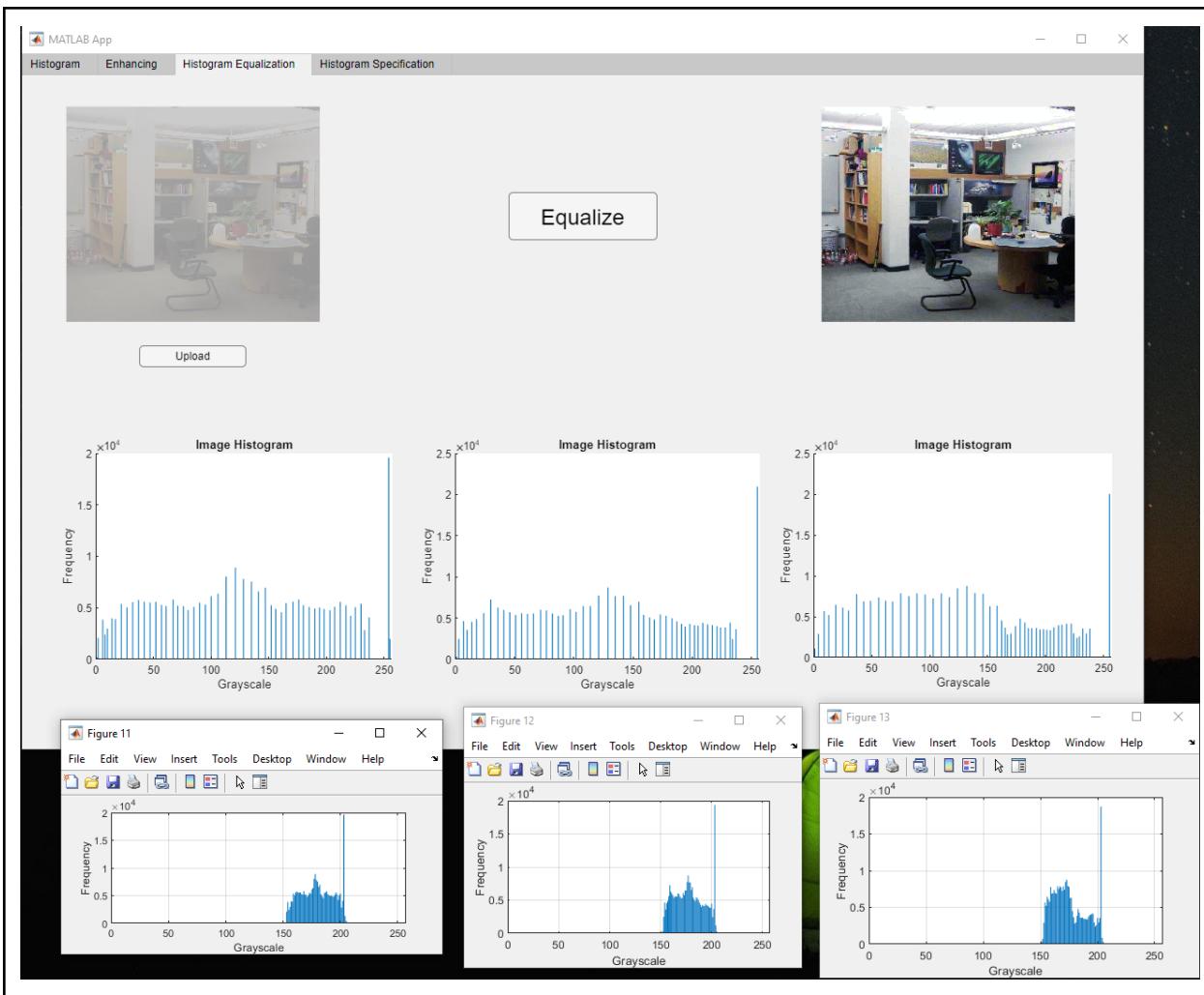


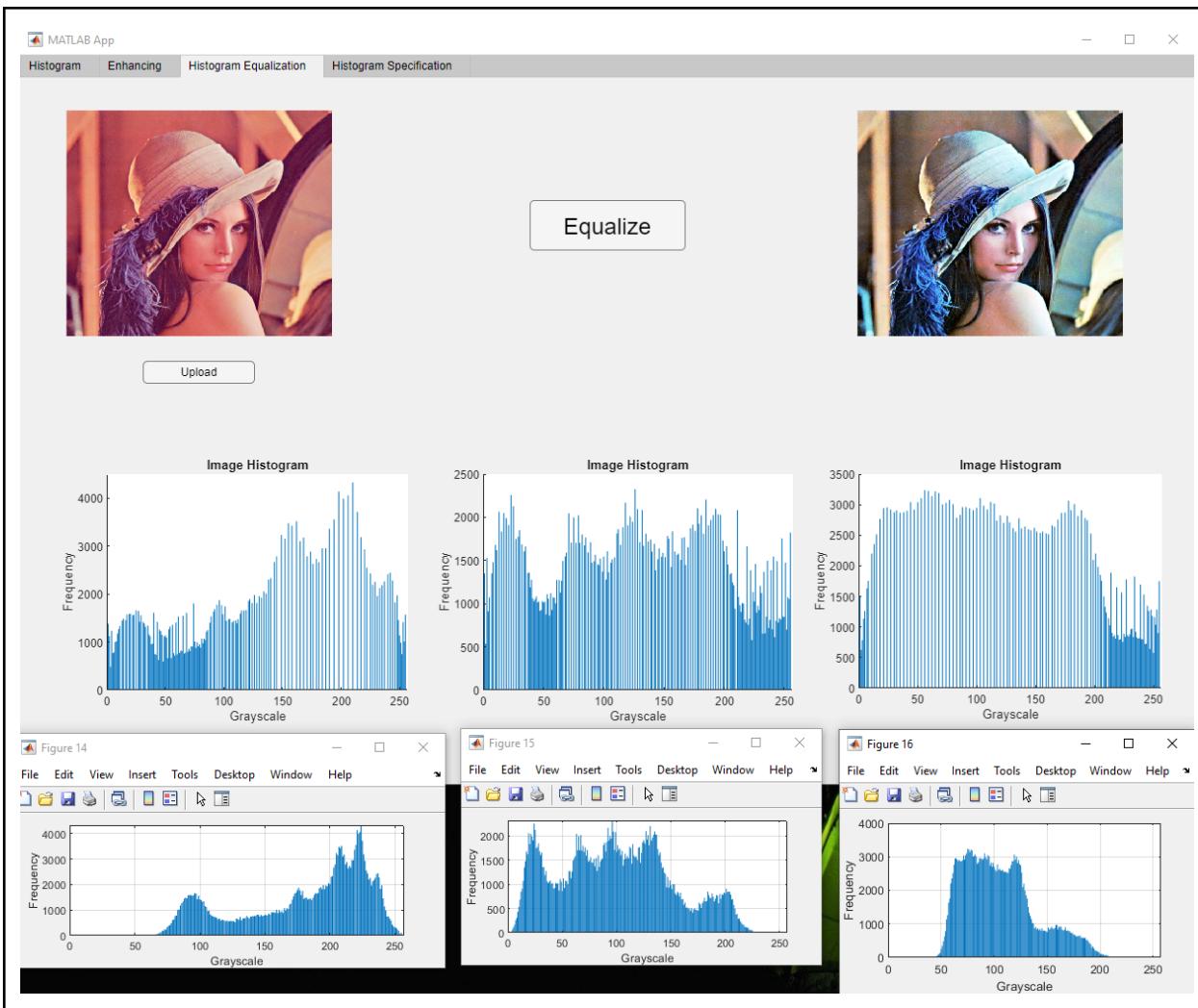
Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra

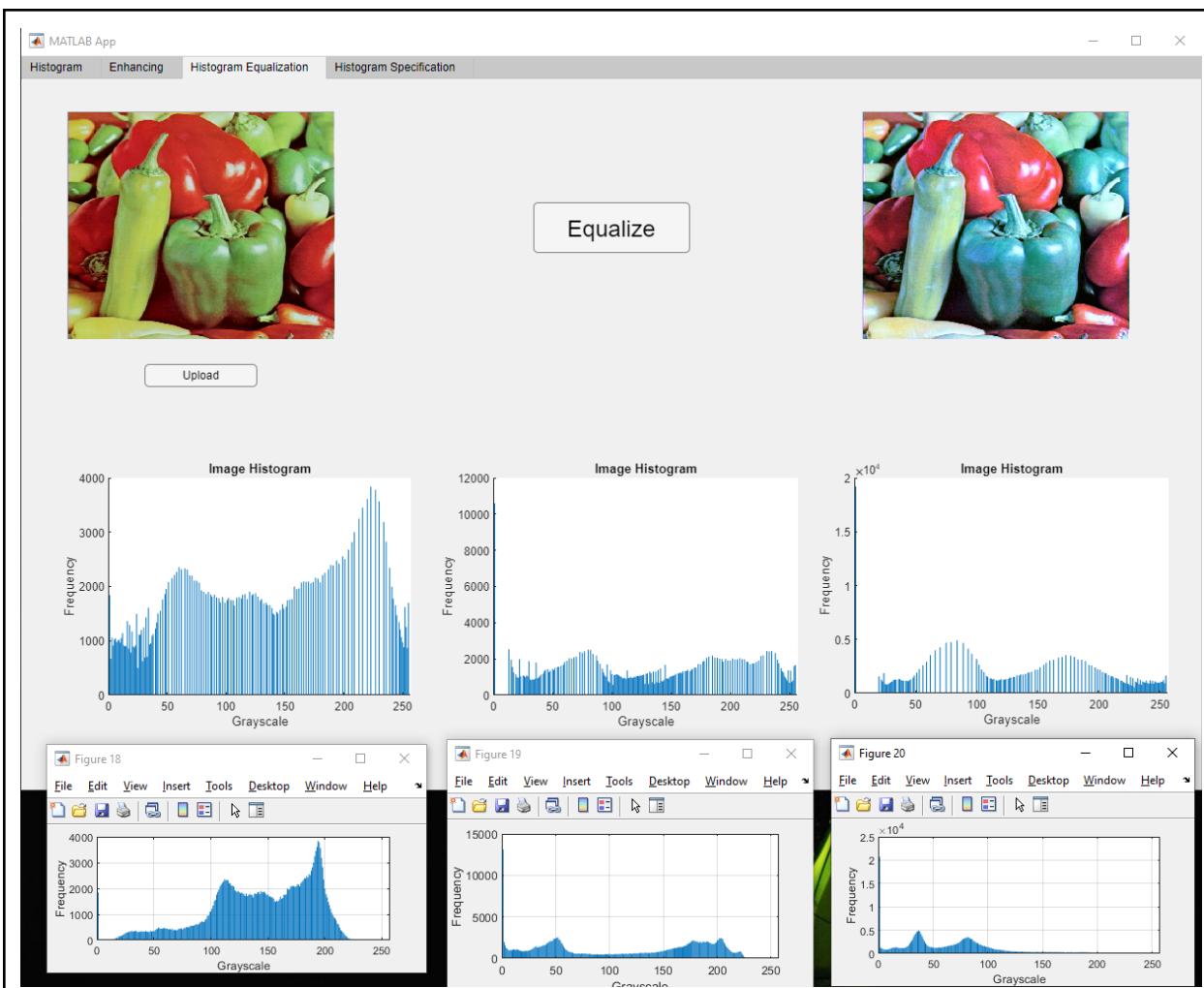


Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra









Pada berbagai contoh eksekusi kode program di GUI tersebut dapat dilihat bahwa aplikasi perataan histogram menunjukkan hasil yang memuaskan dan seharusnya sudah benar. Untuk analisis terkait kode programnya sendiri, mempertegas apa yang telah ada di komen-komen pada potongan kode, intinya adalah memetakan derajat keabuan yang lama menjadi derajat keabuan yang baru dengan memodifikasi menggunakan cara yang telah dijabarkan di perkuliahan dan terdapat di PPT.

Sebagai tambahan, urutan histogram adalah red - green - blue.

- d. Soal 4, melakukan histogram specification dan menunjukkan hasilnya

Pada soal keempat ini, dilakukan histogram specification atau pencocokan histogram untuk memperoleh histogram yang diinginkan pengguna. Dengan menggunakan gambar yang histogramnya digunakan sebagai acuan, program harapannya dapat mengubah atau memodifikasi histogram citra awal sedemikian sehingga histogram hasil modifikasi akan memiliki karakteristik yang mirip dengan histogram citra acuan. Sama seperti sebelumnya untuk citra grayscale, histogram yang dihasilkan hanya 1 karena komponennya hanya 1 layer sedangkan

untuk citra berwarna terdapat 3 layer yaitu red, green, dan blue. Berikut potongan kode program, contoh pengaplikasian pada GUI, dan analisis tambahan untuk soal kedua ini.

```
function newimg = specificationHistogram(app, img, imgspec, rows, cols, rgb)
    % secara umum modifikasi yang ada di PPT dengan variabel dan informasi yang dimiliki
    [xax, imghist] = app.procCHistogram(img,rows,cols,rgb); % bentuk histogram citra awal
    newxax = 1:256; % placeholder nilai derajat keabuan (x axis pada histogram) yang baru
untuk citra awal
if rgb>1
    newxax(:,:,2) = 1:256;
    newxax(:,:,3) = 1:256;
end
% lakukan perataan histogram
for layer=1:rgb
    inc = 0;
    for i=1:256
        inc = inc + imghist(1,i,layer);
        newxax(1,i,layer) = (inc/(rows*cols))*256;
    end
    newxax = int64(newxax);
    newimg = img;
end
newxax = int64(newxax);

[xax, imghist1] = app.procCHistogram(imgspec,rows,cols,rgb); % bentuk histogram citra acu
newxax1 = 1:256; % placeholder nilai derajat keabuan (x axis pada histogram) yang baru
untuk citra acu
if rgb>1
    newxax1(:,:,2) = 1:256;
    newxax1(:,:,3) = 1:256;
end
% lakukan perataan histogram
for layer=1:rgb
    inc = 0;
    for i=1:256
        inc = inc + imghist1(1,i,layer);
        newxax1(1,i,layer) = (inc/(rows*cols))*256;
    end
    newxax1 = int64(newxax1);
    newimg = img;
end

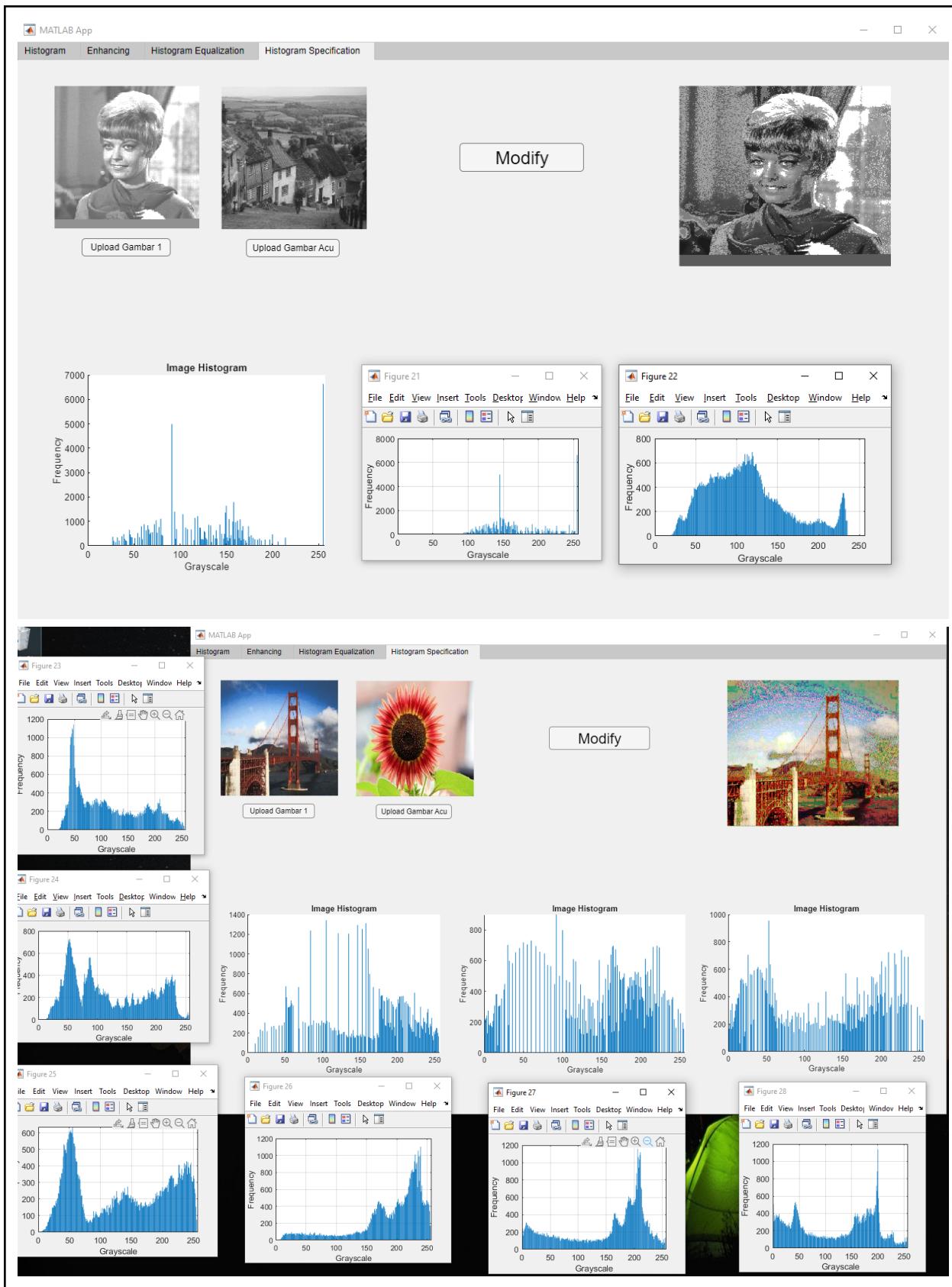
invxax = 1:256; % placeholder pemetaan nilai derajat keabuan (x axis pada histogram) dari
citra awal ke citra acu
if rgb>1
```

```
invxax(:,:,2) = 1:256;
invxax(:,:,3) = 1:256;
end

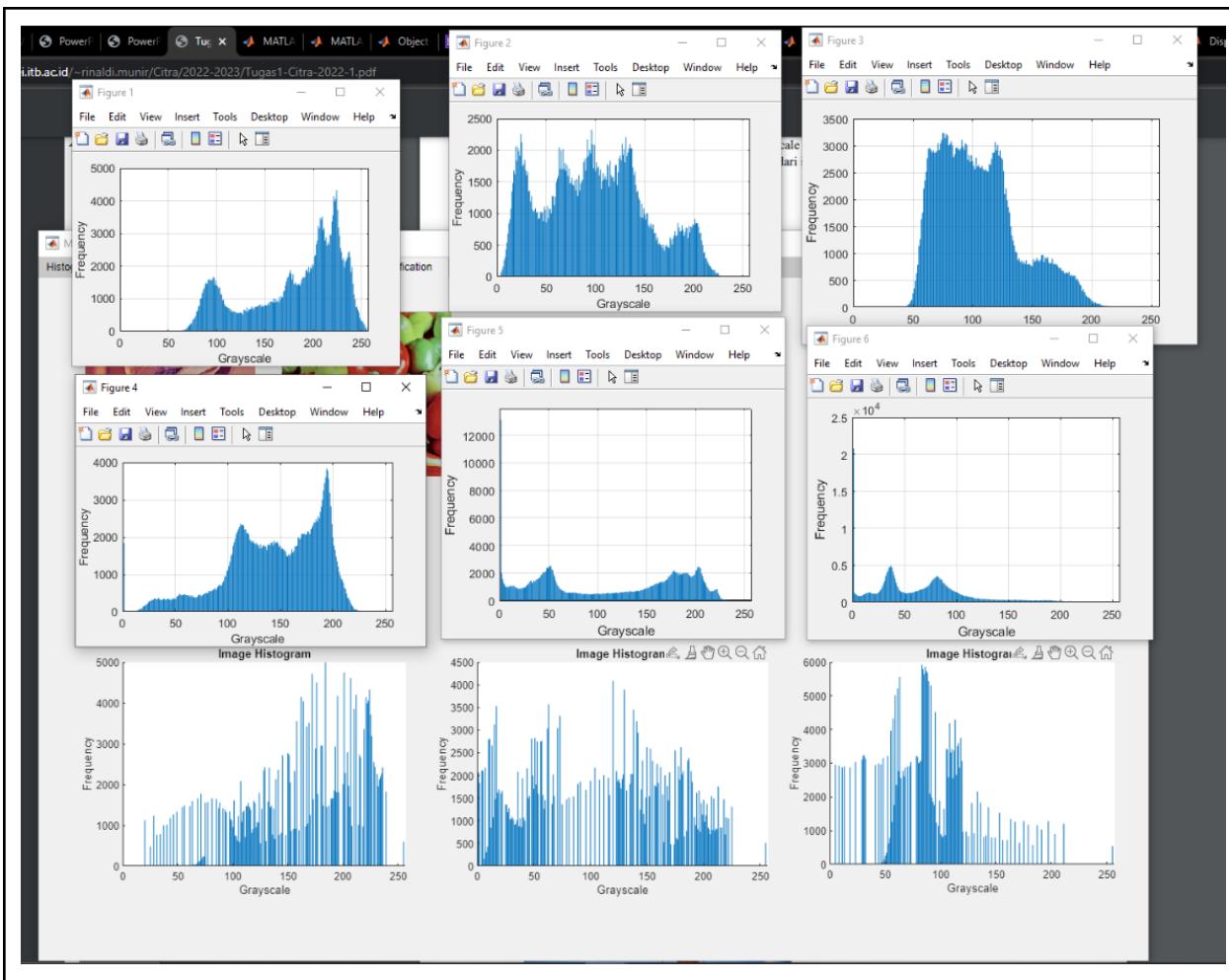
% pencocokan histogram menggunakan transformasi balikan
% xax adalah x axis [indeks derajat keabuan 1-256]
for layer=1:rgb
    for i=1:256
        for j=1:256
            % jika nilai derajat keabuannya antara elemen dari index xax citra awal dan
            % citra acu sama
            % maka nilai derajat keabuan untuk xax citra awal adalah indeks dari elemen
            % citra acu tersebut
            if newxax(1,i,layer) == newxax1(1,j,layer)
                invxax(1,i,layer) = j;
            end
        end
    end
end

for layer=1:rgb
    for c = 1:cols
        for r = 1:rows
            if img(r,c,layer) == 0
                continue
            end
            newimg(r,c,layer) = invxax(1,img(r,c,layer),layer);
            % nilai derajat keabuan per pixelnya diganti dengan yang baru
            % ingat placeholder nilai derajat keabuan yang baru juga memiliki indeks
            sebanyak 256
            % maka dari itu, dapat dibayangkan bahwa [indeks] merupakan derajat keabuan
            lama
            % dan [elemen dari indeks] merupakan derajat keabuan yang baru
        end
    end
end
end
```

Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra



Naufal Yahya Kurnianto 13519141
IF4073 Interpretasi dan Pengolahan Citra





perataan histogram menunjukkan hasil yang memuaskan dan seharusnya sudah benar menurut saya khususnya untuk komponen histogram. Walaupun begitu, berbagai citra yang dihasilkan terlihat menjadi lumayan rusak karena adanya suatu faktor. Faktor yang dapat saya analisis, untuk citra berwarna bridge dengan flower adalah jumlah frekuensi kemunculan nilai derajat keabuan yang lumayan kontras. Pada citra bridge, pixel-pixel mayoritas memiliki nilai derajat keabuan yang rendah dan sebaliknya untuk citra flower. Maka dari itu, dengan menggunakan algoritma kode yang saya terapkan, program mencoba membagi nilai-nilai derajat keabuan yang rendah itu untuk ditarik menuju derajat keabuan yang tinggi dengan distribusi yang bisa dibilang tidak normal. Terwujudlah citra yang lumayan rusak. Lain dari itu, untuk citra grayscale girl dengan goldhill, citra girl memiliki frekuensi nilai derajat keabuan yang agak stabil dengan pengecualian yaitu nilai derajat keabuan sangat tinggi yang frekuensinya juga sangat tinggi (outlier dari sisanya namun frekuensinya besar). Dengan program yang berusaha menggeser outlier tersebut untuk membentuk histogram yang relatif rata di tengah menghasilkan citra yang lumayan hancur karena kombinasi keabuan yang awalnya terlihat normal menjadi tergeser sehingga muncul kombinasi citra grayscale antar pixel yang tidak cocok. Untuk analisis kode programnya tersendiri, selain yang sudah ada pada komentar kode di atas, intinya adalah pemetaan derajat keabuan citra awal menjadi invers dari data derajat keabuan citra acuan sehingga membentuk pemetaan derajat keabuan yang baru.

Sebagai tambahan, urutan histogram adalah red - green - blue.

3. Github Repository

<https://github.com/ayahyaaa/pengcit/tree/main/tucil1>