Tugas 1 Pemrosesan Citra Digital

Disusun oleh:

Varraz Hazzandra Abrar	13521020
Akbar Al Fattah	13522036

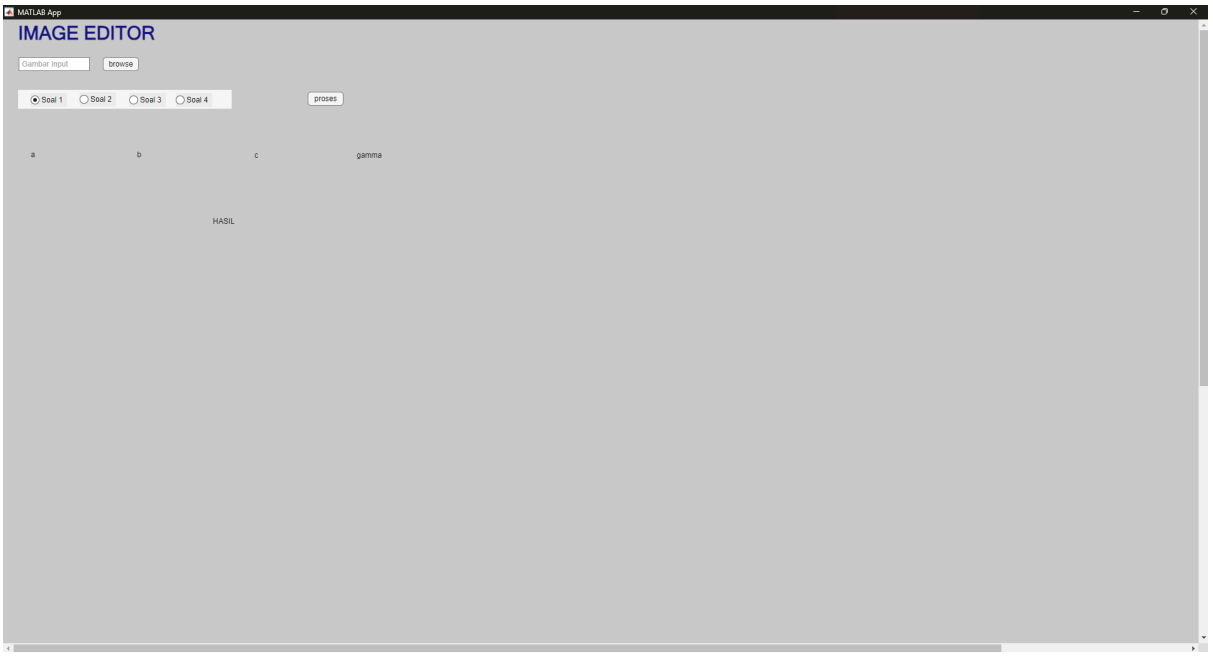
IF4073 PEMROSESAN CITRA DIGITAL

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2025

GUI PROGRAM


MATLAB App
IMAGE EDITOR
Gambar Input
browse
Soal 1
Soal 2
Soal 3
Soal 4
proses
a
b
c
gamma
HASIL

SOAL 1

KODE

a. Fungsi custom_image_histogram

```
% custom_image_histogram - fungsi untuk mengubah matriks gambar menjadi histogram
function counts = custom_image_histogram(img)
    % buat semua ke double
    img = double(img);

    % ubah ke [0,255] dari range [0,1]
    if max(img(:)) <= 1
        img = img * 255;
    end

    % clamp semua nilai agar pasti dalam rentang [0,255]
    img(img < 0) = 0;
    img(img > 255) = 255;

    % bulatkan nilai ke integer terdekat
    img = round(img);

    counts = zeros(256,1);

    % hitung histogram manual
    [rows, cols] = size(img);
    for r = 1:rows
        for c = 1:cols
            value = img(r,c);
            counts(value+1) = counts(value+1) + 1;
        end
    end
end
```

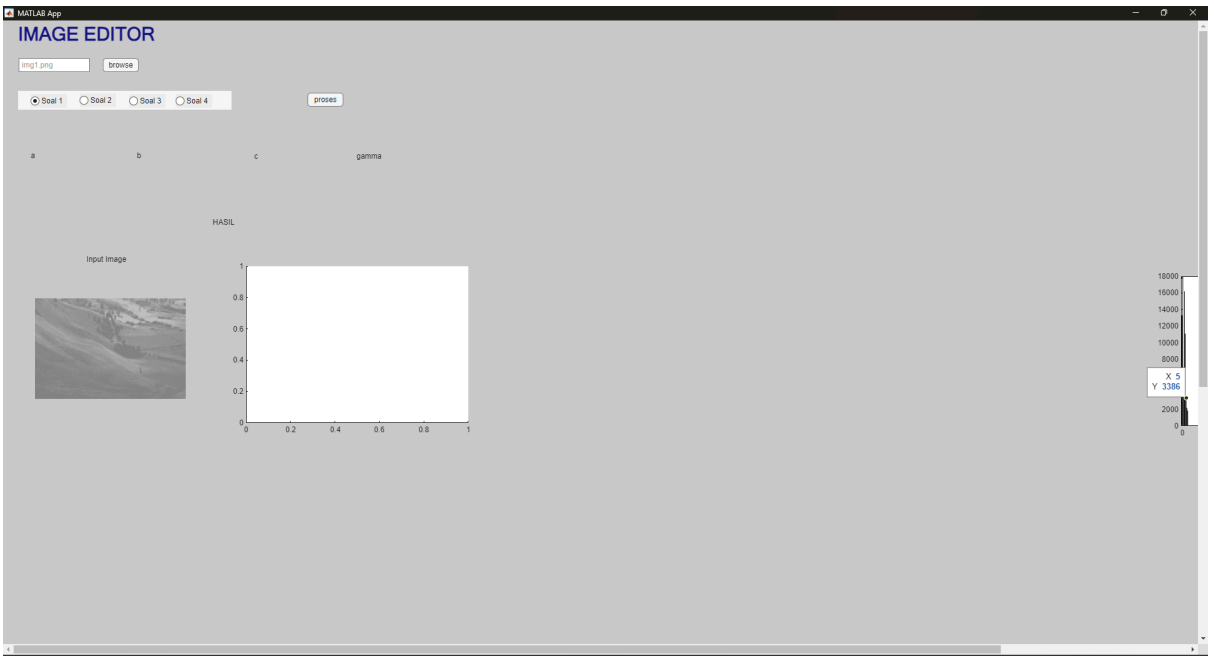
b. Fungsi Utama soal 1 (main_1_histogram.m)

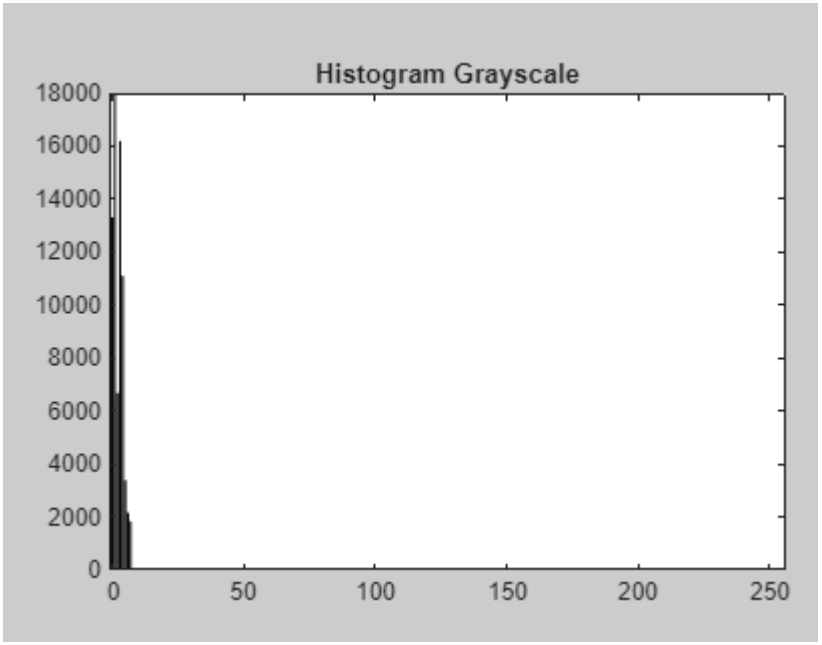
```
function main_1_histogram()
    img = browse_image();

    if isempty(img)
        return;
    end
    % tampilkan citra
    figure;
    imshow(img);
    title('Citra Asli');
    % cek apakah gambar grayscale atau berwarna

    if size(img, 3) == 1
        disp('Citra adalah gambar grayscale.');

        % hitung histogram grayscale (hanya sekali)
        counts = custom_image_histogram(img);
        bins = 0:255;

        %tampilkan histogram citra grayscale
        figure;
        bar(bins, counts, 'k');
        title('Histogram Citra Grayscale (fungsi histogram dibuat sendiri)');
        xlabel('Intensitas (0-255)');
        ylabel('Jumlah Piksel');
    else
        disp('Citra adalah gambar berwarna.');
        %pisah matriks gambar RGB yang terdiri atas 3 channel menjadi matriks R, G, dan B
        R = img(:,:,1);
        G = img(:,:,2);
        B = img(:,:,3);

        % Hitung histogram tiap channel R,G,B (3 kali)
        binsR = 0:255;
        binsG = 0:255;
        binsB = 0:255;
        countR = custom_image_histogram(R);
        countG = custom_image_histogram(G);
        countB = custom_image_histogram(B);

        % Tampilkan histogram RGB
        figure;
        bar(binsR, countR, 'r');
        title('Histogram Citra R/Merah (fungsi histogram dibuat sendiri)');
        xlabel('Intensitas (0-255)');
```

```
        ylabel('Jumlah Piksel');
        figure;
        bar(binsB, countB, 'b');
        title('Histogram Citra B/Biru (fungsi histogram dibuat sendiri)');
        xlabel('Intensitas (0-255)');
        ylabel('Jumlah Piksel');
    end
end
```

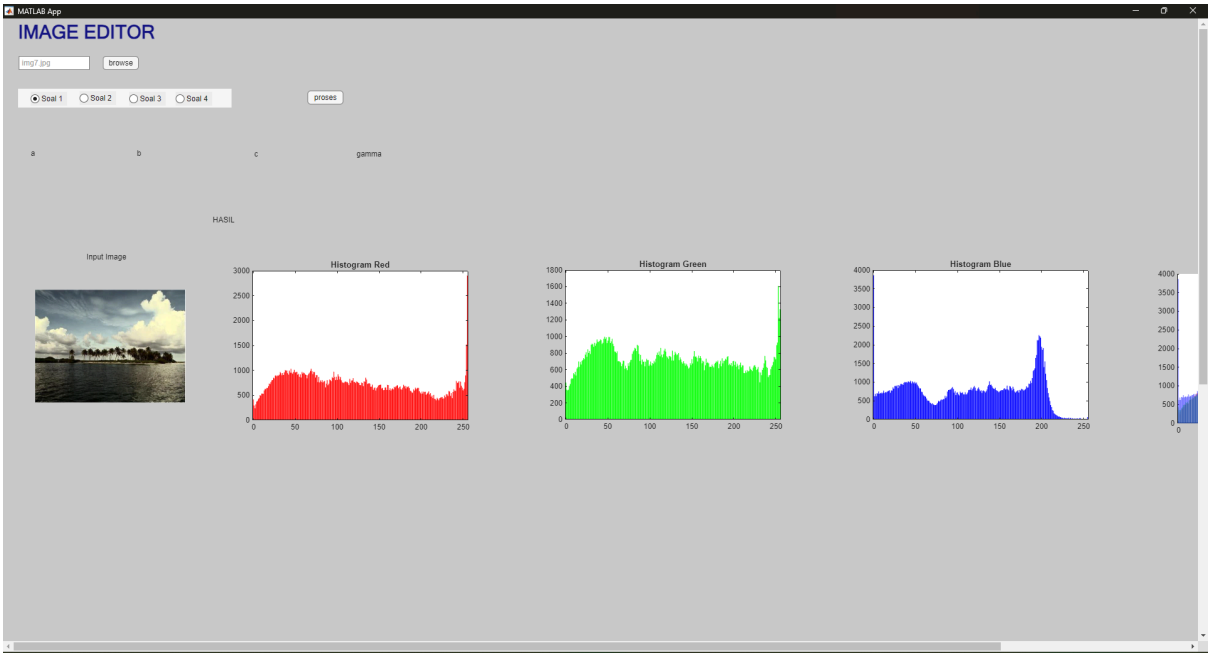
HASIL

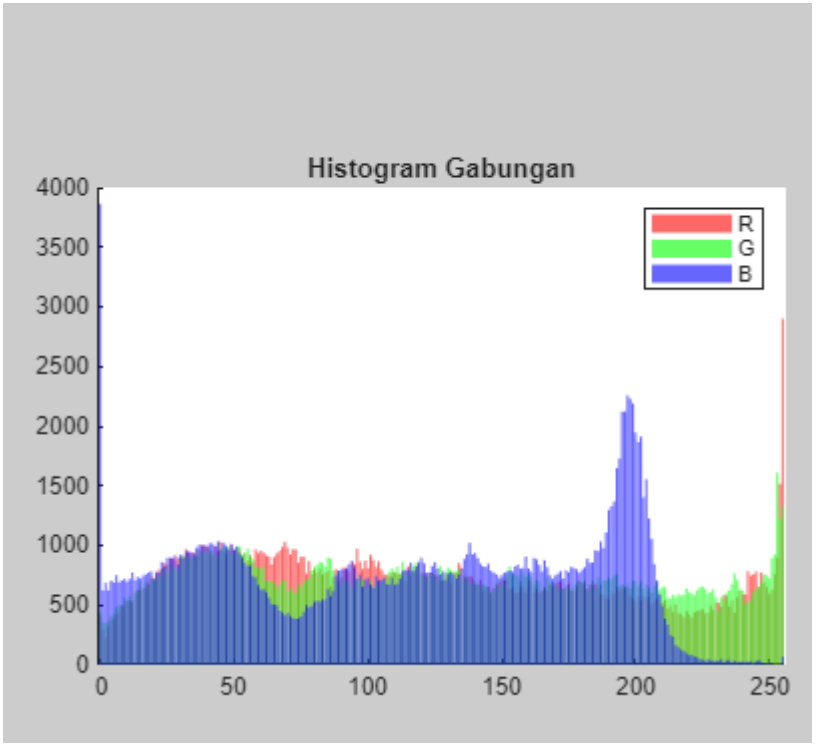
1. Gambar Grayscale





2. Gambar Berwarna





ANALISIS

Yang dilakukan oleh kode adalah menghitung kemunculan dari value x pada setiap piksel gambar. Namun, karena perbedaan tipe data value piksel di MATLAB, maka sebelum membuat histogram, tipe data setelah melakukan imread disamakan terlebih dahulu

SOAL 2

2A BRIGHTENING

KODE

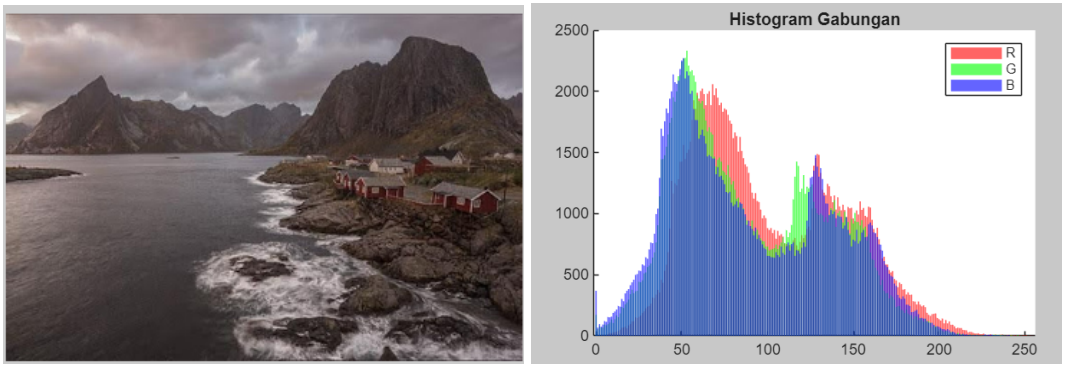
brigthener.m

```
function brightener(app, inputImgData, a, b)
   % Menghasilkan 2 output, jadi kita tampilkan di 2 set komponen output

   img = im2double(inputImgData);
   b_norm = b / 255.0;

   % Hitung kedua hasil
   outputB = img + b_norm;
   outputAB = a * img + b_norm;

   % Konversi kembali ke uint8 untuk ditampilkan
   outputImg_B = im2uint8(outputB);
   outputImg_AB = im2uint8(outputAB);

   % --- Update UI Output 1 ---
   app.outputImage.ImageSource = outputImg_B;
   app.outputImage.Visible = 'on';
   app.outputImageLabel.Visible = 'on';
   app.outputImageLabel.Text = 'Hasil (f + b)';
   app.plotFourHistograms(outputImg_B, 'outputHist');

  app.outputImage_2.ImageSource = outputImg_AB;
   app.outputImage_2.Visible = 'on';
   app.outputImageLabel_2.Visible = 'on';
   app.outputImageLabel_2.Text = 'Hasil (a*f + b)';
   app.plotFourHistograms(outputImg_AB, 'outputHist_2'); % Plot 4 histogram
untuk hasil 2
end
```

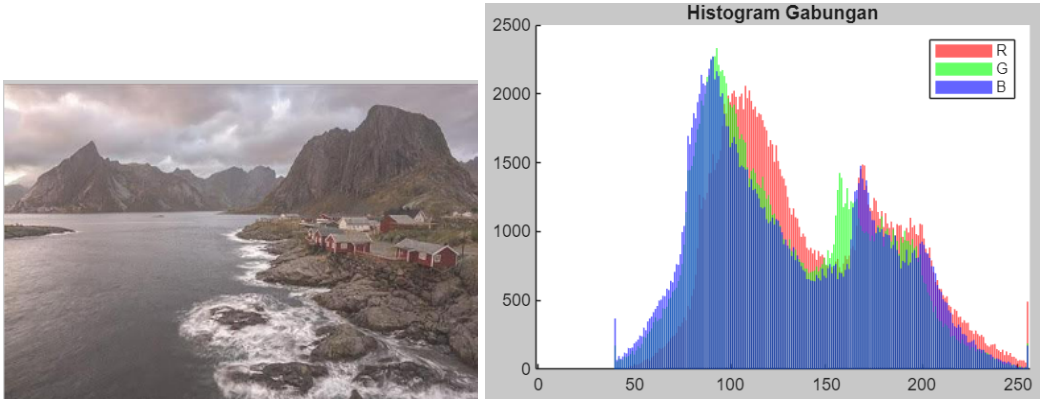
HASIL

Gambar Berwarna

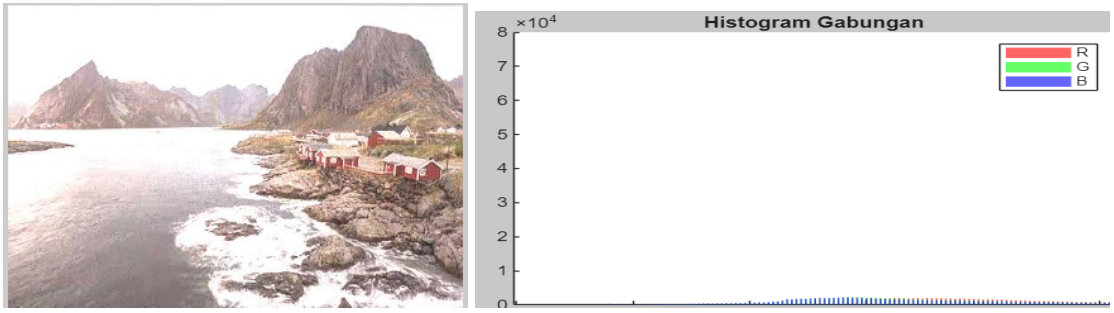
input



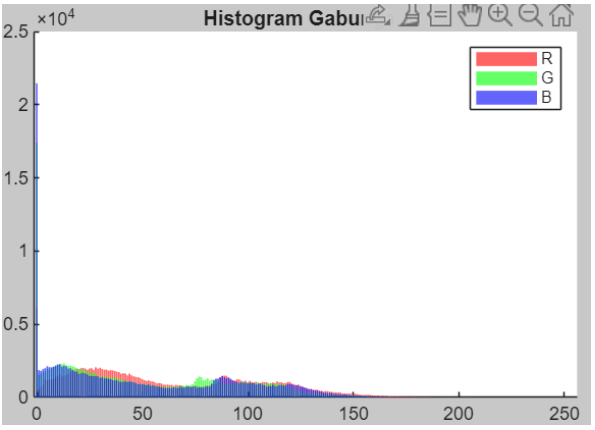
output img + 40



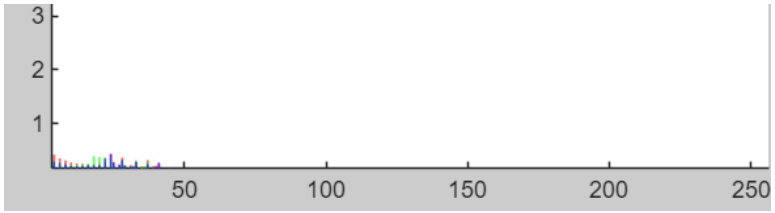
output 2*img + 40



output img - 40



output 0.5*img -40



Gambar Grayscale

ANALISIS

Pencerahan citra ini menerapkan transformasi linear pada nilai intensitas setiap piksel. Penambahan nilai positif b akan menggeser seluruh histogram ke kanan, menghasilkan gambar yang lebih cerah secara merata. Sebaliknya, nilai negatif b akan menggeser histogram ke kiri dan menggelapkan gambar. Penggunaan konstanta a memengaruhi kontras. Nilai a > 1 akan meregangkan rentang histogram sehingga perbedaan antara area gelap dan terang

semakin jelas (kontras meningkat), sementara a<1 akan memampatkan rentang histogram (kontras menurun).

2B NEGATIVE IMAGE

KODE

```
function negative(app, inputImg)
   % Konversi ke tipe double untuk perhitungan
   inputImg = im2double(inputImg) ;
   outputImg = 255- inputImg;
   reverseOutputImg = 255-outputImg;
   % [M, N, C] = size(inputImg);
   %
   % outputImg = zeros(M, N, C);
   %
   % for i = 1:M
   %     for j = 1:N
   %         for k = 1:C
   %             outputImg(i, j, k) = 255 - inputImg(i, j, k);
   %         end
   %     end
   % end
   %
   % % Reverse output (negatif dari negatif)
   % reverseOutputImg = zeros(M, N, C);
   %
   % for i = 1:M
   %     for j = 1:N
   %         for k = 1:C
   %             reverseOutputImg(i, j, k) = 255 - outputImg(i, j, k);
   %         end
   %     end
   % end
       % --- Update UI Output 1 ---
   app.outputImage.ImageSource = outputImg;
   app.outputImage.Visible = 'on';
   app.outputImageLabel.Visible = 'on';
   app.outputImageLabel.Text = 'Citra Negatif';
   app.plotFourHistograms(outputImg, 'outputHist');
   % --- Update UI Output 2 ---
   app.outputImage_2.ImageSource = reverseOutputImg;
   app.outputImage_2.Visible = 'on';
   app.outputImageLabel_2.Visible = 'on';
   app.outputImageLabel_2.Text = 'Negatif dari Negatif (Asli)';
   app.plotFourHistograms(reverseOutputImg, 'outputHist_2');
   %
   % % Simpan sementara
   % imwrite(uint8(outputImg), 'temp.png');
   % imwrite(uint8(reverseOutputImg), 'tempreverse.png');
   % % Tampilkan gambar dan histogram
   % app.outputImage.ImageSource = 'temp.png';
```

```
    % plot(app.outputHist, imhist(uint8(outputImg)));
    %
    % app.outputImage2.ImageSource = 'tempreverse.png';
    % plot(app.outputHist2, imhist(uint8(reverseOutputImg)));
    %
    % % Tampilkan semua komponen
    % app.outputHist.Visible = 'on';
    % app.outputImage.Visible = 'on';
    % app.outputImageLabel.Visible = 'on';
    %
    % app.outputHist2.Visible = 'on';
    % app.outputImage2.Visible = 'on';
    % app.outputImageLabel2.Visible = 'on';
end
```

HASIL

Gambar Berwarna

Gambar Grayscale

ANALISIS

Transformasi citra negatif bekerja dengan membalik nilai intensitas setiap piksel. Piksel yang tadinya gelap (mendekati 0) menjadi terang (mendekati 255), begitu juga sebaliknya. Pada citra berwarna, ini akan menghasilkan warna komplementer (misalnya, merah menjadi cyan). Teknik ini sangat berguna untuk menonjolkan detail berwarna putih atau abu-abu yang berada di area gelap pada gambar asli.

2C TRANSFORMASI LOG

KODE

```
function outputImg = logTransform (app,inputImg, c)
    img= double(inputImg);
    hasil = c * log(1 + img);
    outputImg = im2uint8(hasil);
```

```
        % Simpan sementara
    imwrite(uint8(outputImg), 'temp.png');
    % Tampilkan gambar dan histogram
    app.outputImage.ImageSource = 'temp.png';
    app.outputImageLabel.Text = 'Hasil Transformasi Log';
    app.plotFourHistograms(outputImg, 'outputHist_2');
    app.outputImage.Visible = 'on';
    app.outputImageLabel.Visible = 'on';
        % Sembunyikan set output kedua
    app.outputImage_2.Visible = 'off';
    app.outputImageLabel_2.Visible = 'off';
    app.outputHist_2G.Visible = 'off';
    app.outputHist_2B.Visible = 'off';
    app.outputHist_2R.Visible = 'off';
    app.outputHist_2Gabungan.Visible = 'off';


end
```

HASIL

Gambar Berwarna

Gambar Grayscale

ANALISIS

Transformasi logaritmik memetakan rentang nilai intensitas rendah (piksel gelap) pada citra input ke rentang yang lebih lebar pada citra output. Efeknya, detail-detail pada area gelap menjadi lebih terlihat jelas dan cerah. Sebaliknya, area yang sudah terang akan sedikit terkompresi. Transformasi ini sangat efektif untuk gambar yang terlalu gelap (*underexposed*).

2D TRANSFORMASI PANGKAT

KODE

```
function outputImg = powerTransform(app,inputImg, c, gamma)
```

```
    img = double(inputImg);
    hasil = c * (img .^ gamma);
    outputImg = im2uint8(hasil);

    % --- Update UI ---
    app.outputImage.ImageSource = outputImg;
    app.outputImage.Visible = 'on';
    app.outputImageLabel.Visible = 'on';
    app.outputImageLabel.Text = 'Hasil Transformasi Pangkat';
    app.plotFourHistograms(outputImg, 'outputHist');
    app.outputImage_2.Visible = 'off';
    app.outputImageLabel_2.Visible = 'off';
    app.outputHist_2G.Visible = 'off';
    app.outputHist_2B.Visible = 'off';
    app.outputHist_2R.Visible = 'off';
    app.outputHist_2Gabungan.Visible = 'off';
end
```

HASIL

Gambar Berwarna

Gambar Grayscale

ANALISIS

Transformasi pangkat memberikan fleksibilitas lebih dalam mengatur kecerahan citra. Jika nilai gamma lebih kecil daripada 1, kurva transformasi akan melengkung ke atas, menghasilkan efek pencerahan yang mirip dengan transformasi log, efektif untuk meningkatkan detail di area gelap. Jika lebih besar daripada 1, kurva akan melengkung ke bawah, menghasilkan efek penggelapan yang berguna untuk meredam area yang terlalu terang (*overexposed*) pada gambar.

2E PEREGANGAN CITRA

KODE

```
function contrastStrecth(app, inputImg)
    if size(inputImg, 3) == 1
        img_double = double(inputImg);
        r_min = min(img_double(:));
        r_max = max(img_double(:));
        if r_min == r_max, outputImg = inputImg; return; end
        result = (img_double - r_min) / (r_max - r_min);
        outputImg = im2uint8(result);
    else
        hsv = rgb2hsv(inputImg); % Perbaiki: langsung gunakan data gambar
        V = hsv(:, :, 3);
        v_min = min(V(:));
        v_max = max(V(:));
        if v_min == v_max, outputImg = inputImg; return; end
        V_stretched = (V - v_min) / (v_max - v_min);
        hsv_baru = cat(3, hsv(:,:,1), hsv(:,:,2), V_stretched);
        outputImg = im2uint8(hsv2rgb(hsv_baru));
    end

    % --- Update UI ---

    app.outputImage.ImageSource = outputImg;
    app.outputImage.Visible = 'on';
    app.outputImageLabel.Visible = 'on';
    app.outputImageLabel.Text = 'Hasil Peregangan Kontras';
    app.plotFourHistograms(outputImg, 'outputHist');
    % Sembunyikan set output kedua jika tidak dipakai
    app.outputImage_2.Visible = 'off';
    app.outputImageLabel_2.Visible = 'off';
    app.outputHist_2G.Visible = 'off';
    app.outputHist_2B.Visible = 'off';
    app.outputHist_2R.Visible = 'off';
    app.outputHist_2Gabungan.Visible = 'off';
end
```

HASIL

Gambar Berwarna

Gambar Grayscale

ANALISIS

Teknik peregangan citra meningkatkan kontras citra dengan "meregangkan" rentang nilai intensitas piksel yang ada agar memenuhi rentang dinamis penuh (misalnya, dari [50, 150] menjadi [0, 255]). Ini adalah transformasi linear per-bagian (*piecewise linear*) yang sangat efektif untuk citra yang terlihat "pucat" atau "berkabut", di mana nilai-nilai pikselnya

terkonsentrasi di area sempit pada histogram. Hasilnya adalah gambar dengan perbedaan yang lebih jelas antara bagian tergelap dan bagian paling terang.

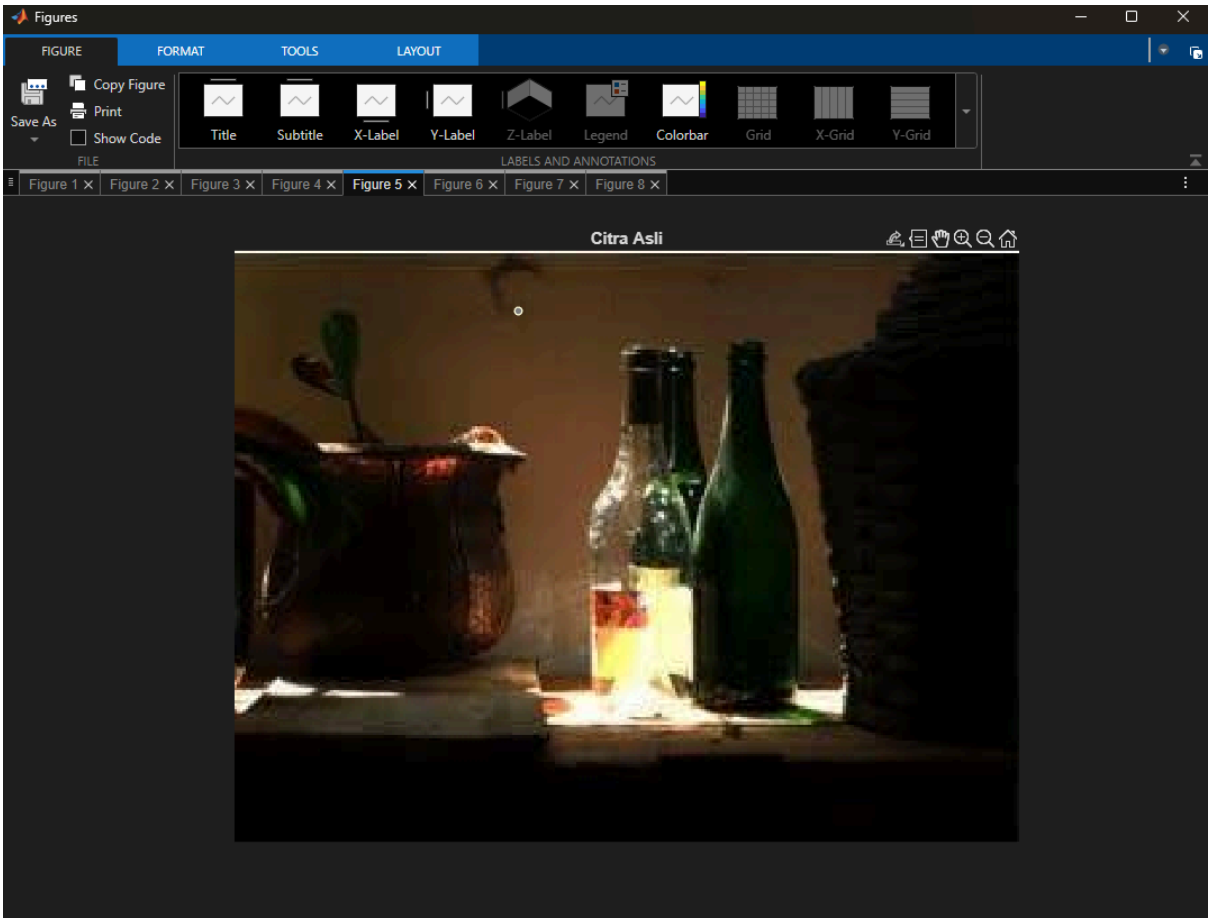
SOAL 3

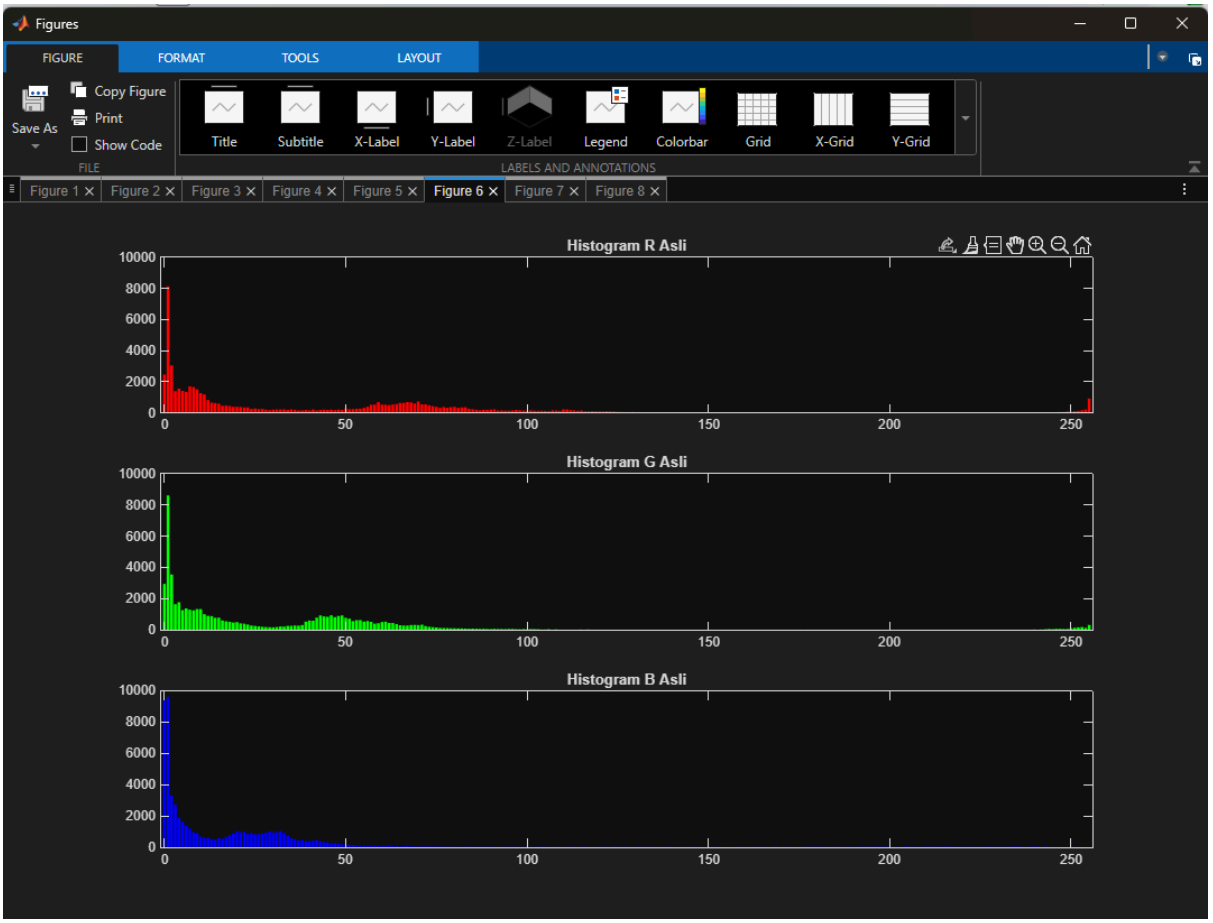
KODE

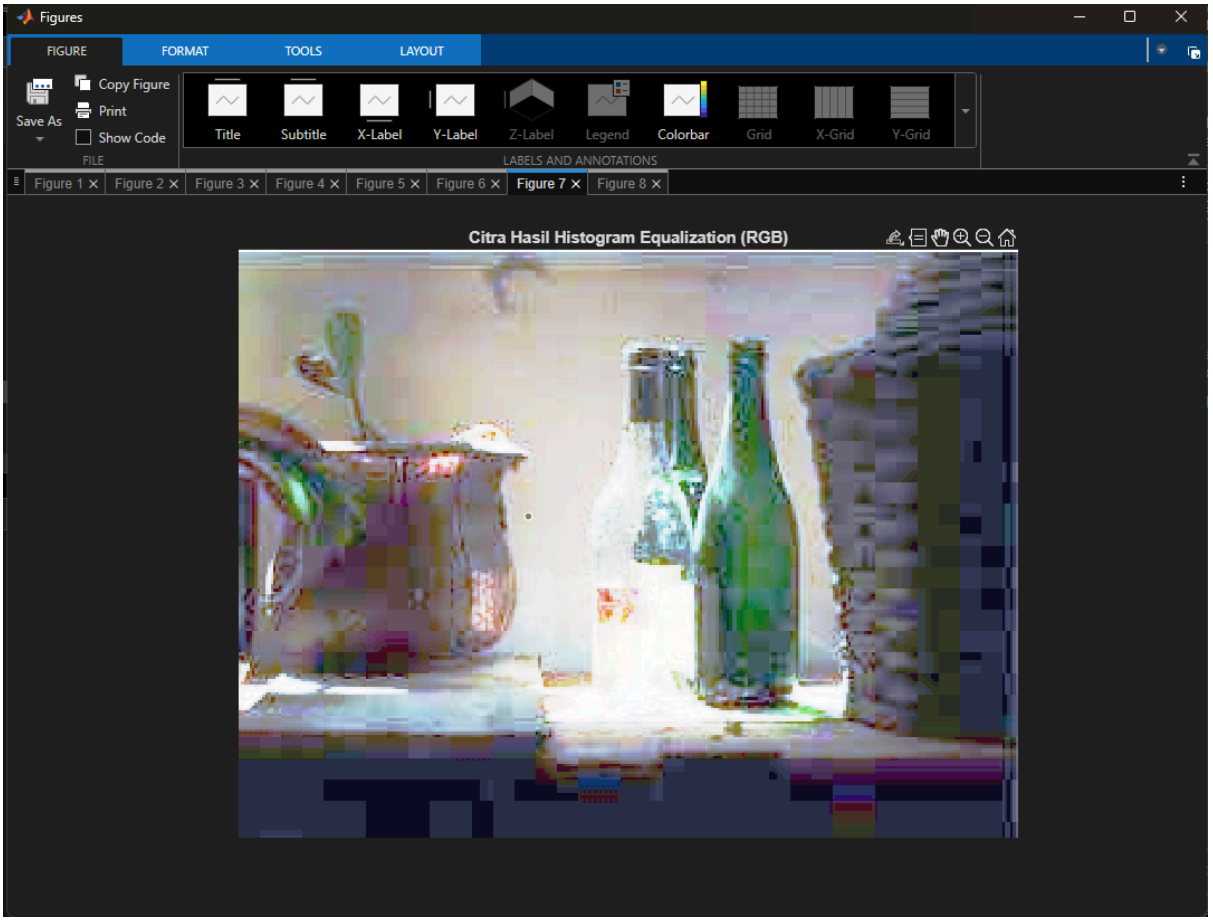
```
function eq_img = custom_histogram_equalization2(img, counts)
    % ubah ke double untuk perhitungan
    img = double(img);
    % total pixel
    total_pixels = numel(img);
    % hitung PDF (probability density function)
    pdf = counts / total_pixels;
    % hitung CDF (cumulative distribution function)
    cdf = cumsum(pdf);
    % normalisasi CDF ke rentang 0-255 (TANPA mengurangi cdf_min)
    cdf_scaled = round(cdf * 255);
    % buat citra baru berdasarkan nilai CDF
    eq_img = img;
    [rows, cols] = size(img);
    for r = 1:rows
        for c = 1:cols
            old_val = img(r,c);
            eq_img(r,c) = cdf_scaled(old_val+1);
        end
    end
function main_3_histogram_equalization()
    img = browse_image();
    if isempty(img)
        return;
    end
    figure;
    imshow(img);
    title('Citra Asli');
    if size(img, 3) == 1
        disp('Citra adalah gambar grayscale.');
        % hitung histogram citra asli (pakai fungsi buatan)
        counts = custom_image_histogram(img);
        bins = 0:255;
        % tampilkan histogram asli
        figure;
        bar(bins, counts, 'k');
        title('Histogram Citra Asli (Grayscale)');
        xlabel('Intensitas (0-255)');
        ylabel('Jumlah Piksel');
        % lakukan perataan histogram
        eq_img = custom_histogram_equalization2(img, counts);
        % tampilkan citra hasil perataan
        figure;
        imshow(uint8(eq_img));
        title('Citra Hasil Histogram Equalization (Grayscale)');
```

```
        % hitung histogram hasil
        eq_counts = custom_image_histogram(eq_img);
        % tampilkan histogram hasil
        figure;
        bar(bins, eq_counts, 'k');
        title('Histogram Citra Setelah Equalization (Grayscale)');
        xlabel('Intensitas (0-255)');
        ylabel('Jumlah Piksel');
    else
        disp('Citra adalah gambar berwarna.');
        % pisahkan channel R, G, B
        R = img(:,:,1);
        G = img(:,:,2);
        B = img(:,:,3);
        % hitung histogram tiap channel
        countR = custom_image_histogram(R);
        countG = custom_image_histogram(G);
        countB = custom_image_histogram(B);
        bins = 0:255;
        % tampilkan histogram asli
        figure;
        subplot(3,1,1); bar(bins, countR, 'r'); title('Histogram R Asli');
        subplot(3,1,2); bar(bins, countG, 'g'); title('Histogram G Asli');
        subplot(3,1,3); bar(bins, countB, 'b'); title('Histogram B Asli');
        % lakukan perataan histogram per channel
        R_eq = custom_histogram_equalization2(R, countR);
        G_eq = custom_histogram_equalization2(G, countG);
        B_eq = custom_histogram_equalization2(B, countB);
        % gabungkan kembali
        eq_img = cat(3, uint8(R_eq), uint8(G_eq), uint8(B_eq));
        % tampilkan citra hasil
        figure;
        imshow(eq_img);
        title('Citra Hasil Histogram Equalization (RGB)');
        % hitung histogram hasil
        eq_countR = custom_image_histogram(R_eq);
        eq_countG = custom_image_histogram(G_eq);
        eq_countB = custom_image_histogram(B_eq);
        % tampilkan histogram hasil
        figure;
        subplot(3,1,1); bar(bins, eq_countR, 'r'); title('Histogram R
Setelah Equalization');
        subplot(3,1,2); bar(bins, eq_countG, 'g'); title('Histogram G
Setelah Equalization');
        subplot(3,1,3); bar(bins, eq_countB, 'b'); title('Histogram B
Setelah Equalization');
    end
end
```

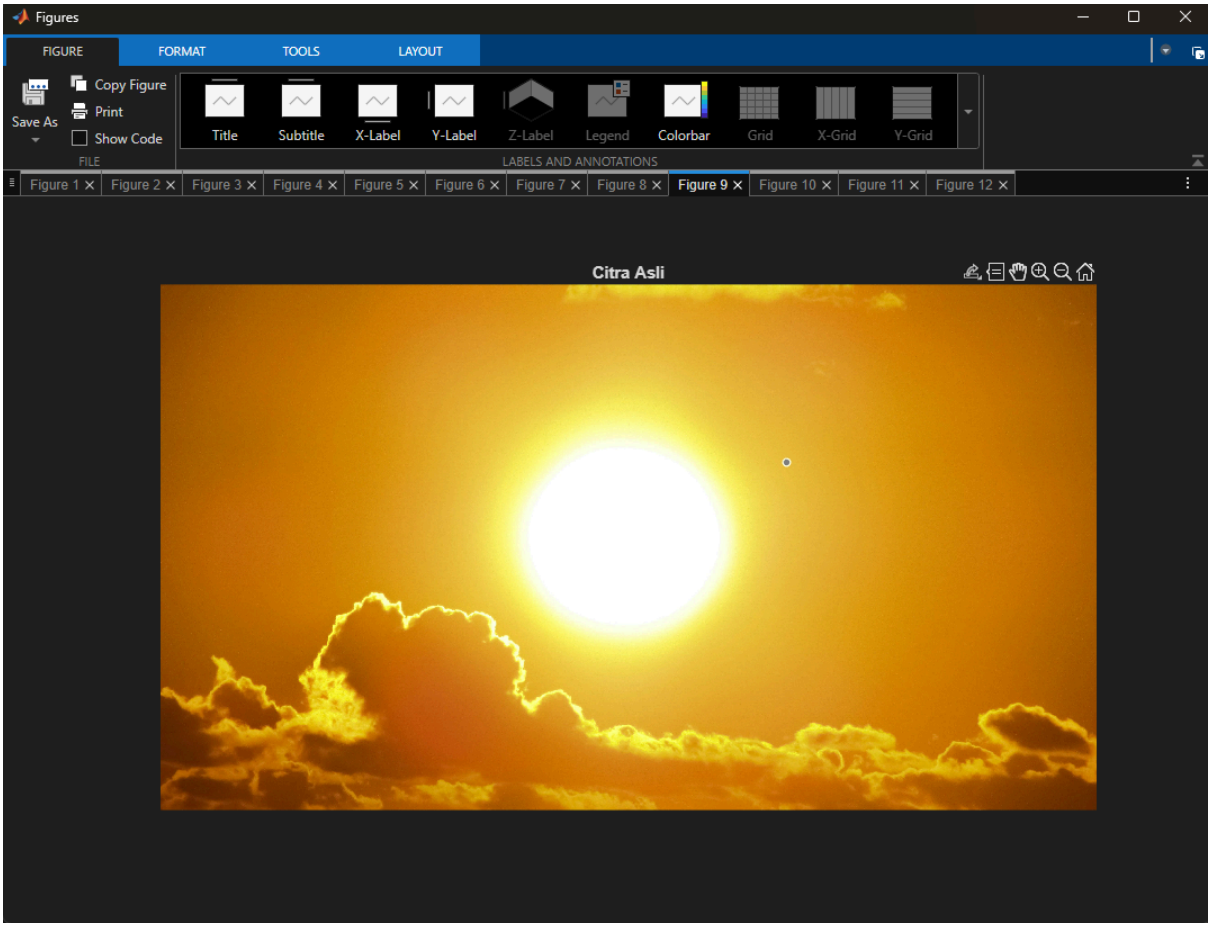
HASIL

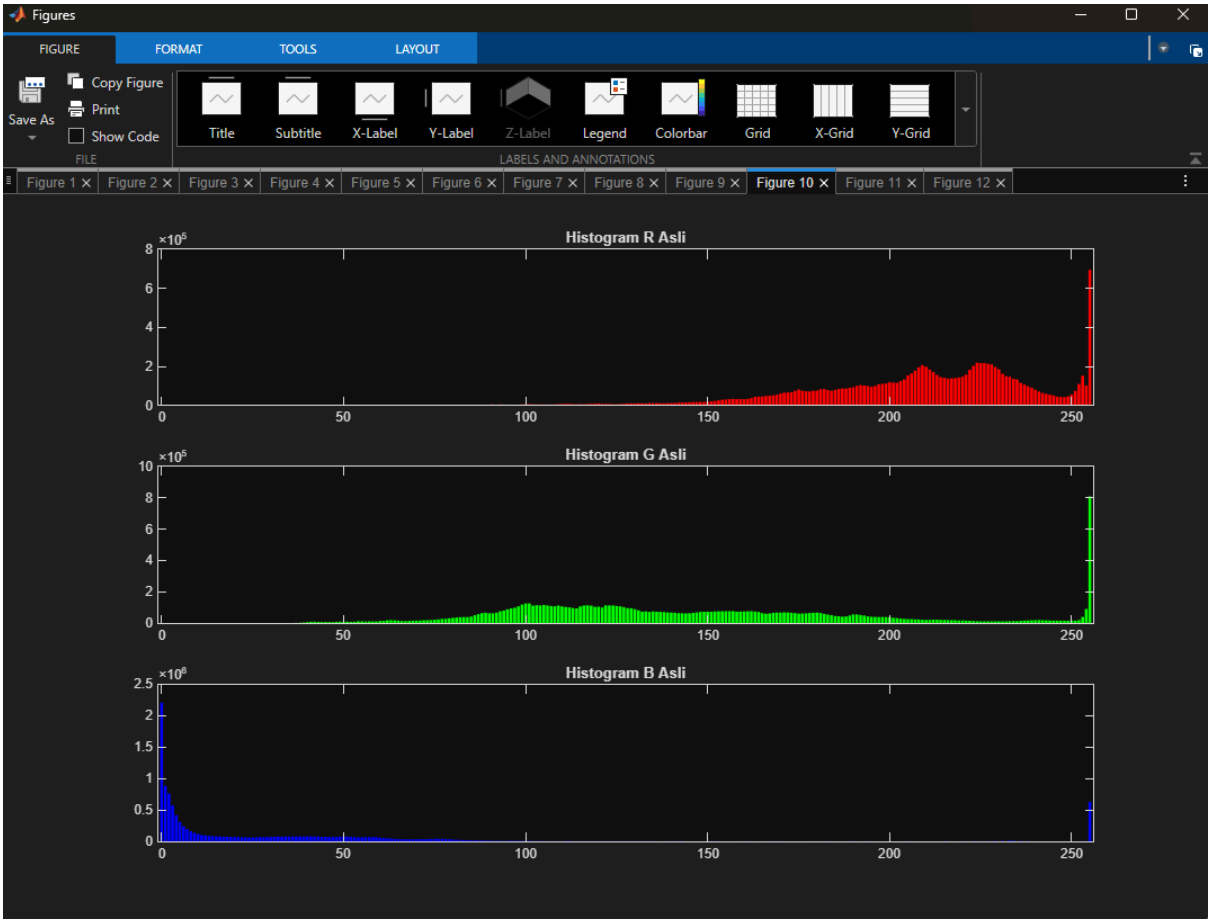

Figures
FIGURE
FORMAT
TOOLS
LAYOUT
Save As
Copy Figure
Print
Show Code
FILE
Title
Subtitle
X-Label
Y-Label
Z-Label
Legend
Colorbar
Grid
X-Grid
Y-Grid
LABELS AND ANNOTATIONS
Figure 1
Figure 2
Figure 3
Figure 4
Figure 5
Figure 6
Figure 7
Figure 8
Citra Asli

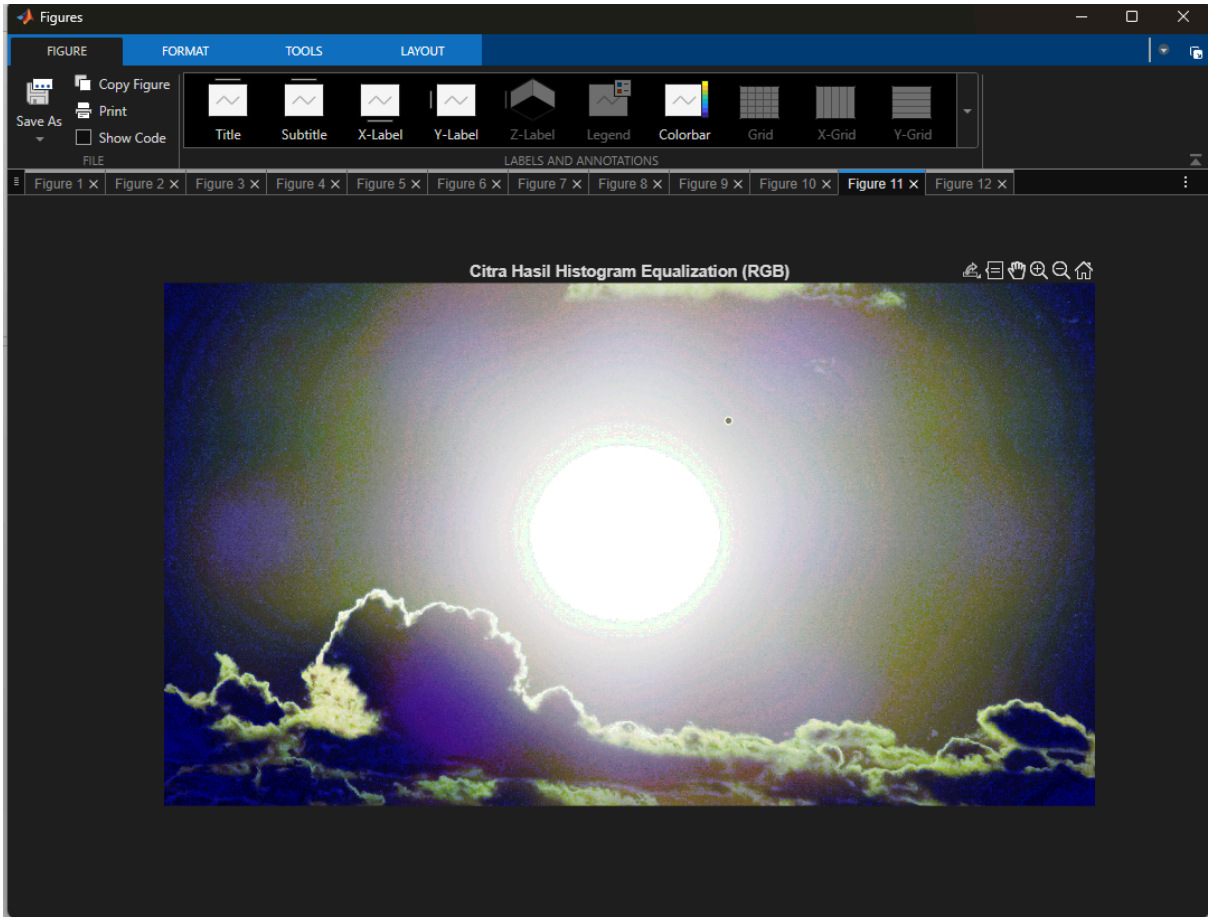

Figures
FIGURE
FORMAT
TOOLS
LAYOUT
Save As
Copy Figure
Print
Show Code
FILE
Title
Subtitle
X-Label
Y-Label
Z-Label
Legend
Colorbar
Grid
X-Grid
Y-Grid
LABELS AND ANNOTATIONS
Figure 1
Figure 2
Figure 3
Figure 4
Figure 5
Figure 6
Figure 7
Figure 8
Histogram R Asli
Histogram G Asli
Histogram B Asli
10000
8000
6000
4000
2000
0
0
50
100
150
200
250

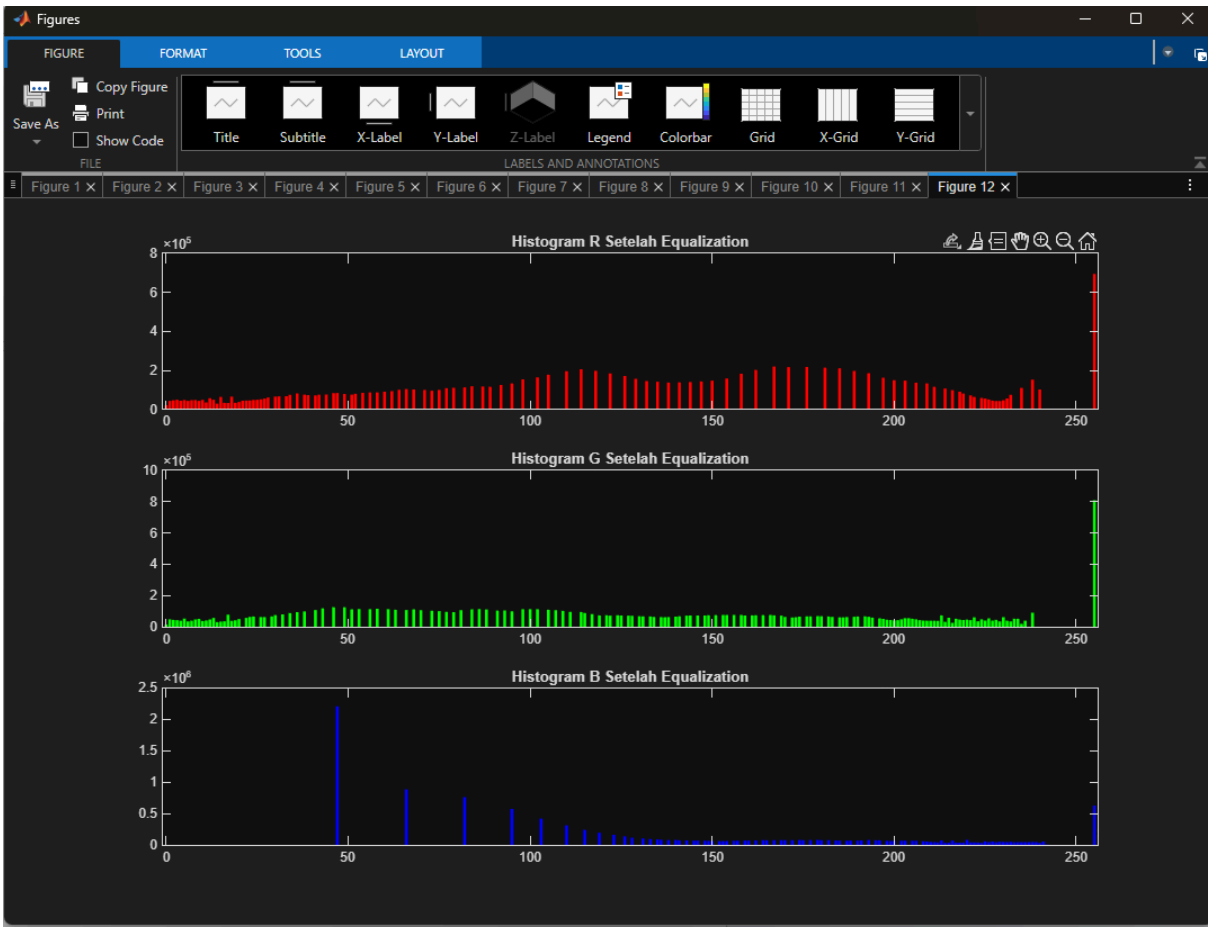

Figures
FIGURE
FORMAT
TOOLS
LAYOUT
Save As
Copy Figure
Print
Show Code
FILE
Title
Subtitle
X-Label
Y-Label
Z-Label
Legend
Colorbar
Grid
X-Grid
Y-Grid
LABELS AND ANNOTATIONS
Figure 1
Figure 2
Figure 3
Figure 4
Figure 5
Figure 6
Figure 7
Figure 8
Citra Hasil Histogram Equalization (RGB)


Figures
FIGURE
FORMAT
TOOLS
LAYOUT
Save As
Copy Figure
Print
Show Code
FILE
Title
Subtitle
X-Label
Y-Label
Z-Label
Legend
Colorbar
Grid
X-Grid
Y-Grid
LABELS AND ANNOTATIONS
Figure 1
Figure 2
Figure 3
Figure 4
Figure 5
Figure 6
Figure 7
Figure 8
Histogram R Setelah Equalization
Histogram G Setelah Equalization
Histogram B Setelah Equalization
10000
8000
6000
4000
2000
0
0
50
100
150
200
250


Figures
FIGURE
FORMAT
TOOLS
LAYOUT
Save As
Copy Figure
Print
Show Code
FILE
Title
Subtitle
X-Label
Y-Label
Z-Label
Legend
Colorbar
Grid
X-Grid
Y-Grid
LABELS AND ANNOTATIONS
Figure 1
Figure 2
Figure 3
Figure 4
Figure 5
Figure 6
Figure 7
Figure 8
Figure 9
Figure 10
Figure 11
Figure 12
Citra Asli


Figures
FIGURE
FORMAT
TOOLS
LAYOUT
Save As
Copy Figure
Print
Show Code
FILE
Title
Subtitle
X-Label
Y-Label
Z-Label
Legend
Colorbar
Grid
X-Grid
Y-Grid
LABELS AND ANNOTATIONS
Figure 1
Figure 2
Figure 3
Figure 4
Figure 5
Figure 6
Figure 7
Figure 8
Figure 9
Figure 10
Figure 11
Figure 12
Histogram R Asli
×10^5
Histogram G Asli
×10^5
Histogram B Asli
×10^6


Figures
FIGURE
FORMAT
TOOLS
LAYOUT
Save As
Copy Figure
Print
Show Code
FILE
Title
Subtitle
X-Label
Y-Label
Z-Label
Legend
Colorbar
Grid
X-Grid
Y-Grid
LABELS AND ANNOTATIONS
Figure 1
Figure 2
Figure 3
Figure 4
Figure 5
Figure 6
Figure 7
Figure 8
Figure 9
Figure 10
Figure 11
Figure 12
Citra Hasil Histogram Equalization (RGB)



ANALISIS

Ada dua contoh gambar input di atas, yaitu gambar yang terlalu gelap dan gambar yang terlalu terang. Kedua gambar tersebut memiliki histogram yang tidak merata pada setiap channel warna. Setelah dilakukan normalisasi, maka gambar yang terlalu gelap akan terlihat lebih terang dan gambar yang terlalu terang akan terlihat lebih gelap setelah normalisasi histogram. Hal tersebut terjadi karena normalisasi histogram akan membuat setiap histogram pada setiap channel warna menjadi lebib merata sehingga gambar output tidak terlalu terang ataupun terlalu gelap

SOAL 4

KODE

```
function [matched_image, mapping_table] = hist_match(inputImg, refImg)
%HIST_MATCH Melakukan Histogram Specification (Matching) pada sebuah citra.
%   [matched_image, mapping_table] = hist_match(inputImg, refImg)
%   menyesuaikan histogram dari 'inputImg' agar sesuai dengan histogram
%   dari 'refImg'.
%
%   Input:
%       inputImg     - Citra yang akan diubah (grayscale atau RGB).
%       refImg - Citra yang histogramnya menjadi target (grayscale atau
RGB).
%
%   Output:
%       matched_image   - Citra hasil dengan histogram yang sudah
disesuaikan.
%       mapping_table   - Tabel pemetaan intensitas yang digunakan
(opsional).
is_color = size(inputImg, 3) == 3;
if ~is_color % grayscale
   % Langkah 1: Hitung CDF (Cumulative Distribution Function) dari input
dan referensi.
   % imhist menghitung histogram, cumsum mengakumulasikannya, dan numel
menormalisasi.
   cdfInput = cumsum(custom_image_histogram(inputImg)) / numel(inputImg);
   cdfRef = cumsum(custom_image_histogram(refImg)) / numel(refImg);
   % Langkah 2: Buat tabel pemetaan (lookup table).
   mapping_table = zeros(256, 1);
   for intensity_level = 1:256
       % Untuk setiap tingkat keabuan 'i' pada citra input, cari tingkat
       % keabuan 'j' pada citra referensi yang memiliki nilai CDF terdekat.
       cdf_value = cdfInput(intensity_level);
       [~, nearest_index] = min(abs(cdf_value - cdfRef));
       mapping_table(intensity_level) = nearest_index - 1; % -1 karena
nilai piksel 0-255
   end
   % Langkah 3: Terapkan pemetaan ke setiap piksel citra input.
   % Operasi ini tervektorisasi, jauh lebih cepat daripada for-loop.
   matched_image = uint8(mapping_table(double(inputImg) + 1));
else % rgb

   % Untuk menghindari pergeseran warna (color shift), kita hanya akan
   % memodifikasi kanal intensitas (Value) dari model warna HSV.

   % Konversi kedua citra ke HSV
   hsv_input = rgb2hsv(inputImg);
   hsv_reference = rgb2hsv(refImg);

   % Ekstrak kanal V (Value/Intensitas) dari keduanya
   V_input = hsv_input(:, :, 3);
   V_reference = hsv_reference(:, :, 3);

   % Karena kanal V berjenis double [0,1], kita ubah ke uint8 [0,255]
   % agar bisa menggunakan imhist.
   V_input_uint8 = uint8(V_input * 255);
   V_reference_uint8 = uint8(V_reference * 255);

   % Lakukan proses matching persis seperti pada citra grayscale,
   % tetapi hanya pada kanal V.
   cdfInput_V = cumsum(custom_image_histogram(V_input_uint8)) /
numel(V_input_uint8);
   cdfRef_V = cumsum(custom_image_histogram(V_reference_uint8)) /
```

```
numel(V_reference_uint8);

   mapping_table = zeros(256, 1);
   for intensity_level = 1:256
       cdf_value = cdfInput_V(intensity_level);
       [~, nearest_index] = min(abs(cdf_value - cdfRef_V));
       mapping_table(intensity_level) = nearest_index - 1;
   end

   % Terapkan mapping ke kanal V dari citra input
   V_matched_uint8 = uint8(mapping_table(V_input_uint8 + 1));

   % Kembalikan V yang sudah di-match ke format double [0,1]
   V_matched = double(V_matched_uint8) / 255;

   % Gabungkan kembali dengan kanal Hue (H) dan Saturation (S) ASLI dari
citra input.
   % Ini adalah kunci untuk mempertahankan warna asli.
   final_hsv = cat(3, hsv_input(:, :, 1), hsv_input(:, :, 2), V_matched);

   % Konversi citra HSV hasil kembali ke RGB untuk ditampilkan.
   matched_image = hsv2rgb(final_hsv);
end
end
```

HASIL

Gambar Berwarna

Gambar Grayscale

ANALISIS

PEMBAGIAN TUGAS

Varraz Hazzandra Abrar	13521020	● 2. Perbaikan kualitas pada citra dengan metode image brightening, citra negatif dan balikan citra negatif, transformasi log, transformasi pangkat ● 4. Melakukan perbaikan citra dengan teknik histogram specification ● GUI
Akbar Al Fattah	13522036	● 1. Histogram gambar grayscale dan Gambar berwarna ● 3. Histogram Equalization dari gambar grayscale dan gambar berwarna

LAMPIRAN

Pranala GitHub: https://github.com/DeltDev/Tugas-Pengolahan-Citra-1