

**PENGOLAHAN CITRA DIGITAL**  
**“PENGOLAHAN CITRA BERWARNA”**



**OLEH :**

**Wahid Radikal Akhlak**

**F55121022**

**A**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**JURUSAN TEKNOLOGI INFORMASI**

**FAKULTAS TEKNIK**

**UNIVERSITAS TADULAKO**

**2023**

## A. Dasar warna

Manusia sebenarnya melihat warna adalah karena cahaya yang dipantulkan oleh objek. Dalam hal ini, spektrum cahaya kromatis berkisar antara 400-700 nm (Zhou, dkk., 2010). Istilah kromatis berarti kualitas warna cahaya yang ditentukan oleh panjang gelombang.

Karakteristik persepsi mata manusia dalam yang membedakan antara satu warna dengan warna yang lain berupa *hue*, *saturation*, dan *brightness*.

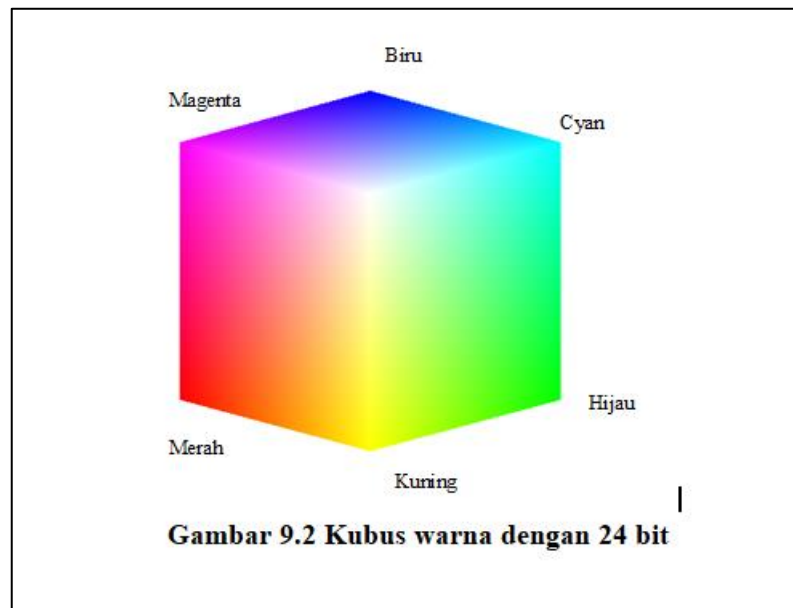
- ❖ *Hue* merujuk ke warna yang dikenal manusia, seperti merah dan hijau. Properti ini mencerminkan warna yang ditangkap oleh mata manusia yang menanggapi berbagai nilai panjang gelombang cahaya. Sebagai contoh, bila mata menangkap panjang gelombang antara 430 dan 480 nanometer, sensasi yang diterima adalah warna biru, sedangkan jika panjang gelombang berkisar antara 570 sampai dengan 600 nm, warna yang terlihat adalah kuning (Crane, 1997), sedang campuran merah dan hijau terlihat kuning.
- ❖ *Saturation* menyatakan tingkat kemurnian warna atau seberapa banyak cahaya putih yang tercampur dengan *hue*. Setiap warna murni bersaturasi 100% dan tidak mengandung cahaya putih sama sekali. Dengan kata lain, suatu warna murni yang bercampur dengan cahaya putih memiliki saturasi antara 0 dan 100%.
- ❖ *Brightness* atau kadang disebut *lightness* (kecerahan) menyatakan intensitas pantulan objek yang diterima mata. Intensitas dapat dinyatakan sebagai perubahan warna putih menuju abu-abu dan terakhir mencapai ke warna hitam, atau yang dikenal dengan istilah aras keabuan.

## B. Ruang warna

Gonzalez & Woods (2002) mendefinisikan ruang warna (atau kadang disebut sistem warna atau model warna) sebagai suatu spesifikasi sistem koordinat dan suatu subruang dalam sistem tersebut dengan setiap warna dinyatakan dengan satu titik di dalamnya. Tujuan dibentuknya ruang warna adalah untuk memfasilitasi spesifikasi warna dalam bentuk suatu standar. Ruang warna yang paling dikenal pada perangkat komputer adalah RGB, yang sesuai dengan watak manusia dalam menangkap warna. Namun, kemudian dibuat banyak ruang warna, antara lain HSI, CMY, LUV, dan YIQ.

## 1. Ruang Warna RGB

Ruang warna RGB biasa diterapkan pada monitor CRT dan kebanyakan sistem grafika komputer. Ruang warna ini menggunakan tiga komponen dasar yaitu merah (R), hijau (G), dan biru (B). Setiap piksel dibentuk oleh ketiga komponen tersebut. Model RGB biasa disajikan dalam bentuk kubus tiga dimensi, dengan warna merah, hijau, dan biru berada pada pojok sumbu (Gambar 9.1). Warna hitam berada pada titik asal dan warna putih berada di ujung kubus yang berseberangan. Gambar 9.2 memperlihatkan kubus warna secara nyata dengan resolusi 24 bit. Perlu diketahui, dengan menggunakan 24 bit, jumlah warna mencapai 16.777.216.



## 2. Ruang Warna CMY/CMYK

Model warna CMY (*cyan, magenta, yellow*) mempunyai hubungan dengan RGB sebagai berikut:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (9.1)$$

Dalam hal ini, R, G dan B berupa nilai warna yang telah dinormalisasi, dengan jangkauan [0, 1].

Pada CMY, warna hitam diperoleh jika C, M, dan Y bernilai sama. Namun, pada aplikasi printer, warna hitam ditambahkan tersendiri sehingga membentuk CMYK, dengan K menyatakan warna hitam. Alasannya, kalau ada warna hitam, warna dapat diambilkan secara langsung dari tinta hitam, tanpa perlu mencampur dengan warna lain. Lagipula, tinta warna hitam lebih murah daripada tinta berwarna dan paling sering digunakan terutama untuk teks.



Perlu diketahui, konversi dari CMY ke CMYK dapat menggunakan berbagai cara perhitungan. Salah satu rumus yang digunakan sebagai berikut (Crane, 1997):

$$K = \min(C, M, Y) \quad (9.2)$$

$$C' = C - K \quad (9.3)$$

$$M' = M - K \quad (9.4)$$

$$Y' = Y - K \quad (9.5)$$

Dengan pendekatan seperti itu, salah satu dari C', M', atau Y' akan bernilai 0. Namun, ada pula yang menggunakan rumus seperti berikut (Dietrich, 2003):

$$K = \min(C, M, Y) \quad (9.6)$$

$$C = (C - K)/(1-K) \quad (9.7)$$

$$M = (M - K)/(1-K) \quad (9.8)$$

$$Y = (Y - K)/(1-K) \quad (9.9)$$

### 3. Ruang Warna YIQ

Ruang warna YIQ, yang juga dikenal dengan nama ruang warna NTSC, dirumuskan oleh NTSC ketika mengembangkan sistem televisi berwarna di Amerika Serikat. Pada

model ini, Y disebut *luma* (yang menyatakan luminans) dan I serta Q disebut *chroma*. Konversi YIQ berdasarkan RGB sebagai berikut:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0,299 & 0,587 & 0,114 \\ 0,596 & -0,274 & -0,322 \\ 0,211 & -0,523 & 0,312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (9.2)$$

Komposisi RGB untuk Y secara statistis optimal untuk ditampilkan pada penerima TV hitam-putih.

Matriks yang berisi koefisien konversi mempunyai ciri-ciri sebagai berikut:

- 1) jumlah elemen dalam baris pertama bernilai 1;
- 2) jumlah koefisien elemen dalam baris kedua maupun baris ketiga bernilai nol.

Adapun konversi RGB dari YIQ sebagai berikut:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1,000 & 0,956 & 0,621 \\ 1,000 & -0,272 & -0,647 \\ 1,000 & -1,106 & 1,703 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix} \quad (9.3)$$

#### 4. Ruang Warna YCbCr

Ruang warna YCbCr biasa digunakan pada video digital. Pada ruang warna ini, komponen Y menyatakan intensitas, sedangkan C<sub>b</sub> dan C<sub>r</sub> menyatakan informasi warna. Proses konversi dari RGB dilakukan dengan beberapa cara. Contoh berikut didasarkan pada rekomendasi CCIR 601-1 (Crane, 1997):

$$Y = 0,29900R + 0,58700G + 0,11400B \quad (9.4)$$

$$C_b = -0,16874R - 0,33126G + 0,5000B \quad (9.5)$$

$$C_r = +0,5000R - 0,41869G - 0,08131B \quad (9.6)$$

Adapun pengonversian dari YCbCr ke RGB sebagai berikut:

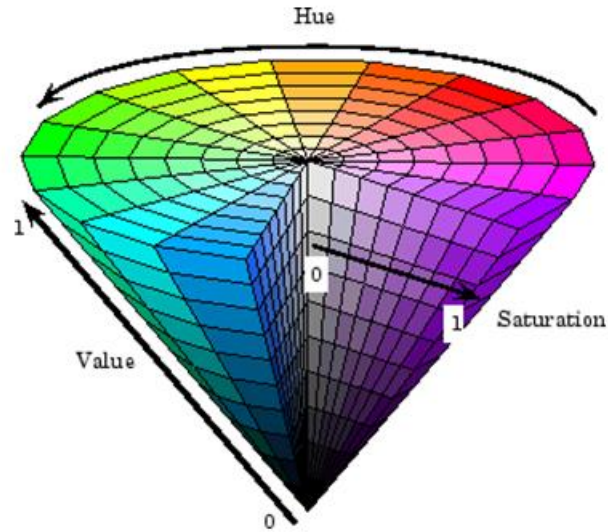
$$R = Y + 1,40200C_r \quad (9.7)$$

$$G = Y - 0,34414C_b - 0,71414C_r \quad (9.8)$$

$$B = Y + 1,77200C_b \quad (9.9)$$

## 5. Ruang Warna HSI, HSV, dan HSL

HSV dan HSL merupakan contoh ruang warna yang merepresentasikan warna seperti yang dilihat oleh mata manusia. H berasal dari kata “hue”, S berasal dari “saturation”, L berasal dari kata “luminance”, I berasal dari kata “intensity”, dan V berasal dari “value”.



**Gambar 9.4 Ruang warna HSV**

(Sumber: MATLAB)

Model HSV, yang pertama kali diperkenalkan A. R. Smith pada tahun 1978, ditunjukkan pada Gambar 9.4. Untuk mendapatkan nilai H, S, V berdasarkan R, G, dan B, terdapat beberapa cara. Cara yang tersederhana (Acharya & Ray, 2005) adalah seperti berikut.

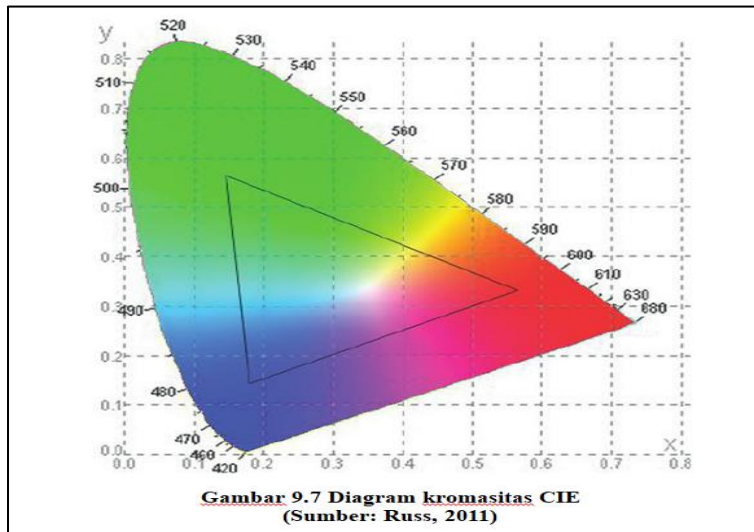
$$H = \tan \left( \frac{3(G-B)}{(R-G)+(R-B)} \right) \quad (9.10)$$

$$S = 1 - \frac{\min(R,G,B)}{V} \quad (9.11)$$

$$V = \frac{R+G+B}{3} \quad (9.12)$$

## 6. Ruang Warna CIELAB

CIELAB adalah nama lain dari CIE  $L^*a^*b^*$ . Diagram kromasitas CIE (Commission Internationale de L'Eclairage) ditunjukkan pada Gambar 9.7. Pada diagram tersebut, setiap perpaduan  $x$  dan  $y$  menyatakan suatu warna. Namun, hanya warna yang berada dalam area ladam (tapal kuda) yang bisa terlihat. Angka yang berada di tepi menyatakan panjang gelombang cahaya. Warna yang terletak di dalam segitiga menyatakan warna-warna umum di monitor CRT, yang dapat dihasilkan oleh komponen warna merah, hijau, dan biru.



Transformasi RGB ke CIELAB dimulai dengan melakukan perhitungan sebagai berikut:

$$X = 0,412453R + 0,357580G + 0,180423B \quad (9.40)$$

$$Y = 0,212671R + 0,715160G + 0,072169B \quad (9.41)$$

$$Z = 0,019334R + 0,119193G + 0,950227B \quad (9.42)$$

Selanjutnya,  $L^*a^*b^*$  didefinisikan sebagai berikut:

$$L^* = 116f\left(\frac{Y}{Y_n}\right) - 16 \quad (9.43)$$

$$a^* = 500 \left[ f\left(\frac{X}{X_n}\right) - f\left(\frac{Y}{Y_n}\right) \right] \quad (9.44)$$

$$b^* = 200 \left[ f\left(\frac{Y}{Y_n}\right) - f\left(\frac{Z}{Z_n}\right) \right] \quad (9.45)$$

Dalam hal ini,  $f(q)$  dihitung seperti berikut:

$$f(q) = \begin{cases} q^{\frac{1}{3}}, & \text{jika } q > 0,008856 \\ 7,787q + 16/116, & \text{untuk yang lain} \end{cases} \quad (9.46)$$

$X_n, Y_n, Z_n$  diperoleh melalui  $R=G=B=1$  dengan jangkauan  $R, G, B$  berupa  $[0, 1]$ .

### C. Memperoleh statistika warna

Fitur warna dapat diperoleh melalui perhitungan statistis seperti rerata, deviasi standar, *skewness*, dan kurtosis (Martinez & Martinez, 2002). Sebagai contoh, fitur-fitur tersebut dapat digunakan untuk kepentingan identifikasi tanaman hias (Kadir, dkk., 2011b dan Kadir, dkk., 2011c). Perhitungan dikenakan pada setiap komponen R, G, dan B.

Rerata memberikan ukuran mengenai distribusi dan dihitung dengan menggunakan rumus:

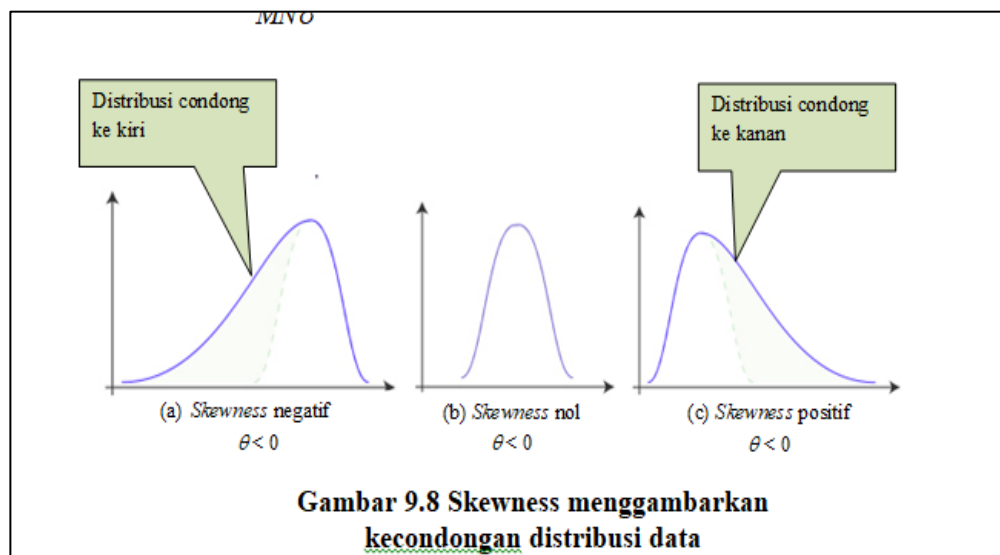
$$\mu = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N P_{ij} \quad (9.47)$$

Varians menyatakan luas sebaran distribusi. Akar kuadrat varians dinamakan sebagai deviasi standar. Adapun rumus yang digunakan untuk menghitungnya sebagai berikut:

$$\sigma = \sqrt{\frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N (P_{ij} - \mu)^2} \quad (9.48)$$

*Skewness* atau kecondongan menyatakan ukuran mengenai ketidaksimetrisan. Distribusi dikatakan condong ke kiri apabila memiliki nilai *skewness* berupa bilangan negatif. Sebaliknya, distribusi dikatakan condong ke kanan apabila memiliki nilai *skewness* berupa bilangan positif. Jika distribusi simetris, koefisien *skewness* bernilai nol. Ilustrasi *skewness* dapat dilihat pada Gambar 9.8. *Skewness* dihitung dengan cara seperti berikut:

$$\theta = \frac{\sum_{i=1}^M \sum_{j=1}^N (P_{ij} - \mu)^3}{MN\sigma^3} \quad (9.49)$$

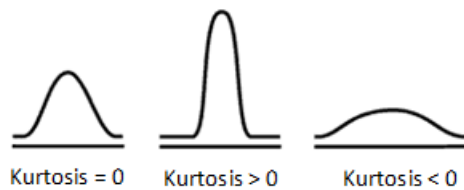




Kurtosis merupakan ukuran yang menunjukkan sebaran data bersifat meruncing atau menumpul. Perhitungannya seperti berikut:

$$\gamma = \frac{\sum_{i=1}^M \sum_{j=1}^N (P_{ij} - \mu)^4}{MN\sigma^4} - 3 \quad (9.50)$$

Definisi di atas membuat distribusi normal standar memiliki kurtosis nol. Nilai positif mengindikasikan bahwa distribusi bersifat lancip dan nilai negatif menyatakan distribusi yang datar (lihat Gambar 9.9). Perlu diketahui, pada Persamaan 9.47 hingga 9.50, M adalah tinggi citra, N menyatakan lebar citra, dan  $P_{ij}$  adalah nilai warna pada baris  $i$  dan kolom  $j$ .



**Gambar 9.9 Kurtosis menggambarkan keruncingan distribusi normal**

#### D. Mengatur kecerahan dan kontras

Pada Bab 3 telah dijelaskan cara mengatur kontras dan kecerahan pada citra berskala keabuan. Cara seperti itu, secara prinsip dapat diterapkan pada citra berwarna. Untuk melihat efek kecerahan dan kontras, cobalah beberapa perintah berikut.

```
>> Img = imread('C:\Image\inns.png'); ⌘
>> imshow(Img) ⌘
>>
```

Kode di atas digunakan untuk melihat citra inns.png (Gambar 9.10.(a)). Selanjutnya,

```
>> C = Img + 30; ⌘
>> imshow(C) ⌘
>>
```

membuat citra dicerahkan sejauh 30. Hasilnya ditunjukkan pada Gambar 9.10(b). Lalu, cobalah kode berikut:

```
>> K = 2 * C; ⌘
>> imshow(K) ⌘
```

>>

Hasilnya dapat dilihat pada Gambar 9.10(c). Hal yang menarik dapat diperoleh dengan hanya memberikan kontras pada komponen R. Caranya:

```
>> K = C; ⌘
```

```
>> K(:, :, 1) = 2 * K(:, :, 1); ⌘
```

>>

Kode di atas mula-mula membuat K bernilai sama dengan C (efek pencerahan). Selanjutnya,

$K(:, :, 1) = 2 * K(:, :, 1);$

membuat hanya komponen R saja yang dinaikkan dua kali. hasilnya ditunjukkan pada Gambar 9.10(d).



(a) Citra inns.png



(b) Efek pencerahan melalui  
 $C = \text{Img} + 30;$



(c) Efek kontras dan pencerahan melalui  
 $C = 2 \times (\text{Img} + 30)$



(d) Efek pencerahan melalui  $C = \text{Img} + 30$   
pada komponen R, G, dan B  
dan kontras sebesar 2 kali hanya pada  
komponen R

**Gambar 9.10 Pencerahan dan peningkatan kontras pada citra berwarna**

## E. Menghitung jumlah warna

Berapa jumlah warna yang menyusun suatu citra? Bila terdapat kebutuhan seperti itu, jumlah warna dapat dihitung dengan memanfaatkan fungsi `jumwarna` berikut.

```
function [jumlah] = jumwarna(berkas)
% JUMWARNA Menghitung jumlah warna pada citra RGB
%     Masukan:
%     berkas berupa citra warna
%     Nilai balik berupa jumlah warna

RGB = double(imread(berkas));
[m,n,d] = size(RGB);

if (d ~= 3)
    disp('Citra harus berupa citra berwarna');
    return;
end
RGB = double(RGB);
Data = zeros(1, m * n); % Array kosong
jum = 0;
for i=1:m
    for j=1:n
        jum = jum + 1;

        r = RGB(i,j,1);
        g = RGB(i,j,2);
        b = RGB(i,j,3);

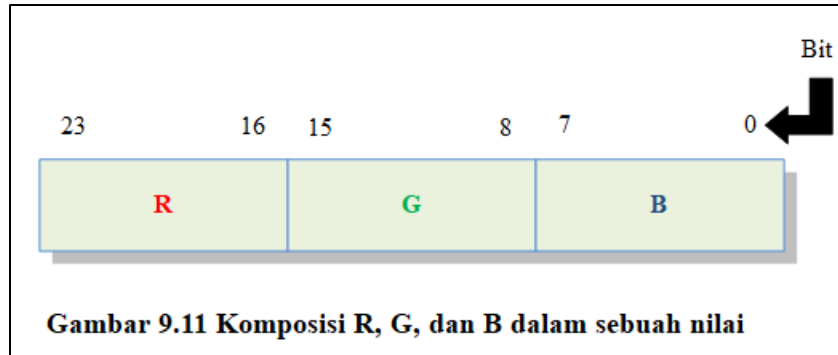
        Data(jum) = bitshift(r,16) + bitshift(g, 8) + b;
    end
end
% Urutkan data pada array Data
Data = sort(Data);
% Hitung jumlah warna
jwarna = 1;
for i = 1 : jum - 1
    if Data(i) ~= Data(i+1)
        jwarna = jwarna + 1;
    end
end
jumlah = jwarna;
```

### Pada Akhir Program

Penghitungan warna dilakukan dengan mula-mula menyusun komponen R, G, dan B untuk setiap piksel menjadi sebuah nilai dengan komposisi seperti terlihat pada Gambar 9.11. Untuk keperluan seperti itu, maka:

- G perlu digeser ke kiri sebanyak 8 bit dan
- R perlu digeser ke kiri sebanyak 16 bit.

Pada *Octave* dan *MATLAB*, penggeseran bit dilakukan melalui fungsi **bitshift**.



Setelah nilai gabungan R, G, dan B terbentuk dan diletakkan ke larik Data, isi larik tersebut diurutkan. Pengurutan tersebut dimaksudkan untuk mempermudah penghitungan jumlah warna. Implementasi penghitungan pada data yang telah urut seperti berikut:

```
jwarna = 1;
for i = 1 : jum - 1
    if Data(i) ~= Data(i+1)
        jwarna = jwarna + 1;
    end
end
```

Berdasarkan kode di atas, nilai jwarna dinaikkan sebesar satu sekiranya suatu nilai dan nilai berikutnya tidak sama.

Contoh penggunaan fungsi jumwarna:

```
>> C = jumwarna('C:\Image\lapangan.png')
C = 92475
>>
```

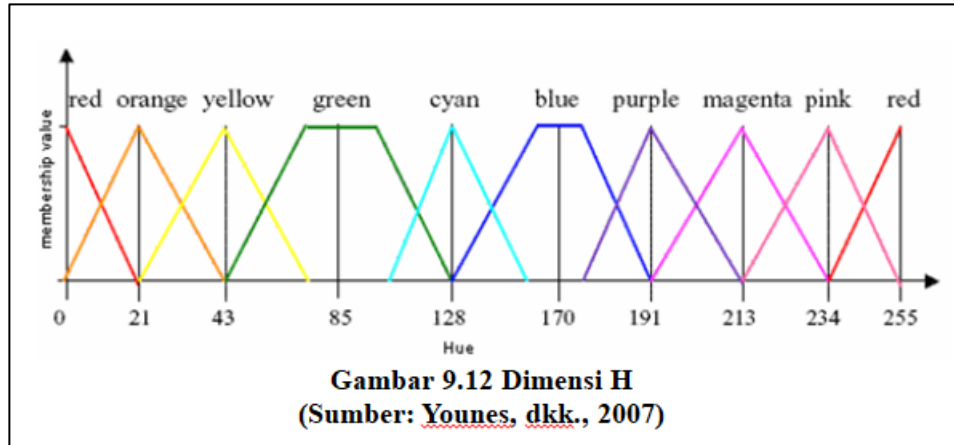
Hasil di atas menyatakan bahwa jumlah warna yang terkandung pada lapangan.png adalah 92.475.

## F. Aplikasi pencarian citra berdasarkan warna dominan.

Contoh aplikasi penentuan warna dominan pada daun dibahas oleh Kadir (2011d). Dengan mengacu tulisan dalam artikel tersebut, contoh aplikasi pada pencarian citra berdasarkan warna dominan akan dijelaskan pada subbab ini. Model yang sederhana akan diimplementasikan dalam bentuk program.

Agar pencarian menurut warna dominan seperti merah atau hijau dapat dilakukan, setiap piksel yang menyusun citra harus dapat dipetakan ke dalam warna alamiah semacam merah atau hijau. Hal ini dapat dilakukan kalau ruang warna yang digunakan berupa HSV. Pada sistem HSV, komponen *hue* sebenarnya menyatakan warna seperti yang biasa dipahami oleh

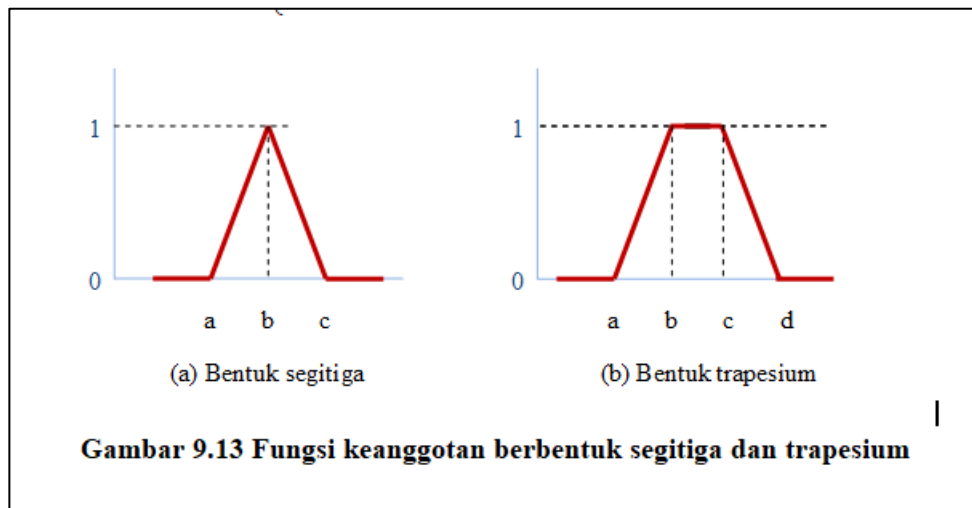
manusia. Younes, dkk. (2007) membuat model *fuzzy* untuk menyatakan warna seperti terlihat pada Gambar 9.12.



Berdasarkan Gambar 9.12, dimungkinkan untuk menerapkan *fuzzy logic* untuk menentukan suatu area warna beserta derajat keanggotaannya. Model tersebut didasarkan pada sumbu melingkar pada komponen *Hue* (H). Mengingat warna merah berada pada nilai H sama dengan nol, maka jangkauan warna merah berada di sekitar angka 0 dan 255.

Dua jenis fungsi keanggotaan *fuzzy* yang digunakan pada Gambar 9.12 berbentuk segitiga dan trapesium dan digambarkan kembali pada Gambar 9.13. Berdasarkan Gambar 9.13 tersebut, fungsi keanggotaan berbentuk segitiga didefinisikan sebagai berikut:

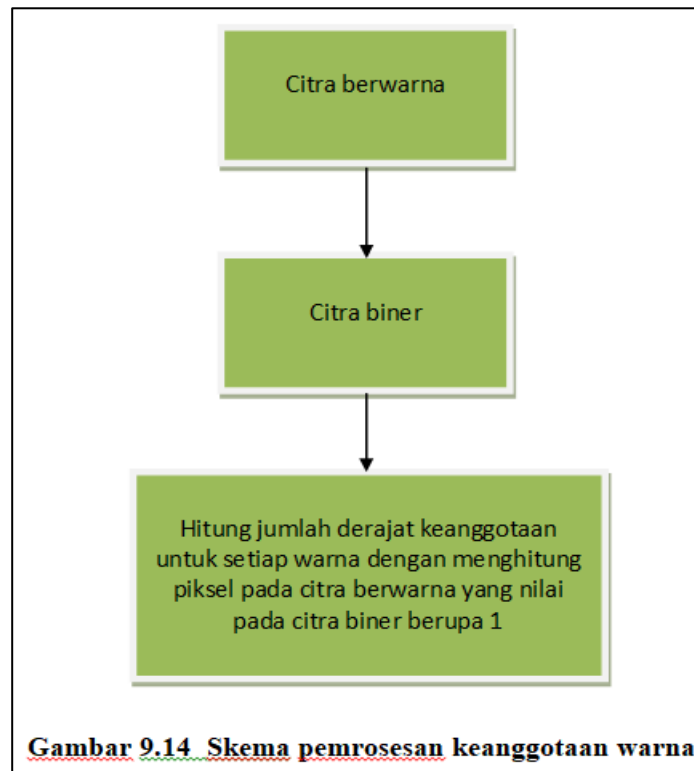
$$\Lambda(x; a, b, c) = \begin{cases} 0 & x \leq a, \\ (x - a) / (b - a) & a < x \leq b, \\ (c - x) / (c - b) & b < x \leq c, \\ 0 & x > c \end{cases} \quad (9.51)$$



Adapun fungsi keanggotaan berbentuk trapesium didefnisikan sebagai berikut:

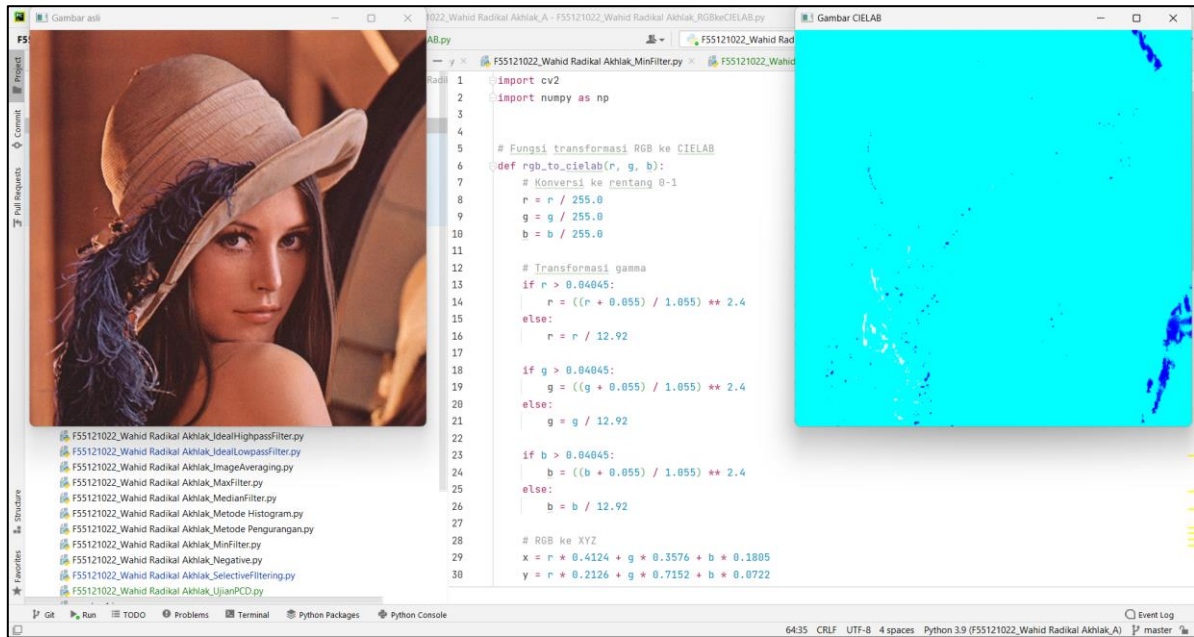
$$\Pi(x; a, b, c) = \begin{cases} 0 & x \leq a, \\ (x - a)/(b - a) & a < x \leq b, \\ 1 & b < x \leq c, \\ (c - x)/(c - b) & c < x \leq d, \\ 0 & x > d \end{cases} \quad (9.52)$$

Khusus untuk warna hitam, putih, dan abu-abu dilakukan penanganan secara khusus, yaitu dengan memperhatikan komponen S dan V. Sebagaimana tersirat pada Gambar 9.4, warna hitam diperoleh manakala V bernilai 0 dan warna putih diperoleh ketika S bernilai 0. Warna abu-abu diperoleh ketika S bernilai rendah. Semakin rendah nilai S maka warna abu-abu semakin terlihat tua. Secara garis besar, proses untuk memasukkan setiap piksel ke dalam suatu kategori warna (merah, hijau, dsb.) dilaksanakan melalui mekanisme seperti terlihat pada Gambar 9.14.

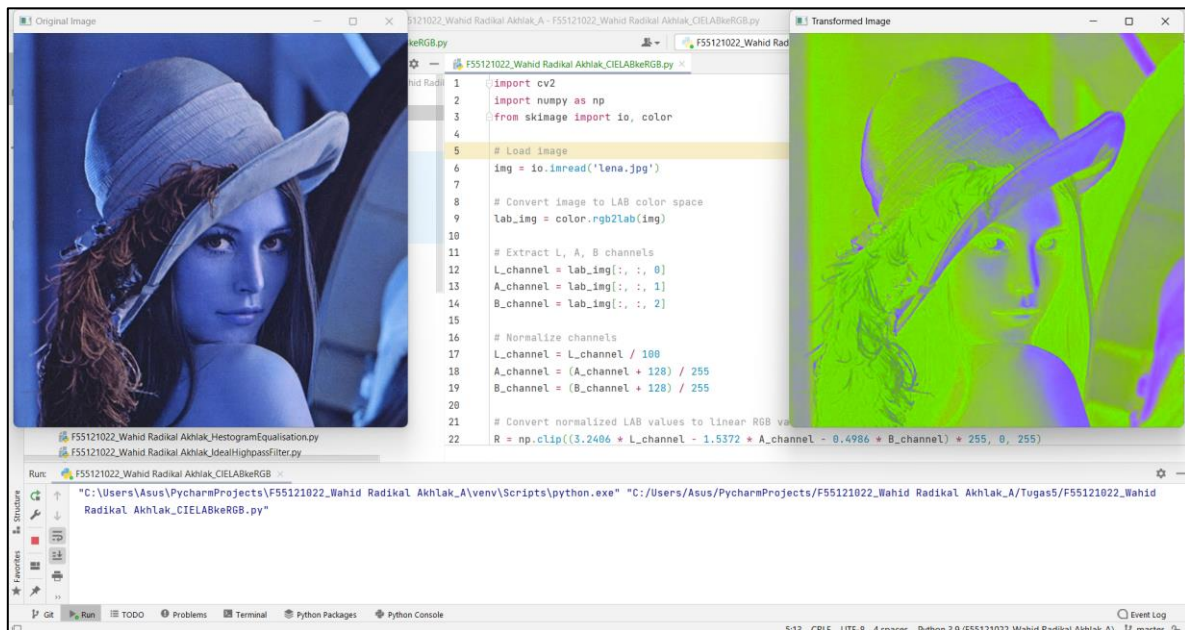


Citra biner digunakan untuk menentukan area pada citra berwarna yang diproses khusus yang merupakan bagian daun. Dalam hal ini bagian yang berisi daun akan bernilai 1 pada citra biner dan 0 untuk latarbelakang. Selanjutnya, nilai H, S, V digunakan untuk menentukan kelompok warna piksel.

## TRANSFORMASI GAMBAR RGB KE CIELAB PADA PYCHARM

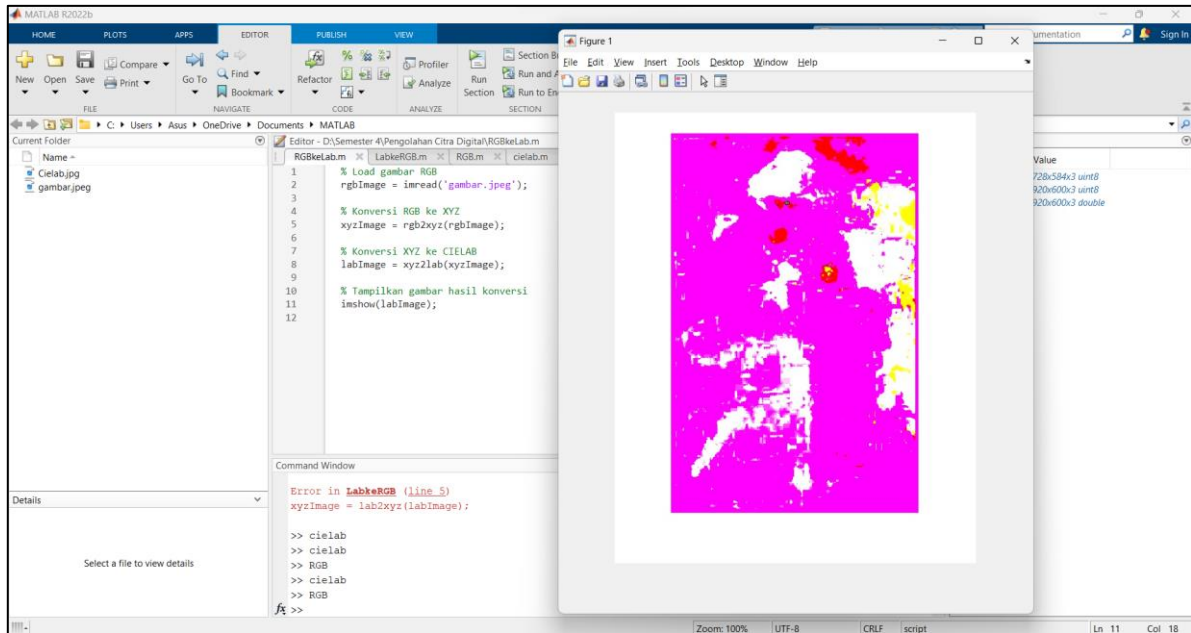


## TRANSFORMASI GAMBAR CIELAB KE RGB PADA PYCHARM

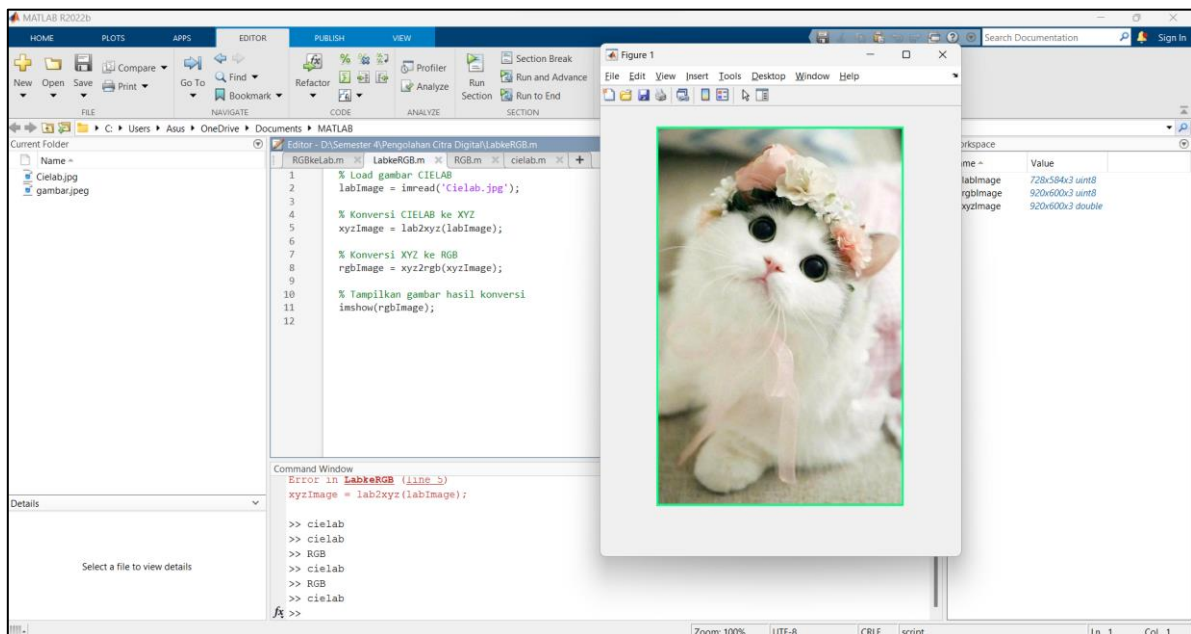




## TRANSFORMASI GAMBAR RGB KE CIELAB PADA MATLAB



## TRANSFORMASI GAMBAR CIELAB KE RGB PADA MATLAB



**LINK GITHUB:**

[https://github.com/WahidRadikalAkhlaq/F55121022\\_WahidRadikalAkhlaq\\_PCD\\_A.git](https://github.com/WahidRadikalAkhlaq/F55121022_WahidRadikalAkhlaq_PCD_A.git)