

## PRAKTIKUM 2

### Pengenalan citra digital

#### Materi

- Tipe Image dan Ukuran
- Image Format
- Image Read and Write
- Image Conversion.

#### Tujuan Praktikum

- Mahasiswa dapat menyimpan *image* baru dengan format tertentu.
- Mahasiswa dapat menjalankan fungsi-fungsi dasar OpenCV untuk *write* , *convert*

#### A. Penyajian

Terdapat 4 tipe dasar dari sebuah image:

##### 1. *Binary*

Piksel citra hanya terdiri dari warna hitam dan putih. Karena hanya ada dua nilai kemungkinan untuk setiap piksel, sehingga hanya perlu satu bit per piksel. Citra biner sangat efisien dalam hal penyimpanan.



Gambar 2.1. *Binary Image*

##### 2. Grayscale

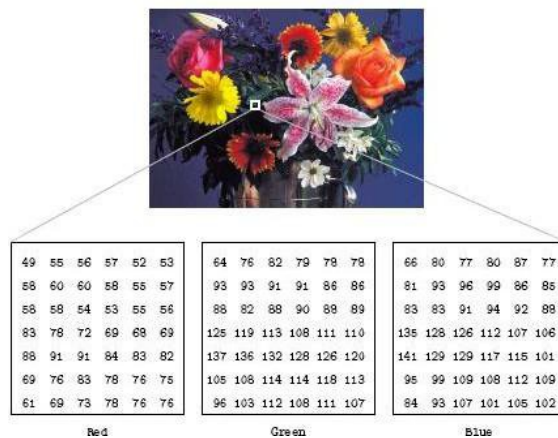
Setiap piksel adalah warna abu-abu, biasanya dari 0 (hitam) sampai 255 (putih). Setiap piksel direpresentasikan oleh 8 bits, atau 1 byte.



Gambar 2.2. *Grayscale Image*

### 3. True Color / RGB (*Red Green Blue*)

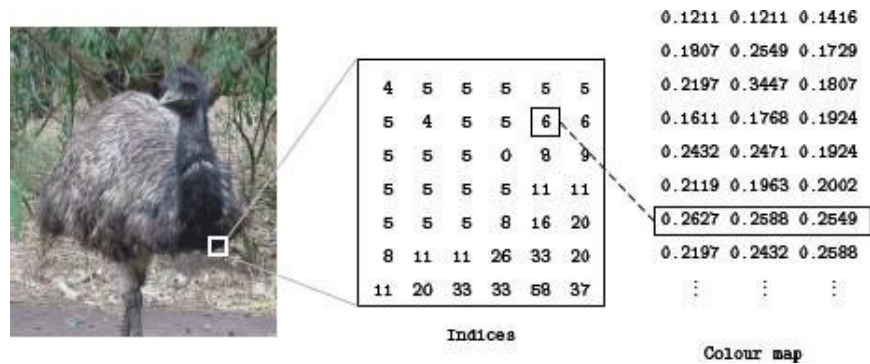
Pada tipe citra ini, setiap piksel mempunyai warna tertentu yaitu jumlah warna merah, hijau dan biru di dalamnya. Masing-masing komponen warna berkisaran dari 0-255, maka total komponen warna untuk setiap citra  $256^3 = 16.777.216$  kemungkinan perbedaan citra dalam sebuah image. Setiap warna direpresentasikan ke dalam 8 *bits* sehingga RGB *Image* disebut juga *24-bit colour images*.



Gambar 2.3. *RGB Image*

### 4. *Indexed*

Kebanyakan citra hanya memiliki sebagian kecil dari lebih enam belas juta kemungkinan warna. Untuk kenyamanan penyimpanan dan penanganan file, citra memiliki sebuah *associated colour map* atau *colour palette*, yang merupakan daftar semua warna yang digunakan dalam sebuah citra. Setiap piksel memiliki nilai yang tidak memberikan warna (seperti pada citra RGB), tetapi indeks untuk warna dalam *map*.



Gambar 2.4. *Indexed colour image*

Dalam OpenCV jenis citra yang telah di Load sebelumnya dapat dikonversi menjadi jenis citra lain dengan fungsi `cvtColor` dengan fungsi sebagai berikut:

```
img = cv2.cvtColor(input_image, flag)
```

Terdapat beberapa format citra:

1. BMP ( Microsoft Windows Bitmap)
2. GIF (Graphics Interchange Files)
3. JPEG (Joint Photographic Experts Group)
4. PNG (Portable Network Graphics)
5. TIFF (Tagged Image File Format)

Pengubahan format citra pada OpenCV dapat dilakukan menggunakan fungsi `imread` yang akan menyimpan citra pada direktori tertentu. Fungsi `imread` sebagai berikut:

```
# Load a color image in grayscale img =  
cv2.imread('image.jpg', 0)
```

Argumen pertama akan memberikan nama file dengan format citra baru yang diinginkan serta direktori dimana citra akan disimpan.

Operasi konversi yang dilakukan ditentukan oleh kode argumen yang tercantum pada tabel di bawah ini.

Conversion code	Meaning
CV_BGR2RGB CV_RGB2BGR CV_RGBA2BGRA CV_BGRA2RGBA	Convert between RGB and BGR color spaces (with or without alpha channel)
CV_RGB2RGBA CV_BGR2BGRA	Add alpha channel to RGB or BGR image
CV_RGBA2RGB CV_BGRA2BGR	Remove alpha channel from RGB or BGR image
CV_RGB2BGRA CV_RGBA2BGR CV_BGRA2RGB CV_BGR2RGBA	Convert RGB to BGR color spaces while adding or removing alpha channel
CV_RGB2GRAY CV_BGR2GRAY	Convert RGB or BGR color spaces to grayscale
CV_GRAY2RGB CV_GRAY2BGR CV_RGBA2GRAY CV_BGRA2GRAY	Convert grayscale to RGB or BGR color spaces (optionally removing alpha channel in the process)
CV_GRAY2RGBA CV_GRAY2BGRA	Convert grayscale to RGB or BGR color spaces and add alpha channel
CV_RGB2BGR565 CV_BGR2BGR565 CV_BGR5652RGB CV_BGR5652BGR CV_RGBA2BGR565 CV_BGRA2BGR565 CV_BGR5652RGBA CV_BGR5652BGRA	Convert from RGB or BGR color space to BGR565 color representation with optional addition or removal of alpha channel (16-bit images)
CV_GRAY2BGR565 CV_BGR5652GRAY	Convert grayscale to BGR565 color representation or vice versa (16-bit images)

## Funksi OpenCV

Terdapat beberapa fungsi OpenCV yang telah kita gunakan, antara lain:

1. Fungsi imread  
Fungsi untuk membaca citra.  
`cv2.imread(filename)`
2. Fungsi cvtColor  
Fungsi untuk melakukan konversi citra  
`cv2.cvtColor(input_image, flag)`
3. Fungsi imwrite  
Fungsi untuk menyimpan citra baru.  
`cv2.imwrite(filename, image)`
4. Fungsi threshold  
Fungsi untuk memberikan threshold tertentu pada sebuah citra dengan menggunakan tipe threshold yang ada.  
`cv2.threshold(source, thresholdValue, maxVal, thresholdingTechnique)`

Kemungkinan nilai dan penjelasan dari paramater thresholdingTechnique adalah sebagai berikut.

- cv2.THRESH\_BINARY
- cv2.THRESH\_BINARY\_INV
- cv.THRESH\_TRUNC
- cv.THRESH\_TOZERO
- cv.THRESH\_TOZERO\_INV

$$\text{Binary} \quad \text{dst}(x, y) = \begin{cases} \text{maxval} & \text{if } \text{src}(x, y) > \text{thresh} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Inverted Binary} \quad \text{dst}(x, y) = \begin{cases} 0 & \text{if } \text{src}(x, y) > \text{thresh} \\ \text{maxval} & \text{otherwise} \end{cases}$$

$$\text{Truncated} \quad \text{dst}(x, y) = \begin{cases} \text{threshold} & \text{if } \text{src}(x, y) > \text{thresh} \\ \text{src}(x, y) & \text{otherwise} \end{cases}$$

$$\text{To Zero} \quad \text{dst}(x, y) = \begin{cases} \text{src}(x, y) & \text{if } \text{src}(x, y) > \text{thresh} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{To Zero Inverted} \quad \text{dst}(x, y) = \begin{cases} 0 & \text{if } \text{src}(x, y) > \text{thresh} \\ \text{src}(x, y) & \text{otherwise} \end{cases}$$

## **B. LATIHAN**

1. Membuat program untuk mengubah citra RGB menjadi citra grayscale kemudian disimpan ke dalam file baru dengan nama “output\_citra.jpg” dengan menggunakan *library* OpenCV pada proyek “ImageLoad” yang telah dikonfigurasi.

### **Grayscale.cpp**

```
import cv2

# buka citra
img = cv2.imread('image.jpg')

# assign row col
row, col, ch = img.shape

# buat salinan dari citra yang akan dijadikan grayscale img_gray =
img.copy()

# ubah color space dari img_gray menjadi grayscale img_gray =
cv2.cvtColor(img_gray, cv2.COLOR_BGR2GRAY)
```

2. Membuat dan menampilkan citra biner

### **Biner.cpp**

```
import cv2

# buka citra
img = cv2.imread('image.jpg')

# assign row col
row, col, ch = img.shape

# buat salinan dari citra yang akan dijadikan grayscale img_gray =
img.copy()

# ubah color space dari img_gray menjadi grayscale img_gray =
cv2.cvtColor(img_gray, cv2.COLOR_BGR2GRAY)

# fungsi threshold. Semua pixel yang bernilai di atas 100 # akan
diubah menjadi 255, selainnya menjadi 0.
ret, thresh = cv2.threshold(img_gray, 100, 255, cv2.THRESH_BINARY)

# tampilkan citra hasil cv2.imshow('threshold')
```

## A. Lembar Kerja Praktikum

Nama: Muhammad Farhadh

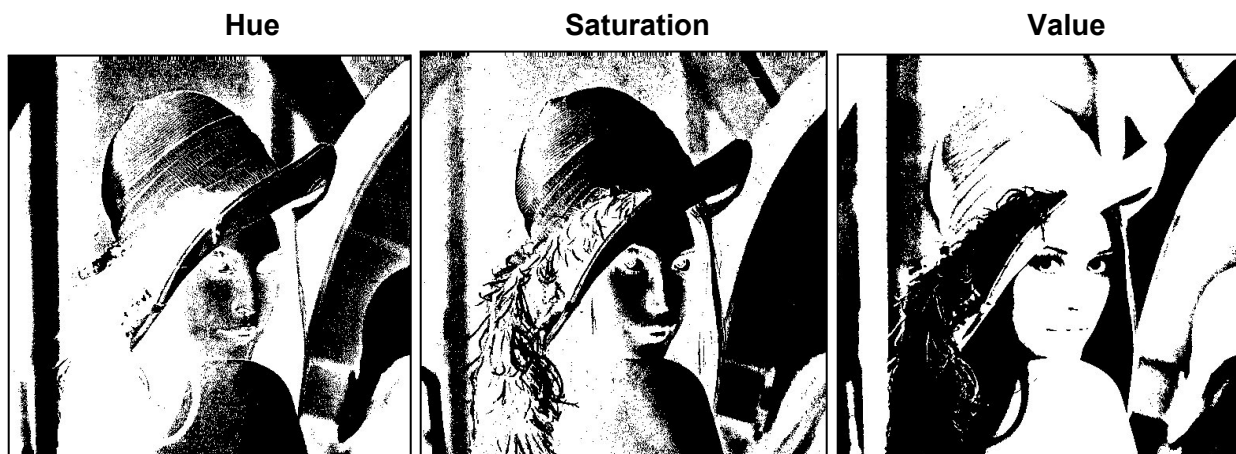
NIM: G640123080

Buatlah sebuah program menggunakan thresholding dengan algoritme sebagai berikut.

1. Baca sebuah citra (citra lenna.png) yang telah di sediakan di LMS.
2. Ubah citra tersebut menjadi HSV Color space.
3. Split citra tersebut menggunakan fungsi split (cv2.split).
4. Kemudian hitung nilai rata-rata dari intensitas pixel dari masing-masing channel.
5. Setelah itu lakukan thresholding pada masing-masing channel tersebut dengan ketentuan sebagai berikut:
  - Jika nilai intensitas piksel dibawah rata-rata, ubah menjadi 0.
  - Jika nilai intensitas piksel diatas rata-rata atau sama dengan rata-rata, ubah menjadi nilai 255.
6. Tampilkan hasil dari masing-masing channel. Setelah itu berikan penjelasan singkat terhadap citra yang dihasilkan.

**Notes:** Seluruh operasi dilakukan secara manual tanpa library, kecuali operasi dasar seperti `imwrite()`, `imshow()`, `imread()`

Output yang diharapkan



```

1 import numpy as np
2 import cv2
3
4 def bgrToHsv(bgrImg):
5     row, col, _ = bgrImg.shape;
6     hsvImg = bgrImg.copy();
7     for i in range(row):
8         for j in range(col):
9             b, g, r = bgrImg[i, j]/255;
10            cMax = np.maximum(np.maximum(r, g), b);
11            cMin = np.minimum(np.minimum(r, g), b);
12            delta = cMax - cMin
13
14            if cMax == r:
15                h = (60 * ((g - b) / delta) + 360) % 360
16            elif cMax == g:
17                h = (60 * ((b - r) / delta) + 120)
18            else:
19                h = (60 * ((r - g) / delta) + 240)
20            h = (h / 2).astype(np.uint8)
21
22            if cMax == 0:
23                s = 0;
24            else:
25                s = delta / cMax;
26            s = (s * 255).astype(np.uint8);
27            v = (cMax * 255).astype(np.uint8);
28
29            hsvImg[i, j] = h, s, v;
30        return hsvImg;
31
32 def threshold(img, th):
33     row, col = img.shape;
34     for i in range(row):
35         for j in range(col):
36             if img[i, j] < th:
37                 img[i, j] = 0;
38             else:
39                 img[i, j] = 255;
40     return img
41
42 img = cv2.imread("Lenna.png");
43 assert img is not None
44
45 hsv_img = bgrToHsv(img)
46 h, s, v = cv2.split(hsv_img);
47 rata2_h = float(np.mean(h));
48 rata2_s = float(np.mean(s));
49 rata2_v = float(np.mean(v));
50
51 h_2 = threshold(h, rata2_h);
52 s_2 = threshold(s, rata2_s);
53 v_2 = threshold(v, rata2_v);
54
55 cv2.imshow("HUE", h_2);
56 cv2.imshow("SATURATION", s_2);
57 cv2.imshow("VALUE", v_2);
58
59 cv2.waitKey(0);

```



#### Pengumpulan LKP :

1. Lampirkan fail .docx yang berisi program, penjelasan baris program, dan screenshot hasil gambar dengan format fail: **LKP2\_NIM>Nama** (Ex: LKP2\_G64140083\_Isfan Adila.docx)
2. Lampirkan file program python yang dibuat dengan format nama file: **LKP2\_NIM>Nama** (Ex: LKP2\_G64140083\_Isfan Adila.py)
3. Kedua file tersebut (.docx dan .py) disimpan dalam arsip .zip, dengan format fail: **LKP2\_NIM>Nama** (Ex: LKP2\_G64140083\_Isfan Adila.zip). Hanya fail .zip ini yang dikumpulkan.