LAPORAN TEORI

PENGOLAHAN CITRA DIGITAL

NAMA : M.Kahlil Ghibran

NIM : 202331066

KELAS : A

DOSEN : Dr. Dra. Dwina Kuswardani, M.Kom

NO.PC : 11

ASISTEN :

1. Clarenca Sweetdiva Pereira
2. Viana Salsabila Fairuz Syahla
3. Kashrina Masyid Azka
4. Sasikirana Ramadhanty Setiawan Putri

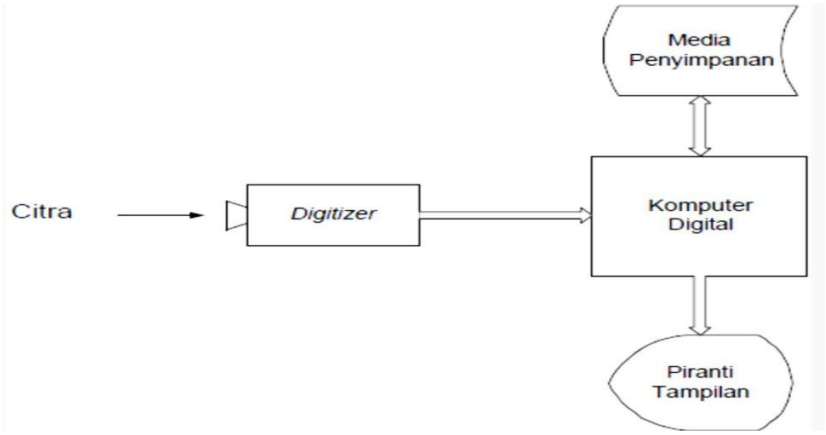
INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2025

1. Jelaskan alur dari gambar Elemen Pemrosesan Digital yang terlampir pada assignment ini.
2. Apa itu splitting citra?
3. Bagaimana cara men-splitting citra?
4. 0.2989 * r[i, j] + 0.587 * g[i, j] + 0.1141 * b[i, j]
   Apa maksud baris kode di atas?
5. Apakah kita bisa mengkonversi gambar RGB langsung ke gambar binary? Jika iya, berikan alasan. Jika tidak, tahapan apa dulu yang harus dilakukan dan mengapa?

1. **Alur dari Elemen Pemrosesan Digital**



1) **Citra (Image)** :
   - Proses dimulai dengan menggunakan citra asli, baik itu citra analog maupun digital, yang masuk ke dalam sistem pemrosesan.

2) **Digitizer** :
   - Digitizer adalah perangkat yang bertugas mengubah citra analog menjadi citra digital melalui dua proses utama:
     - **Sampling** : Proses pembagian area citra menjadi grid piksel kecil.
     - **Kuantisasi** : Proses konversi intensitas cahaya menjadi nilai numerik diskrit.
   - Setelah melalui digitizer, citra analog diubah menjadi citra digital yang dapat diproses oleh computer.

3) **Komputer Digital** :
   - Citra digital kemudian diteruskan ke computer digital untuk dilakukan berbagai operasi pemrosesan.
   - Komputer digital melakukan manipulasi data citra menggunakan algoritma dan Teknik pemrosesan citra seperti perbaikan kontras, filtering, segmentasi, atau transformasi lainnya.

4) **Media Penyimpanan** :
   - Hasil dari pemrosesan citra disimpan dalam media penyimpanan.

5) **Piranti Tampilan** :
   - Setelah pemrosesan selesai, hasil akhir ditampilkan kepada pengguna melalui piranti tampilan seperti monitor, proyektor, atau printer.

2. **Splitting Citra**

Splitting citra adalah proses pemisahan sebuah citra digital menjadi beberapa bagian atau komponen yang lebih kecil dengan tujuan untuk memudahkan analisis, pengolahan, atau pengenalan pola dalam bidang pengolahan citra digital (digital image processing). Dalam konteks ilmu komputer dan teknologi pengolahan citra, splitting citra sering digunakan dalam berbagai aplikasi seperti segmentasi gambar, kompresi citra, pengenalan objek, dan analisis fitur. Proses ini bertujuan untuk memecah citra menjadi beberapa region atau blok yang lebih sederhana agar informasi yang terkandung dalam citra tersebut lebih mudah dipahami dan diolah oleh sistem.

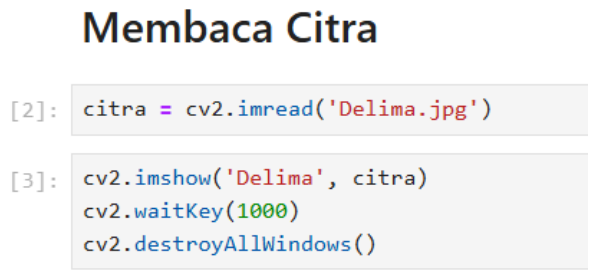
Splitting citra dalam pengolahan citra digital dapat didefinisikan sebagai proses memecah gambar menjadi bagian-bagian atau region yang lebih kecil berdasarkan karakteristik tertentu, seperti intensitas warna, tekstur, atau pola geometris. Proses ini penting untuk memudahkan analisis lebih lanjut, seperti pengenalan objek, pengukuran fitur, dan ekstraksi informasi. Dalam konteks akademik, mahasiswa diajarkan teknik-teknik splitting citra agar dapat memahami cara memisahkan objek dalam gambar secara otomatis menggunakan algoritma yang efektif dan efisien.

3. **Cara men-splitting citra**

Proses splitting dapat dilakukan degan cara memisahkan masing-masing saluran warna menjadi citra grayscale terpisah.

Langkah-langkah:

1) **Baca Citra :**
   Gunakan OpenCV untuk membaca citra.



2) **Memisahkan Saluran Warna :**
Gunakan slicing array NumPy untuk memisahkan masing-masing saluran warna.

4. **0.2989 * r[i, j] + 0.587 * g[i, j] + 0.1141 * b[i, j]**

Baris kode tersebut adalah rumus untuk konversi citra RGB ke Grayscale. Rumus ini menggunakan bobot tertentu untuk setiap saluran warna (merah, hijau, biru) untuk menghasilkan nilai intensitas grayscale yang sesuai. Bobot-bobot tersebut didasarkan pada sensitivitas mata manusia terhadap setiap warna:

- **Merah (Red)** : Bobot 0.2989
- **Hijau (Green)** : Bobot 0.587
- **Biru (Blue)** : Bobot 0.1141

Penjelasan:

- r[i,j], g[i,j], dan b[i,j] merepresentasikan nilai intensitas piksel pada saluran merah, hijau, dan biru, masing-masing.
- Nilai grayscale dihasilkan dengan menjumlahkan kontribusi dari setiap saluran warna berdasarkan bobot yang telah ditentukan.

5. **Apakah bisa mengonversi gambar RGB langsung ke binary?**

Tidak, kita tidak bisa langsung mengonversi gambar RGB ke gambar binary karena RGB memiliki tiga saluran warna (merah, hijau, biru), sedangkan gambar binary hanya memiliki dua nilai (0 atau 1).

LAPORAN PRAKTIKUM

PENGOLAHAN CITRA DIGITAL

NAMA : M.Kahlil Ghibran

NIM : 202331066

KELAS : A

DOSEN : Dr. Dra. Dwina Kuswardani, M.Kom

NO.PC : 11

ASISTEN :

1. Clarenca Sweetdiva Pereira
2. Viana Salsabila Fairuz Syahla
3. Kashrina Masyid Azka
4. Sasikirana Ramadhanty Setiawan Putri

INSTITUT TEKNOLOGI PLN

TEKNIK INFORMATIKA

2025

Langkah-langkah Praktikum:

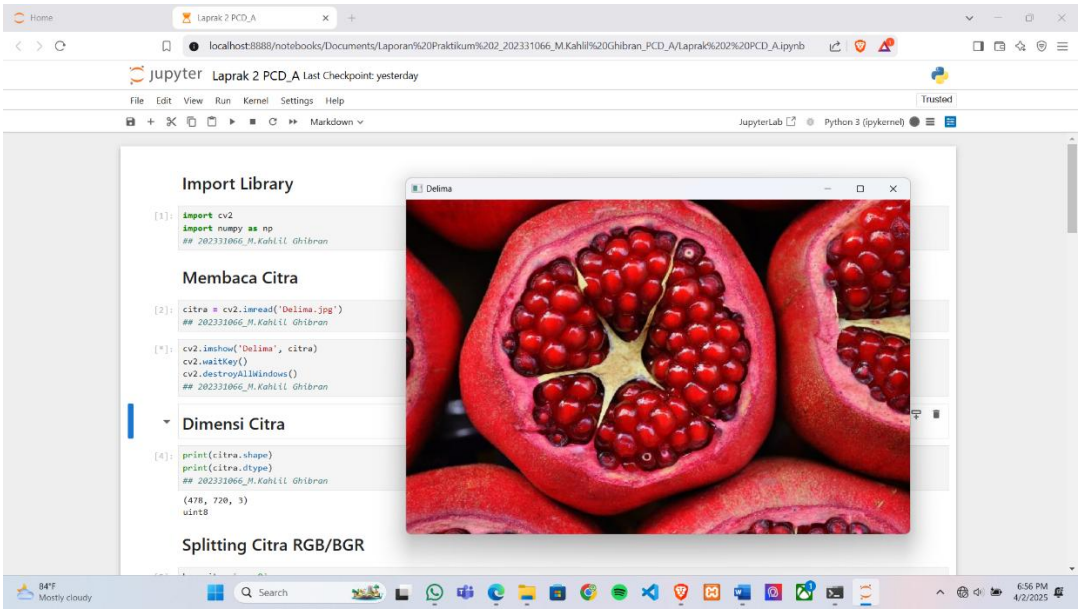
Import Library

```
[1]: import cv2
     import numpy as np
     ## 202331066_M.Kahlil Ghibran
```

1.

Mengimport library OpenCV dan Numpy yang dimana cv2 digunakan untuk membaca, menampilkan, dan melakukan operasi pada citra, sedangkan numpy (np) digunakan umtuk manipulasi array multidimensi yang sangat berguna dalam pengolahan citra.

2. Membaca dan Menampilkan Citra



- Fungsi cv2.imread() digunakan untuk membaca file citra dengan nama ‘Delima.jpg’.
- Citra yang dibaca akan disimpan dalam variabel citra dalam format BGR (Blue-Green-Red).
- cv2.imshow() digunakan untuk menampilkan citra dengan judul jendela ‘Delima’.
- cv2.waitKey() digunakan untuk menunggu input keyboard sebelum jendela ditutup.
- cv2.destroyAllWindows() digunakan untuk menghancurkan semua jendela yang terbuka setelah pengguna menekan tombol.

3. Dimensi Citra

Dimensi Citra

```
[4]: print(citra.shape)
     print(citra.dtype)
     ## 202331066_M.Kahlil Ghibran
```

```
(478, 720, 3)
uint8
```

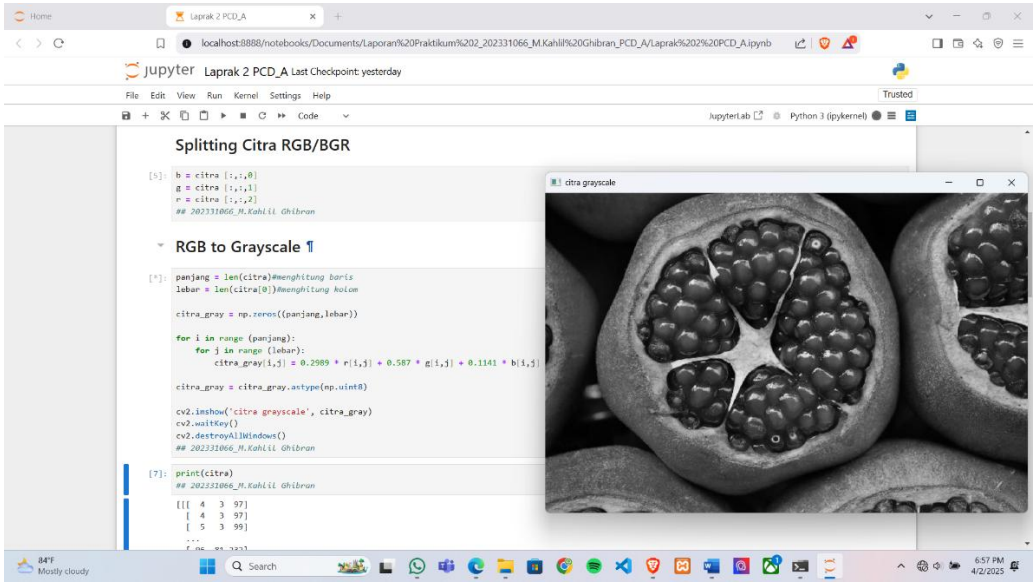
- citra.shape : Menampilkan dimensi citra, yaitu (tinggi, lebar, channel). Misalnya, (478, 720, 3) berarti citra memiliki tinggi 478 piksel, lebar 720 piksel, dan 3 channel (BGR).
- citra.dtype : Menampilkan tipe data pixel citra, biasanya uint8 (unsigned integer 8-bit)

4. Memisahkan Citra RGB/BGR

Splitting Citra RGB/BGR

```
[5]: b = citra [:,:,0]
     g = citra [:,:,1]
     r = citra [:,:,2]
     ## 202331066_M.Kahlil Ghibran
```

5. Konversi Citra RGB/BGR ke Grayscale



- Pada tahap inisialisasi, dimensi citra dihitung menggunakan variabel panjang (jumlah baris) dan lebar (jumlah kolom). Selain itu, sebuah array kosong bernama citra_gray dibuat untuk menyimpan nilai piksel grayscale.

- Proses konversi dilakukan dengan menggunakan dua loop bersarang untuk mengiterasi setiap piksel pada citra. Setiap piksel dikonversi ke grayscale menggunakan rumus yang memberikan bobot tertentu kepada setiap komponen warna (merah, hijau, dan biru). Bobot ini mencerminkan kontribusi masing-masing warna terhadap luminance grayscale.
- Setelah proses konversi selesai, hasil citra grayscale ditampilkan menggunakan fungsi cv2.imshow. Program kemudian menunggu input tombol dari pengguna sebelum menutup jendela tampilan dengan menggunakan cv2.waitKey() dan cv2.destroyAllWindows().
- **Print(citra)** : Digunakan untuk menampilkan isi dari variabel citra.

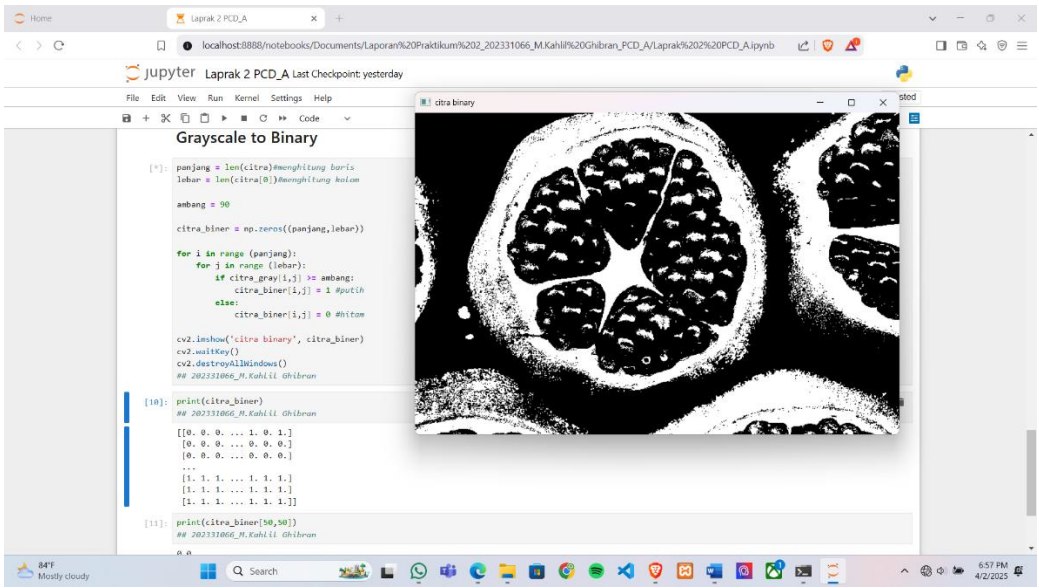
6. **Print(citra_gray)**

```
[8]: print(citra_gray)
     ## 202331066_M.Kahlil Ghibran
```

```
[[ 31  31  31 ... 127  87  99]
 [ 30  30  31 ...  75  78  62]
 [ 27  27  28 ...  55  51  84]
 ...
 [133 128 128 ... 110 106 100]
 [141 135 133 ... 111 107 102]
 [145 139 137 ... 109 107 107]]
```

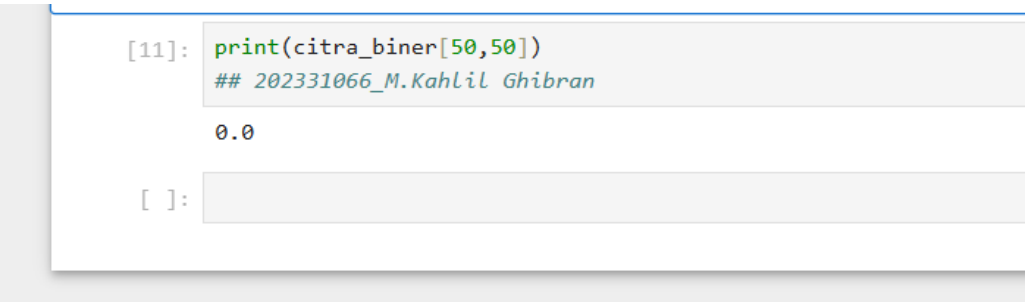
- Ketika menggunakan perintah **print(citra_gray)**, isi dari variabel **citra_gray** akan dicetak ke layar. Output yang ditampilkan berupa matriks dua dimensi, di mana setiap elemen dalam matriks mewakili intensitas piksel pada citra grayscale. Nilai intensitas ini berkisar antara 0 hingga 255, dengan 0 menunjukkan warna hitam (gelap) dan 255 menunjukkan warna putih (terang). Jika ukuran matriks terlalu besar, Python hanya akan menampilkan sebagian isi matriks, dan tanda **...** digunakan untuk menandakan bahwa ada nilai lain yang tidak ditampilkan secara lengkap.

7. Konversi Citra Grayscale ke Binary



- Pada tahap awal, dimensi citra dihitung menggunakan fungsi **len()** untuk mendapatkan jumlah baris (tinggi) dan kolom (lebar) citra. Kemudian, sebuah nilai ambang ditetapkan sebesar 20, yang digunakan sebagai batas untuk mengklasifikasikan piksel menjadi hitam atau putih. Array kosong bernama **citra_biner** dibuat dengan ukuran yang sama dengan citra asli, untuk menyimpan hasil konversi citra biner.
- Proses konversi dilakukan dengan mengiterasi setiap piksel pada citra grayscale. Jika nilai piksel lebih besar atau sama dengan nilai ambang, piksel diubah menjadi 1 (putih); jika lebih kecil, piksel diubah menjadi 0 (hitam). Setelah proses konversi selesai, citra biner ditampilkan menggunakan fungsi **cv2.imshow**. Program kemudian menunggu input dari pengguna sebelum menutup semua jendela tampilan menggunakan **cv2.waitKey()** dan **cv2.destroyAllWindows()**.
- **Print(citra_biner) :** Digunakan untuk menampilkan isi dari variabel citra_biner.

8. Terakhir



- Perintah **print(citra_biner[50, 50])** digunakan untuk menampilkan nilai piksel pada posisi baris ke-50 dan kolom ke-50 dalam matriks **citra_biner**. Output yang dihasilkan adalah 1.0 , yang berarti piksel pada posisi tersebut memiliki nilai aktif atau berwarna putih dalam citra biner. Dalam citra biner, nilai 1.0 biasanya digunakan untuk merepresentasikan piksel yang terang (putih), sedangkan nilai 0.0 digunakan untuk piksel yang gelap (hitam).