

**MAKALAH ALJABAR LINEAR**  
**“CITRA DIGITAL”**



**DOSEN PENGAMPU:**  
**NATALIS RANSI, S.Si., M.Cs**

**DISUSUN OLEH:**  
**LA ODE MUHAMMAD YUDHY PRAYITNO**  
**E1E122064**

**JURUSAN TEKNIK INFORMATIKA**  
**FAKULTAS TEKNIK**  
**UNIVERSITAS HALU OLEO**  
**KENDARI**  
**2023**

## KATA PENGANTAR

Puji dan syukur kita panjatkan kehadirat Allah SWT. karena dengan limpahan rahmat dan karunia yang diberikan sehingga penyusunan makalah Aljabar Linear dengan judul “Citra Digital” ini dapat terselesaikan dengan baik. Makalah ini saya susun sebagai salah satu syarat untuk memenuhi tugas mata kuliah Aljabar Linear.

Shalawat serta salam tak lupa pula untuk Baginda Nabi Allah Muhammad SAW, karena berkat beliau kita yang dahulu berada di zaman kegelapan, dan sekarang berada di zaman yang terang benderang serta zaman dimana teknologi sudah semakin canggih dan berkembang pesat.

Dalam proses penyusunan makalah ini, saya mendapat bantuan dari berbagai pihak baik langsung maupun tidak langsung. Oleh karenanya, dengan segala kerendahan hati saya menyampaikan ucapan terima kasih dan penghargaan yang setinggi-tingginya kepada berbagai pihak tersebut, terutama kepada Bapak Natalis Ransi, S.Si., M.Cs., selaku dosen pengampu mata kuliah Aljabar Linear yang telah menambah wawasan saya karena telah memberikan saya tugas dan materi ini.

Saya menyadari bahwa dalam penyusunan makalah ini masih banyak kekurangan, oleh sebab itu saya sangat mengharapkan kritik dan saran yang membangun.

Akhirnya, saya berharap semoga makalah ini dapat bermanfaat bagi khalayak umum serta pula semoga segala ilmu, bantuan dan dorongan yang telah diberikan dapat bernilai ibadah dan memperoleh balasan yang berlipat ganda dari Allah SWT. Aamiin.

Kendari, 21 Mei 2023

Penyusun

## DAFTAR ISI

KATA PENGANTAR.....	ii
DAFTAR ISI.....	iii
DAFTAR GAMBAR .....	v
DAFTAR TABEL .....	vi
BAB I .....	1
PENDAHULUAN.....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	1
1.3 Tujuan.....	1
BAB II.....	2
PEMBAHASAN .....	2
2.1 Citra.....	2
2.1.1 Citra Digital.....	2
2.1.2 Tipe Citra Digital.....	4
2.1.3 Kuantitas Citra.....	6
2.1.4 Kualitas Citra.....	9
2.2 Computer Vision .....	10
2.3 Pengolahan Citra Digital .....	11
2.3.1 Operasi Piksel.....	11
2.3.2 Metode Ubah Nilai RGB ke Grayscale .....	12
2.3.3 Histogram Citra .....	13
2.3.4 Meningkatkan Kecerahan.....	17
BAB III.....	20
PENERAPAN .....	20
3.1 Persiapan Bahan .....	20
3.2 Praktek.....	20
3.1.1 Langkah Setup Google Colab .....	20

3.1.2 Langkah Operasi Citra Digital.....	21
BAB IV .....	36
PENUTUP.....	36
4.1 Kesimpulan.....	36
4.2 Saran.....	36
DAFTAR PUSTAKA .....	37

## DAFTAR GAMBAR

Gambar 2. 1. Representasi Citra Digital 2 Dimensi.....	3
Gambar 2. 2 Ilustrasi digitalisasi citra (pixel pada koordinat $x=10$ , $y=3$ memiliki nilai 110) .....	3
Gambar 2. 3. Representasi Citra Biner.....	4
Gambar 2. 4. Citra Kapal .....	5
Gambar 2. 5. Citra Pepper.....	6
Gambar 2. 6 Perbandingan isyarat analog dan isyarat diskret .....	6
Gambar 2. 7 Digitalisasi citra biner 8x8 piksel.....	7
Gambar 2. 8 Kuantisasi citra dengan menggunakan berbagai bit.....	8
Gambar 2. 9 Efek resolusi berdasar jumlah piksel pada citra .....	9
Gambar 2. 11. Matriks Citra Digital .....	14
Gambar 2. 12. Histogram Citra Digital.....	15
Gambar 2. 13 Histogram pada citra berwarna secara menyeluruh (I), merah (R), hujau (G), dan biru (B).....	16
Gambar 2. 14 Empat buah citra (a), (b),(c), dan (d) yang memiliki histogram yang sama (e), tetapi mempunyai informasi yang jauh berbeda .....	17
Gambar 2. 15 Efek pencerahan gambar .....	18
Gambar 2. 16 Histogram pada peningkatan citra. Komposisi jumlah intensitas ..	18
Gambar 2. 17 Peningkatan kecerahan pada citra berwarna .....	19

## DAFTAR TABEL

Tabel 2. 1. Garis Besar <i>Human Vision</i> dan <i>Computer Vision</i> .....	10
Tabel 2. 2. Perhitungan Histogram .....	14

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Tak dapat dipungkiri, bahwa perkembangan teknologi pengolahan citra dewasa ini berkembang dengan sangat pesat, baik itu perkembangan jumlah pemakai maupun perkembangan jenis teknologi yang menggunakan pengolahan citra, seperti misalnya bidang biomedis, astronomi, penginderaan jauh, dan arkeologi yang umumnya banyak memerlukan teknik peningkatan mutu citra. Aplikasi lain yang kemudian menyusul adalah pengolahan citra digital di bidang robotika, industri, serta arsip citra dan dokumen.

Proses pengolahan citra yang termasuk dalam kategori peningkatan mutu citra bertujuan untuk memperoleh keindahan gambar, untuk kepentingan analisis citra, dan untuk mengoreksi citra dari segala gangguan yang terjadi pada waktu perekaman data.

### **1.2 Rumusan Masalah**

Adapun rumusan masalah dari makalah Aljabar Linear “Citra Digital” ini adalah sebagai berikut:

1. Apa itu citra?
2. Apa itu citra digital?
3. Apa saja jenis atau tipe dari citra digital?
4. Bagaimana metode pengolahan citra digital?

### **1.3 Tujuan**

Adapun tujuan dari makalah Aljabar Linear “Citra Digital” ini adalah sebagai berikut:

1. Untuk mengetahui apa itu citra.
2. Untuk mengetahui apa itu citra digital.
3. Untuk mengetahui jenis atau tipe dari citra digital.
4. Untuk mengetahui metode pengolahan citra digital.

## **BAB II**

### **PEMBAHASAN**

#### **2.1 Citra**

Terdapat istilah “citra” yang dalam Bahasa Inggris disebut *image*. Secara harfiah, citra (*image*) adalah gambar pada bidang dwimatra (dua dimensi). Contoh citra antara lain adalah foto dan lukisan.

Citra ada dua macam, yaitu citra kontinu dan citra diskrit. Citra kontinu dihasilkan dari sistem optik yang menerima sinyal analog, misalnya mata manusia dan kamera analog. Citra diskrit dihasilkan melalui proses digitalisasi terhadap citra kontinu. Beberapa sistem optik dilengkapi dengan fungsi digitalisasi sehingga ia mampu menghasilkan citra diskrit, misalnya kamera digital dan *scanner*. Citra diskrit disebut juga citra digital. Komputer digital yang umum dipakai saat ini hanya dapat mengolah citra digital.

##### **2.1.1 Citra Digital**

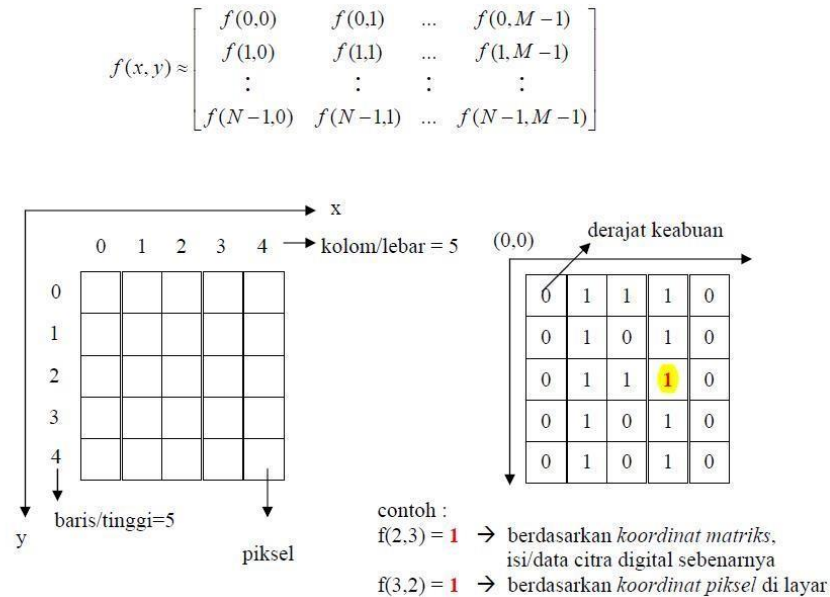
Agar dapat diolah dengan menggunakan komputer digital, maka suatu citra harus direpresentasikan secara numerik dengan nilai-nilai diskrit. Proses representasi citra kontinu menjadi nilai-nilai diskrit disebut digitalisasi. Citra digital adalah hasil proses digitalisasi.

Suatu citra digital mempunyai fungsi dua dimensi  $f(x,y)$  yang memiliki ukuran  $M$  baris dan  $N$  kolom dimana  $x$  dan  $y$  adalah koordinat pada bidang dwimatra dan  $f(x,y)$  adalah intensitas cahaya (*brightness*) atau amplitudo atau derajat keabuan (*gray level*).

Citra digital merupakan suatu matriks dimana indeks baris dan kolomnya menyatakan suatu titik pada citra tersebut dan elemen matriksnya (yang disebut sebagai elemen gambar atau *pixel*) menyatakan nilai derajat keabuan pada titik tersebut. Citra digital berukuran  $N \times M$  (tinggi =  $N$ , lebar =  $M$ ) dinyatakan dengan matriks  $N \times M$ . Bentuk matriks citra digital dapat dilihat pada gambar 2.1.

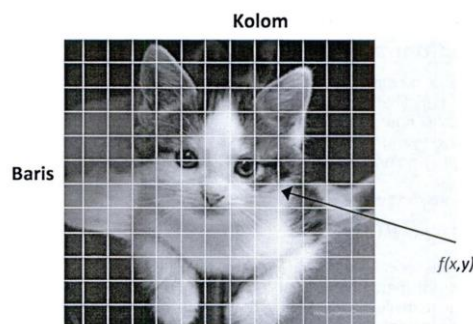


$$\begin{aligned}
 N &= \text{jumlah baris} & 0 \leq y \leq N - 1 \\
 M &= \text{jumlah kolom} & 0 \leq x \leq M - 1 \quad \dots
 \end{aligned}
 \tag{2.1}$$



Gambar 2. 1. Representasi Citra Digital 2 Dimensi

Citra digital pada umumnya berbentuk empat persegi panjang dengan dimensi ukurannya tinggi x lebar. Dimensi citra digital berisi blok-blok kecil yang berupa elemen gambar yang disebut pixel. Pixel-pixel tersebut memuat informasi warna yang menyusun suatu citra. Untuk menunjukkan tingkat pencahayaan pixel, digunakan bilangan bulat yang besarnya 8 *bit* (1 *byte*) untuk setiap pixelnya, dengan rentang antara 0-255, dimana 0 untuk warna hitam, 255 untuk warna putih, dan tingkat keabuan ditandai dengan nilai antara 0-255.



Gambar 2. 2 Ilustrasi digitalisasi citra (pixel pada koordinat  $x=10$ ,  $y=3$  memiliki nilai 110)

### 2.1.2 Tipe Citra Digital

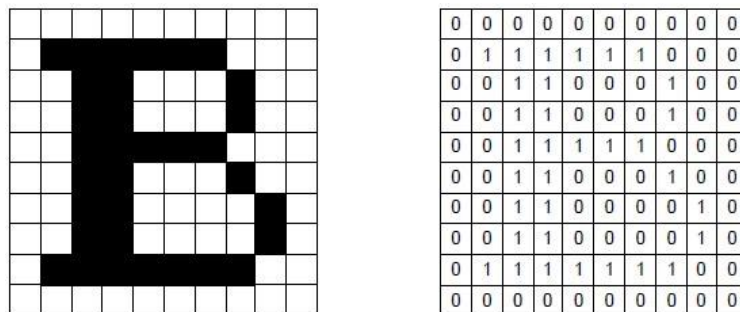
Berdasarkan format penyimpanan nilai warnanya, citra digital terbagi dalam tiga tipe yaitu:

#### 1. Citra Biner

Citra biner (*binary image*) adalah citra yang hanya mempunyai dua nilai derajat keabuan: hitam dan putih. *Pixel* yang bernilai 1 melambangkan warna hitam, sedangkan *pixel* yang bernilai 0 adalah warna putih. Dinyatakan dalam suatu fungsi:

$$f(x,y) \in \{0,1\} \quad \dots (2.2)$$

Berikut ini adalah contoh citra biner dan representasi nilai tiap *pixel* seperti pada gambar 2.2.



Gambar 2. 3. Representasi Citra Biner

#### 2. Citra *Grayscale* (Skala Keabuan)

Citra yang terdiri dari satu layer warna dengan derajat keabuan tertentu. Untuk kebanyakan citra digital 8-bit, maka sistem Grey-scale diukur berdasarkan skala intensitas kecerahan, yang bernilai 0 – 255, dimana yang hitam pekat adalah 0 dan yang terputih adalah 255. Dinyatakan dalam suatu fungsi:

$$f(x,y) \in [0...255] \quad \dots (2.3)$$

Gambar 2.4 menunjukkan contoh citra skala keabuan 8-bit dengan ukuran 512 x 512 pixel [8].



Gambar 2. 4. Citra Kapal

### 3. Citra Berwarna

RGB adalah suatu model warna yang terdiri dari merah, hijau, dan biru, digabungkan dalam membentuk suatu susunan warna yang luas. Setiap warna dasar, misalnya merah, dapat diberi rentang-nilai. Untuk monitor komputer, nilai rentangnya paling kecil = 0 dan paling besar = 255. Pilihan skala 256 ini didasarkan pada cara mengungkap 8 digit bilangan biner yang digunakan oleh mesin komputer. Dengan cara ini, akan diperoleh warna campuran sebanyak  $256 \times 256 \times 256 = 16.777.216$  jenis warna.

Sebuah jenis warna, dapat dibayangkan sebagai sebuah vektor di ruang 3 dimensi yang biasanya dipakai dalam matematika, koordinatnya dinyatakan dalam bentuk tiga bilangan, yaitu komponen-x, komponen-y dan komponen-z. Misalkan sebuah vektor dituliskan sebagai  $r = (x,y,z)$ . Untuk warna, komponen-komponen tersebut digantikan oleh komponen R (Red), G (Green), B (Blue). Jadi, sebuah jenis warna dapat dituliskan sebagai berikut: warna = RGB(xxx, xxx, xxx). Putih = RGB(255,255,255), sedangkan untuk hitam= RGB(0,0,0). Gambar 2.4 menunjukkan contoh citra berwarna 8-bit. Representasi dalam citra digital dinyatakan dalam persamaan:

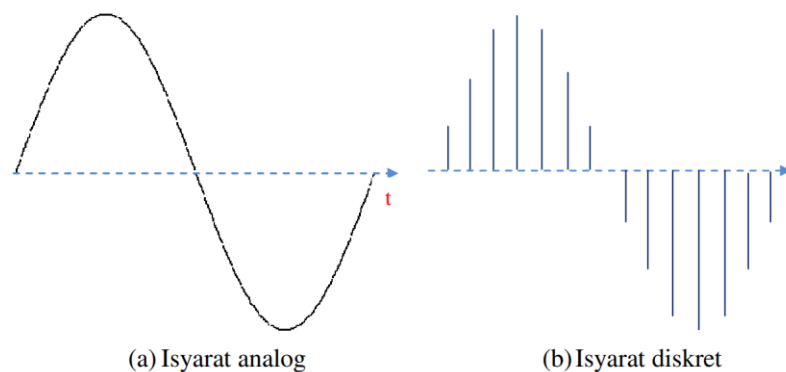
$$\begin{aligned} fR(x,y) &\Sigma [0...255] \\ fG(x,y) &\Sigma [0...255] \\ fB(x,y) &\Sigma [0...255] \end{aligned} \quad \dots (2.4)$$



Gambar 2. 5. Citra Pepper

### 2.1.3 Kuantitas Citra

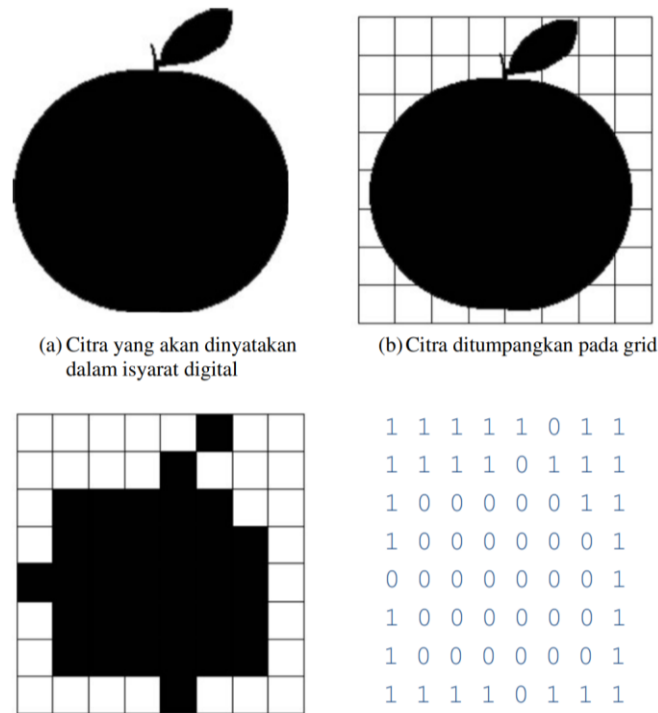
Citra digital sesungguhnya dibentuk melalui pendekatan yang dinamakan kuantisasi. Kuantisasi adalah prosedur yang dipakai untuk membuat suatu isyarat yang bersifat kontinu ke dalam bentuk diskret. Untuk mempermudah pemahaman konsep ini, lihatlah Gambar 2.6. Gambar 2.6(a) menyatakan isyarat analog menurut perjalanan waktu  $t$ , sedangkan Gambar 2.6(b) menyatakan isyarat diskret



Gambar 2. 6 Perbandingan isyarat analog dan isyarat diskret

Pada isyarat digital, nilai intensitas citra dibuat diskret atau terkuantisasi dalam sejumlah nilai bulat. Gambar 2.7(a) menunjukkan contoh citra biner dua nilai intensitas berupa 0 (hitam) dan 1 (putih). Selanjutnya, gambar tersebut ditumpangkan pada grid 8x8 seperti yang diperlihatkan pada Gambar 2.7(b). Bagian gambar yang jatuh pada kotak kecil dengan luas lebih kecil dibanding warna putih latar belakang, seluruh isi kotak dibuat putih. Sebaliknya, jika mayoritas hitam, isi

kotak seluruhnya dibuat hitam. Hasil pengubahan ke citra digital tampak pada Gambar 2.7(c). Adapun Gambar 2.7(d) memperlihatkan bilangan yang mewakili warna hitam (0) dan putih (1). Dengan demikian, citra digital akan lebih baik (lebih sesuai aslinya) apabila ukuran piksel diperkecil atau jumlah piksel diperbanyak.



Gambar 2. 7 Digitalisasi citra biner 8x8 piksel

Bagaimana halnya kalau gambar mengandung unsur warna (tidak sekadar hitam dan putih)? Prinsipnya sama saja, tetapi sebagai pengecualian, warna hitam diberikan tiga unsur warna dasar, yaitu merah (R = red), hijau (G = green), dan biru (B = blue). Seperti halnya pada citra monokrom (hitam-putih) standar, dengan variasi intensitas dari 0 hingga 255, pada citra berwarna terdapat 16.777.216 variasi warna apabila setiap komponen R, G, dan B mengandung 256 aras intensitas. Namun, kepekaan mata manusia untuk membedakan macam warna sangat terbatas, yakni jauh di bawah enam belas juta lebih tersebut. Untuk beberapa keperluan tertentu, jumlah gradasi intensitas saling berbeda. Tabel 2.1 memberikan lima contoh untuk citra beraras keabuan dan Tabel 2.2 menunjukkan empat contoh penggunaan citra berwarna (RGB). Perhatikan bahwa jumlah gradasi juga bisa dinyatakan dalam jumlah digit biner atau bit 0 dan 1 sebagai sandi digital per piksel.

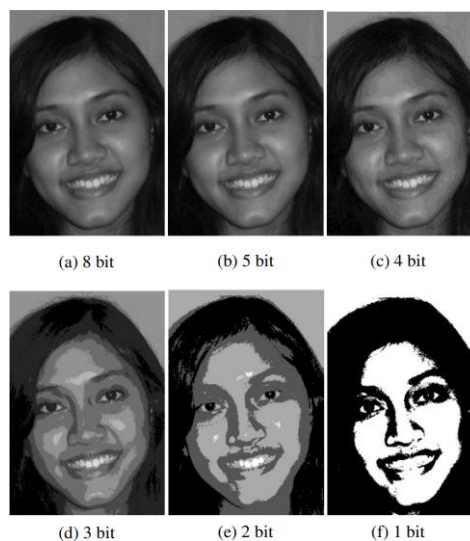
Tabel 2. 1 Jangkauan nilai pada citra keabuan

Komponen warna	Bit per Piksel	Jangkauan	Penggunaan
1	1	0-1	Citra biner: dokumen faksimili
	8	0-255	Umum: foto dan hasil pemindai
	12	0-4095	Kualitas tinggi: foto dan hasil pemindai
	14	0-16383	Kualitas profesional: foto dan hasil pemindai
	16	0-65535	Kualitas tertinggi: citra kedokteran dan astronomi

Tabel 2. 2 Jangkauan nilai pada citra berwarna

Komponen Warna	Bit per Piksel	Jangkauan	Penggunaan
3	24	0-1	RGB umum
	36	0-4095	RGB kualitas tinggi
	42	0-16383	RGB kualitas profesional
4	32	0-255	CMYK (cetakan digital)

Dalam pengolahan citra, kuantisasi aras intensitas menentukan kecermatan hasilnya. Dalam praktik, jumlah aras intensitas piksel dapat dinyatakan dengan kurang dari 8 bit. Contoh pada Gambar 2.8 menunjukkan citra yang dikuantisasi dengan menggunakan 8, 5, 4, 3, 2, dan 1 bit.



Gambar 2. 8 Kuantisasi citra dengan menggunakan berbagai bit

Pada kuantisasi dengan 1 bit, jumlah level sebanyak 2 (2<sup>1</sup>). Oleh karena itu, warna yang muncul berupa hitam dan putih saja. Perlu diketahui, penurunan jumlah aras pada tingkat tertentu membuat mata manusia masih bisa menerima citra dengan baik. Sebagai contoh, citra dengan 4 bit (Gambar 2.8(c)) dan citra dengan 8 bit (Gambar 2.8(a)) praktistertlihat sama. Hal seperti itulah yang mendasari gagasan pemampatan data citra, mengingat citra dengan jumlah bit lebih rendah tentu akan membutuhkan tempat dan transmisi yang lebih hemat.

#### 2.1.4 Kualitas Citra

Di samping cacah intensitas kecerahan, jumlah piksel yang digunakan untuk menyusun suatu citra mempengaruhi kualitas citra. Istilah resolusi citra biasa dinyatakan jumlah piksel pada arah lebar dan tinggi. Resolusi piksel biasa dinyatakan dengan notasi  $m \times n$ , dengan  $m$  menyatakan tinggi dan  $n$  menyatakan lebardalam jumlah piksel. Contoh pada Gambar 2.5 menunjukkan bahwa kalau gambar apel hanya dinyatakan dalam  $8 \times 8$  piksel, citra yang terbentuk sangat berbeda dengan aslinya. Seandainya jumlah piksel yang digunakan lebih banyak, tentu akan lebih mendekati dengan gambar aslinya. Contoh pada Gambar 2.9 memperlihatkan efek resolusi piksel untuk menampilkan gambar yang sama.



Gambar 2. 9 Efek resolusi berdasar jumlah piksel pada citra

Terlihat bahwa pada resolusi tertentu citra menjadi kabur kalau dinyatakan dengan jumlah piksel yang makin sedikit. Resolusi spasial ditentukan oleh jumlah piksel per satuan panjang. Istilah seperti dpi (dot per inch) menyatakan jumlah piksel per inci. Misalnya, citra 300 dpi menyatakan bahwa citra akan dicetak dengan jumlah piksel sebanyak 300 sepanjang satu inci. Berdasarkan hal itu, maka citra dengan resolusi ruang spasial sebesar 300 dpi dicetak di kertas dengan ukuran lebih kecil daripada yang mempunyai resolusi ruang sebesar 150 dpi, meskipun kedua gambar memiliki resolusi piksel yang sama.

## 2.2 Computer Vision

Pada hakikatnya, *computer vision* mencoba meniru cara kerja sistem visual manusia. Dalam proses penglihatan manusia, manusia melihat objek dengan menggunakan indera penglihatan yang berupa mata, lalu citra objek diteruskan ke otak untuk diinterpretasikan sehingga manusia mengerti objek apa yang tampak. Hasil interpretasi ini kemudian digunakan untuk pengambilan keputusan.

Tabel 2. 3. Garis Besar *Human Vision* dan *Computer Vision*

<i>Human Vision</i>	<i>Computer Vision</i>
Menggunakan mata dan <i>visual cortex</i> di dalam otak.	Menggunakan kamera-kamera yang terhubung pada sistem komputer.
Menemukan dari gambar objek apa yang ada dalam penglihatan dimana posisinya, bagaimana mereka bergerak, dan apa bentuknya.	Secara otomatis menginterpretasi gambar-gambar dan mencoba untuk mengerti isinya seperti pada <i>human vision</i> .

Secara garis besar, *computer vision* adalah sebuah teknologi mesin yang mampu mengenali objek yang diamati. Kemampuan untuk mengenali ini merupakan kombinasi dari pengolahan citra dan pengenalan pola. Pengolahan citra adalah proses awal dalam *computer vision* untuk menghasilkan citra yang lebih baik atau lebih mudah diinterpretasikan, sedangkan pengenalan pola adalah proses identifikasi objek pada citra. Proses-proses dalam *computer vision* secara garis besar dapat dibagi menjadi:

1. Proses mengakuisisi citra digital (*Image Acquisition*)
2. Proses pengolahan citra (*Image Processing*)
3. Proses analisis data citra (*Image Analysis*)



#### 4. Proses pemahaman data citra (*Image Understanding*)

*Computer vision* adalah sebuah tantangan untuk mencoba meniru indera manusia yang paling kuat. Bidang ini memiliki sejumlah aplikasi dalam dunia industri, antara lain:

1. Otomatisasi Proses Industri
2. Obat-obatan dan Diagnostik
3. Hiburan: Film dan Video
4. Keamanan dan Pengawasan (*Surveillance*)
5. Visualisasi dan *Augmented Reality*
6. Komunikasi
7. Interaksi Manusia dengan Komputer
8. Kemiliteran dan Penelitian Ruang Angkasa

### 2.3 Pengolahan Citra Digital

Pengolahan citra adalah pemrosesan citra dengan menggunakan komputer agar kualitasnya lebih baik. Tujuannya agar citra dapat dengan mudah diinterpretasi oleh manusia atau komputer. Teknik pengolahan citra memanipulasi citra menjadi citra lain. Jadi, masukannya adalah citra dan keluarannya juga citra. Namun citra keluaran mempunyai kualitas lebih baik daripada citra masukan.

Pada umumnya, operasi-operasi pada pengolahan citra diterapkan pada citra bila:

- a. Perbaikan atau modifikasi citra perlu dilakukan untuk meningkatkan kualitas penampakan atau menonjolkan beberapa aspek informasi yang terkandung di dalam citra,
- b. Elemen di dalam citra perlu dikelompokkan, dicocokkan, atau diukur,
- c. Sebagian citra perlu digabung dengan citra yang lain.

#### 2.3.1 Operasi Piksel

Pada pengolahan citra terdapat istilah operasi piksel atau kadang disebut operasi piksel-ke-piksel. Operasi piksel adalah operasi pengolahan citra yang memetakan hubungan setiap piksel yang bergantung pada piksel itu sendiri. Jika  $f(y, x)$  menyatakan nilai sebuah piksel pada citra  $f$  dan  $g(y, x)$  menyatakan piksel hasil pengolahan dari  $f(y, x)$ , hubungannya dapat dinyatakan dengan

$$g(y, x) = T(f(y, x)) \quad \dots (2.4)$$

Dalam hal ini, T menyatakan fungsi atau macam operasi yang dikenakan terhadap piksel  $f(y, x)$ .

### 2.3.2 Metode Ubah Nilai RGB ke *Grayscale*

Untuk memudahkan pengolahan citra, biasanya citra RGB akan dikonversi menjadi citra skala keabuan (*grayscale*). Karena pada citra RGB pengolahan citra akan dilakukan dengan menghitung pada 3 *channel* warna, sedangkan pada citra skala keabuan cukup 1 *channel* warna sehingga memudahkan proses komputasi. Format citra skala keabuan memakai warna hitam sebagai warna minimal (0) dan warna putih (255) sebagai warna maksimalnya, sehingga warna antaranya adalah abu-abu seperti ditunjukkan pada gambar.

Berdasarkan catatan dokumentasi GIMP, dapat kita telusuri bahwa terdapat tiga macam metode algoritma untuk mengubah nilai R G B menjadi *grayscale* antara lain:

#### 1. *Lightnes*

Algoritmanya adalah mencari nilai tertinggi dan terendah dari nilai R G B, kemudian nilai tertinggi dan terendah tersebut dijumlahkan lantas dikalikan dengan 0.5 (dibagi 2). Secara matematis dapat dirumuskan:

$$Grayscale = (\max(R,G,B) + \min(R,G,B)) * 0.5 \quad \dots (2.5)$$

#### 2. *Average*

Algoritmanya adalah dengan menjumlahkan seluruh nilai R G B, kemudian dibagi 3, sehingga diperoleh nilai rata-rata dari R G B, nilai rata-rata itulah yang dapat dikatakan sebagai *grayscale*. Rumus matematisnya adalah:

$$Grayscale = (R + G + B) / 3 \quad \dots (2.6)$$

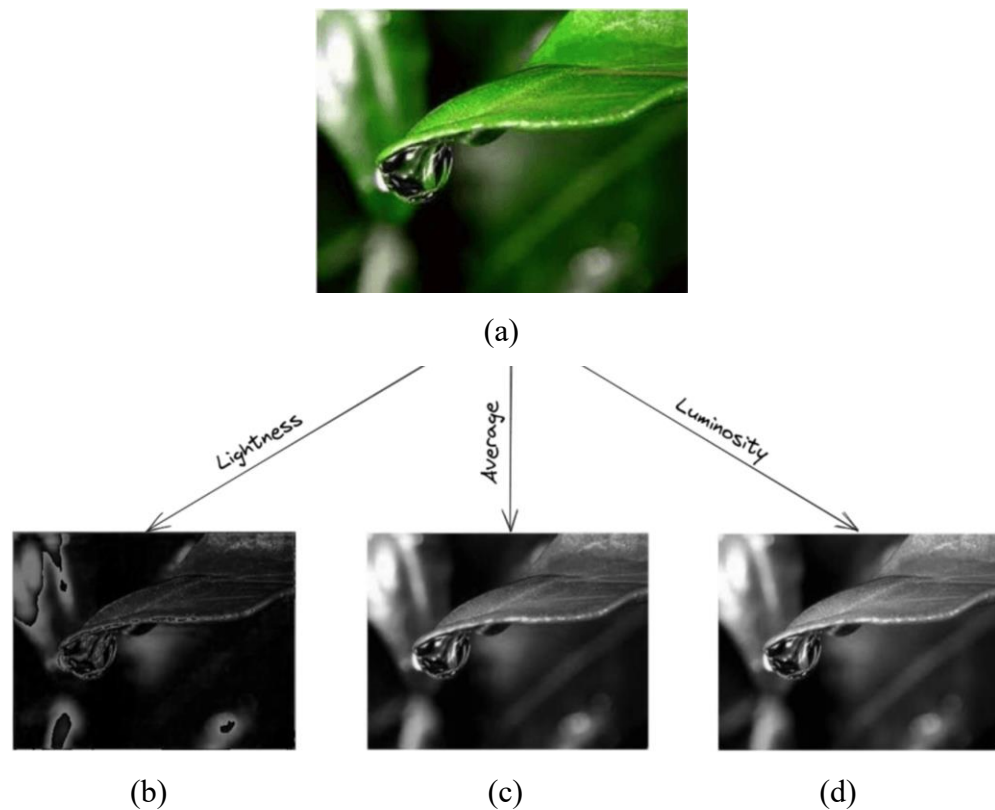
#### 3. *Luminosity*

Algoritmanya adalah dengan mengalikan setiap nilai R G B dengan konstanta tertentu yang sudah ditetapkan nilainya, kemudian hasil perkalian seluruh nilai R G B dijumlahkan satu sama lain. Rumus matematisnya adalah:

$$Grayscale = (0.3 * R) + (0.59 * G) + (0.11 * B) \quad \dots (2.7)$$

Dari ketiga macam algoritma diatas, masing-masing metode walaupun sama-sama menghasilkan warna *grayscale*, namun tingkat derajat *grayscale*-nya berbeda-beda, sehingga setiap metode menghasilkan warna abu-abu yang tidak sama persis

satu sama lain. Berikut gambaran perbedaan hasil gambar dengan ketiga metode *grayscale*:



Gambar 2. 10. Hasil Metode Grayscale

Gambar 2.10a adalah gambar asli yang berformat RGB. Gambar 2.10(b), 2.10(c), dan 2.10(d) masing-masing adalah hasil konversi *grayscale* dengan metode *Lightness* (2.10 (b)), *Average* (2.10(c)), dan *Luminosity* (2.10(d)).

Seperti yang diharapkan, metode *Lightnes* adalah yang terburuk, karena metode ini sama sekali gagal menangkap kecerahan gambar input. Sedangkan metode *Average* sering kali menghasilkan citra grayscale yang terlihat lebih gelap. Di antara dua metode lainnya, *Luminosity* adalah yang terbaik, karena lebih akurat dalam menangkap kecerahan gambar .

### 2.3.3 Histogram Citra

Histogram citra adalah grafik yang menggambarkan penyebaran nilai-nilai intensitas *pixel* dari suatu citra atau bagian tertentu di dalam citra. Dari sebuah histogram dapat diketahui frekuensi kemunculan nisbi (*relative*) dari intensitas pada citra tersebut. Histogram juga dapat menunjukkan banyak hal tentang kecerahan (*brightness*) dan kontras (*contrast*) dari sebuah gambar.

Umumnya setiap *pixel* memiliki nilai derajat keabuan. Misalkan citra dengan L derajat keabuan, yaitu dari nilai 0 sampai L 1. Secara matematis histogram citra dihitung dengan rumus

$$H(i) = n_i/n \quad \dots (2.8)$$

yang dalam hal ini:

$H(i)$  = frekuensi piksel dengan derajat keabuan  $i$

$n_i$  = jumlah *pixel* yang memiliki derajat keabuan  $i$

$n$  = jumlah seluruh *pixel* di dalam citra

Sebagai contoh penghitungan histogram citra, misalkan matriks citra digital berukuran 8 x 8 *pixel* dengan derajat keabuan dari 0 sampai 15 (ada 16 buah derajat keabuan):

3	7	7	8	10	12	14	10
2	0	0	0	1	8	15	15
14	6	5	9	8	10	9	12
12	12	11	8	8	10	11	1
0	2	3	4	5	13	10	14
4	5	0	0	1	0	2	2
15	13	11	10	9	9	8	7
2	1	0	10	11	14	13	12

Gambar 2. 10. Matriks Citra Digital

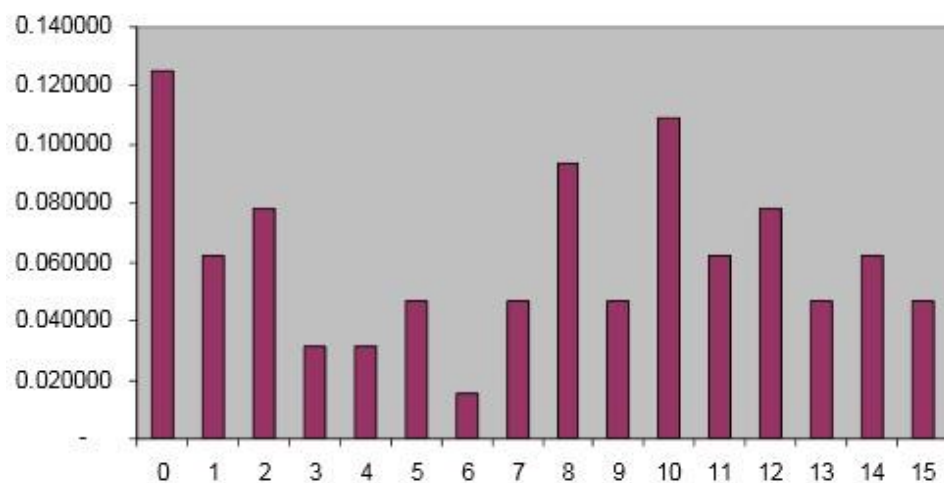
Tabulasi perhitungan histogramnya dapat dilihat pada tabel 2.4. Dapat dilihat bahwa semakin besar nilai  $n_i$  maka semakin besar pula nilai  $h_i$ .

Tabel 2. 4. Perhitungan Histogram

$i$	$n_i$	$h_i = n_i/n \quad (n=64)$
0	8	0.125
1	4	0.0625
2	5	0.078125
3	2	0.03125
4	2	0.03125
5	3	0.046875

6	1	0.015625
7	3	0.0468875
8	6	0.09375
9	3	0.046875
10	7	0.109375
11	4	0.0625
12	5	0.078125
13	3	0.046875
14	4	0.0625
15	3	0.046875

Histogram citra digital dari matriks citra digital diatas dapat dilihat pada Gambar 2.12 berikut ini:



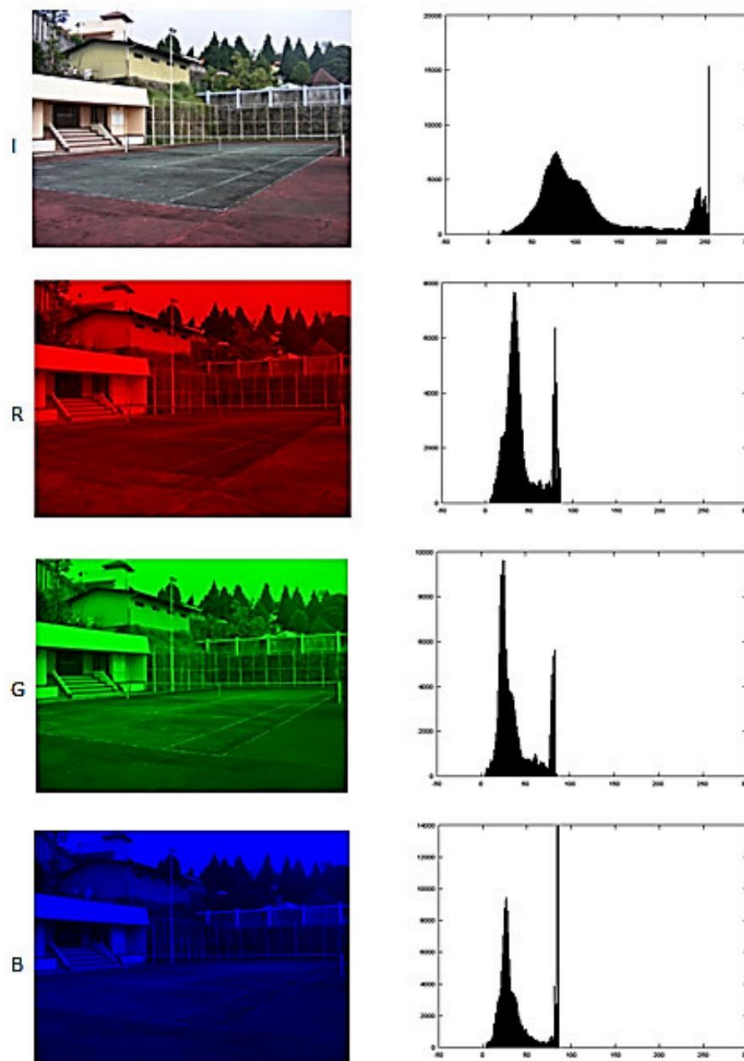
Gambar 2. 11. Histogram Citra Digital

Pada pengolahan citra, histogram mempunyai peran yang cukup penting. Manfaat yang dapat didapatkan seperti berikut.

1. Berguna untuk mengamati penyebaran intensitas warna dandapat dipakai untuk pengambilan keputusan misalnya dalam peningkatan kecerahan atau pereganggan kontras serta sebaran warna.
2. Berguna untuk penentuan batas-batas dalam pemisahan objek dari latarbelakangnya.

3. Memberikan persentase komposisi warna dan tekstur intensitas untuk kepentingan identifikasi citra.

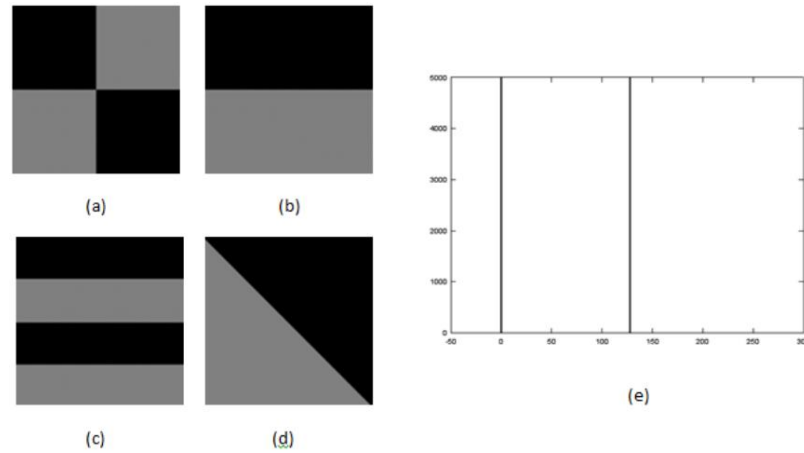
Khusus pada citra berwarna, histogram dapat diterapkan pada gabungan komponen-komponen RGB penyusunnya ataupun per komponen. Gambar 2.13 menunjukkan contoh mengenai hal itu. Pada gambar tersebut, I menyatakan histogram gabungan intensitas warna, R untuk komponen warna merah, G untuk komponen warna hijau, dan B untuk komponen warna biru.



Gambar 2. 12 Histogram pada citra berwarna secara menyeluruh (I), merah (R), hijau (G), dan biru (B)

Histogram tidak mencerminkan susunan posisi warna piksel di dalam citra. Oleh karena itu, histogram tidak dapat dipakai untuk menebak bentuk objek yang terkandung di dalam citra. Sebagai contoh, Gambar 2.14 memperlihatkan empat

buah citra yang memiliki histogram yang sama, tetapi bentuk masingmasing jauh berbeda. Dengan demikian, histogram tidak memberikan petunjuk apapun tentang bentuk yang terkandung dalam keempat citra tersebut.



Gambar 2. 13 Empat buah citra (a), (b),(c), dan (d) yang memiliki histogram yang sama (e), tetapi mempunyai informasi yang jauh berbeda

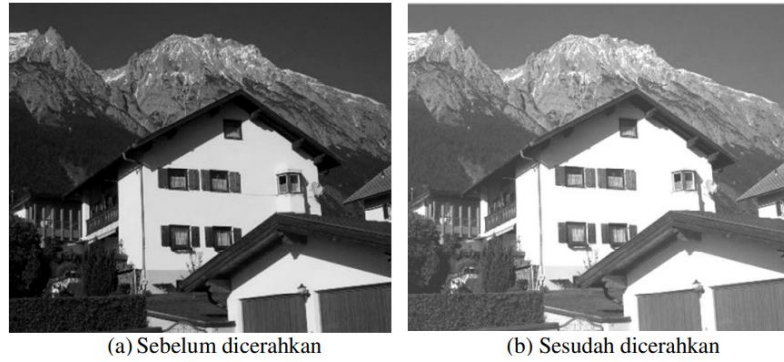
#### 2.3.4 Meningkatkan Kecerahan

Operasi dasar yang sering dilakukan pada citra adalah peningkatan kecerahan (brightness). Operasi ini diperlukan dengan tujuan untuk membuat gambar menjadi lebih terang. Secara matematis, peningkatan kecerahan dilakukan dengan cara menambahkan suatu konstanta terhadap nilai seluruh piksel. Misalkan,  $f(y, x)$  menyatakan nilai piksel pada citra berskala keabuan pada koordinat  $(y, x)$ . Maka, citra baru

$$g(y, x) = f(y, x) + \beta \quad \dots \dots (2.9)$$

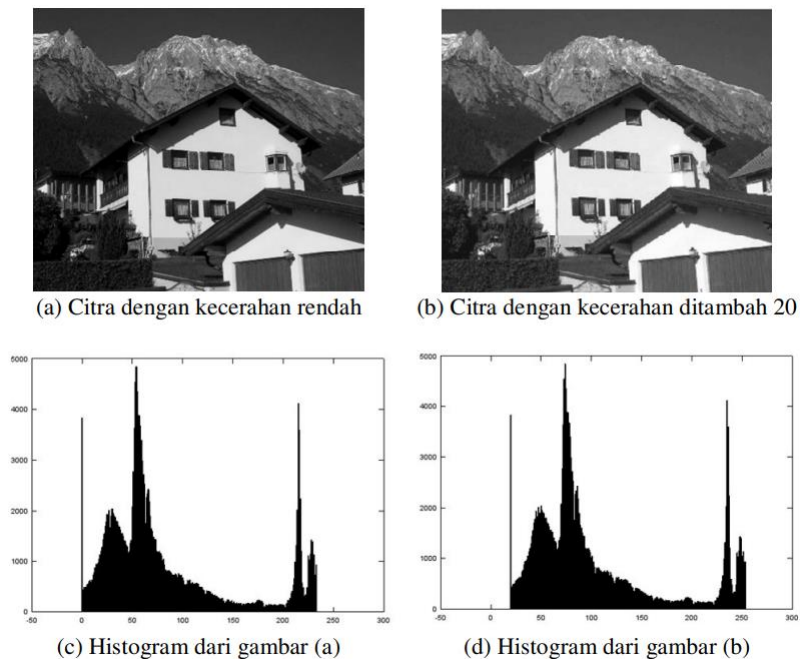
telah meningkat nilai kecerahan semua pikselnya sebesar terhadap citra asli  $f(y, x)$ . Apabila  $\beta$  berupa bilangan negatif, kecerahan akan menurun atau menjadi lebih gelap.

Sebagai contoh, terdapat citra seperti pada Gambar 2. 15



Gambar 2. 14 Efek pencerahan gambar

Jika dilihat melalui histogram, peningkatan kecerahan sebenarnya berefek pada penggeseran komposisi intensitas piksel ke kanan bila  $\beta$  berupa bilangan positif atau ke kiri jika  $\beta$  berupa bilangan negatif di Persamaan 2.10. Gambar 2.16 memperlihatkan keadaan ketika pencerahan dilakukan.



Gambar 2. 15 Histogram pada peningkatan citra. Komposisi jumlah intensitas

Perhatikan, warna hitam (ditandai dengan garis tunggal yang menonjol di ujung kiri histogram) ikut tergeser. Jadi, warna hitam tidak lagi menjadi hitam kalau peningkatan kecerahan dilakukan dengan cara seperti di depan.

Selain itu, menambah kecerahan sebuah citra dapat juga menggunakan persamaan (2)

$$g(y, x) = \alpha f(y, x) \quad \dots \dots (2.10)$$



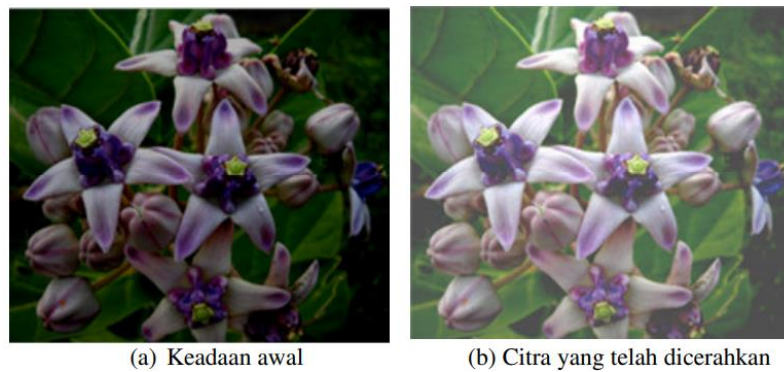
Persamaan (2) menunjukkan bahwa setiap pixel pada citra awal dikalikan dengan sebuah konstanta  $\alpha$ , untuk menghasilkan sebuah citra yang baru. Persamaan (2.10) ini juga sering disebut sebagai fungsi yang dapat merenggangkan kontras sebuah image/citra.

Selanjutnya persamaan (1) dan (2) dapat dikombinasikan menjadi sebuah persamaan (2.11)

$$g(y, x) = \alpha f(y, x) + \beta \quad \dots \dots (2.11)$$

dapat dijelaskan bahwa persamaan (3) sebuah citra baru dihasilkan dari hasil perkalian sebuah setiap pixel dengan nilai  $\alpha$  lalu ditambahkan sebuah nilai  $\beta$  pada citra awal. Persamaan (2.11) ini juga dapat menyebabkan penambahan kecerahan sebuah citra.

Bagaimana kalau ingin mencerahkan pada citra berwarna? Secara prinsip, hal itu sama saja dengan pada citra berskala keabuan. Tentu saja, dalam hal ini, penambahan konstanta dilakukan pada ketiga komponen penyusun warna. Gambar 2.17 memperlihatkan perbedaan antara gambar pada keadaan awal dan setelah dicerahkan. Gambar 2.17(a) menyatakan citra pada RGB dan Gambar 2.17(b) menyatakan citra pada RGB2.



Gambar 2. 16 Peningkatan kecerahan pada citra berwarna

## **BAB III**

### **PENERAPAN**

#### **3.1 Persiapan Bahan**

Sebelum melakukan penerapan gambar berdasarkan operasi-operasi citra digital, penulis terlebih dahulu mempersiapkan bahan-bahan sebagai berikut:

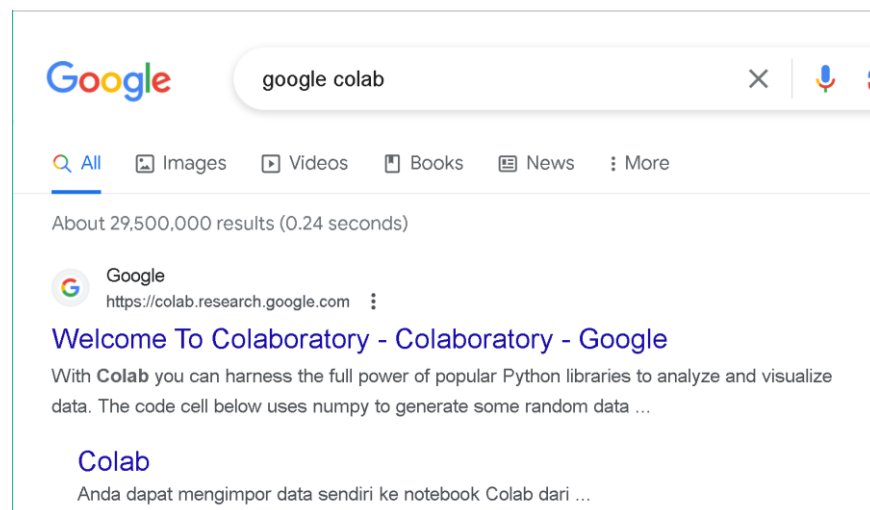
1. Mempersiapkan software Google Colab yang sudah terintegrasi dengan bahasa pemrograman python, dan dalam bentuk file Jupyter (.ipynb) serta telah terinstal Open CV di dalamnya
2. Menyediakan gambar-gambar sesuai yang dibutuhkan.

#### **3.2 Praktek**

Berikut langkah-langkah pengimplementasian operasi-operasi dalam citra digital:

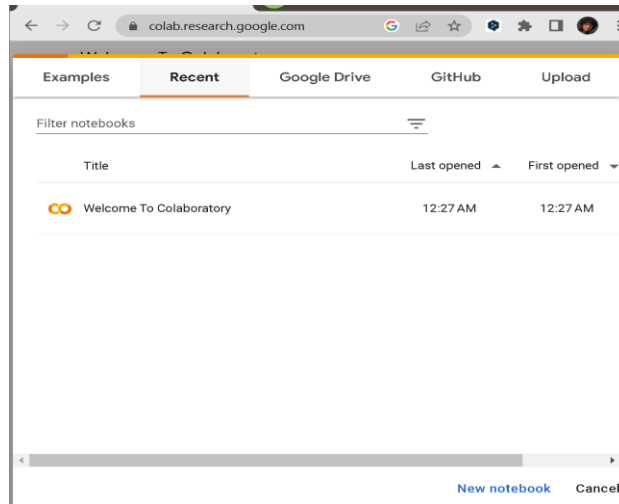
##### **3.1.1 Langkah Setup Google Colab**

1. Masuk dalam halaman google untuk masuk ke Google Colab, dengan mengetik <https://colab.research.google.com>



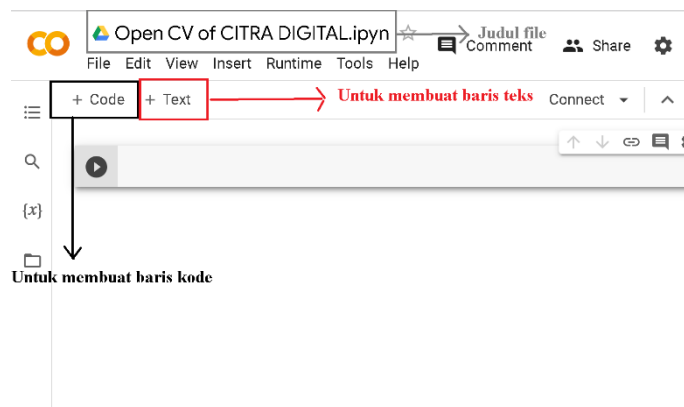
Gambar 3. 1Tampilan Halaman Google Colab

2. Setelah masuk dalam tampilan google colab, di sini kemudian penulis membuat New notebook, untuk menyimpan source code dalam melakukan operasi citra digital.



Gambar 3. 2 Tampilan awal Google Colab

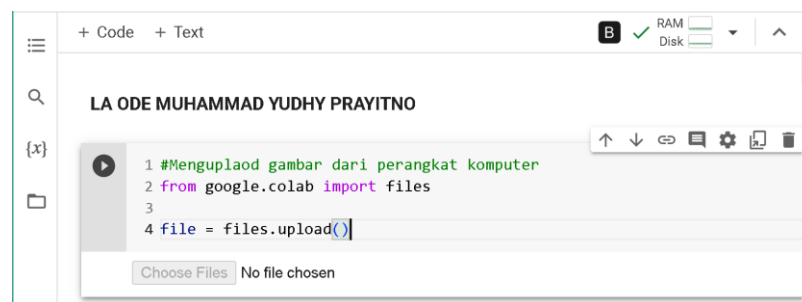
3. Kemudian, setelah membuat file baru, maka kita sudah bisa menjalankan operasi-operasi citra digital sesuai dengan pembahasan pada bab 2.



Gambar 3. 3 Tampilan setelah membuat file

### 3.1.2 Langkah Operasi Citra Digital

Pertama-tama dalam melakukan operasi-operasi citra digital, diperlukan image untuk diproses sesuai dengan operasi citra digital yang diinginkan, berikut kode untuk mengupload gambar dari device masuk ke dalam file Jupyter kita.



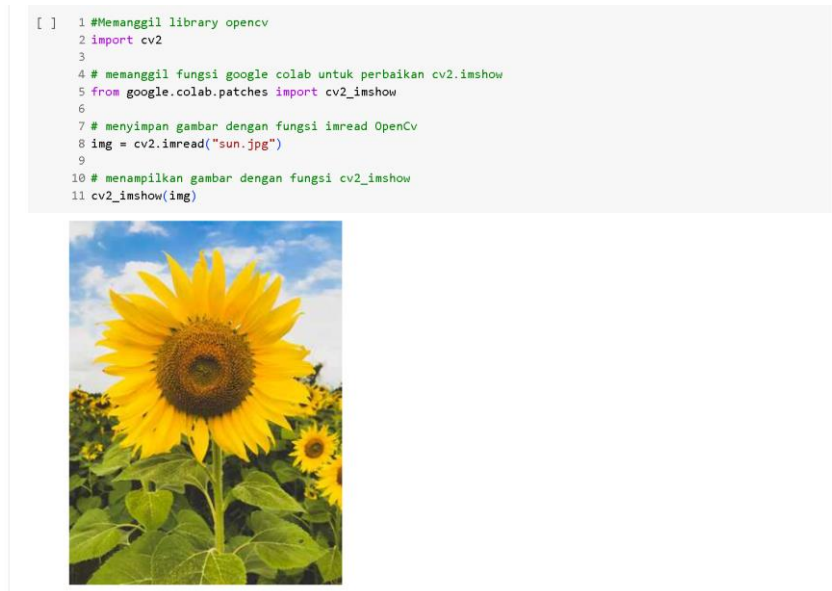
Gambar 3. 4 Source code untuk mengupload gambar

Kemudian untuk menjalankannya menekan ikon play dibagian pojok sebelah kiri, lalu menambahkan gambar dari device kita. Lakukan langkah ini sesuai banyak gambar yang ingin di upload. Jika ingin melihat gambar sudah terupload atau belum jalankan perintah berikut.



Gambar 3. 5 Perintah melihat file yang sudah terupload

Disini terlihat bahwa ada 2 gambar yang baru saya upload berdasarkan kode di gambar 3.4, yaitu file gambar coba.jpeg dan sun.jpg. Kemudian disini kita bisa untuk menampilkan gambar dari hasil file gambar yang telah diupload. Untuk menampilkannya jalankan perintah berikut.



Gambar 3. 6 Perintah menampilkan gambar

Berdasarkan kode di atas, langkah-langkah yang dilakukan adalah memanggil library OpenCV untuk pemrosesan citra, menggunakan fungsi cv2\_imshow dari Google Colab untuk menampilkan gambar, membaca gambar menggunakan cv2.imread, dan menampilkan gambar tersebut menggunakan cv2\_imshow. Setelah itu jika kita ingin mengetahui dimensi dari gambar di atas, maka perlu untuk menjalankan kode berikut.

```
1 print(img.shape)
(400, 300, 3)
```

Gambar 3. 7 Tampilan dimensi gambar

Dari gambar di atas gambar sun.jpg berisi tiga elemen, yaitu tinggi (height), lebar (width), dan jumlah saluran warna (channels) dari gambar, yang menunjukkan tinggi 400 piksel, lebar 300 piksel, dan 3 saluran warna (merah, hijau, biru). Kemudian, selain cara menampilkan gambar menggunakan fungsi `cv2\_imshow` pada gambar 3.6, ada cara lain yang dapat digunakan menggunakan library `matplotlib`.



Gambar 3. 8 Menampilkan gambar menggunakan library `matplotlib`.

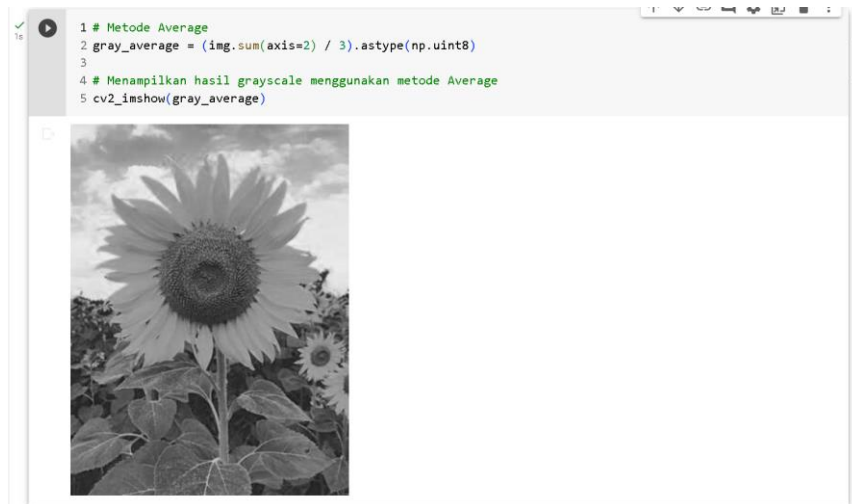
Pada kode di atas, terlebih dahulu gambar "sun.jpg" dibaca menggunakan fungsi `cv.imread` dari OpenCV, kemudian diubah saluran warnanya dari BGR ke RGB menggunakan fungsi `cv.cvtColor`. Setelah itu, gambar yang telah diubah saluran warnanya ditampilkan menggunakan fungsi `plt.imshow` dari matplotlib. Terakhir, dengan memanggil `plt.show()`, gambar ditampilkan dalam jendela tampilan. Terlihat bahwa ada sedikit perbedaan dengan menampilkan gambar menggunakan `cv2.imshow` yaitu adanya seperti sumbu x dan y, sumbu x menandakan lebar piksel gambar yaitu sesuai pada gambar 3.7 yaitu 300, dan sumbu y menampilkan tinggi piksel gambar yaitu 400.

Kemudian lanjut pada operasi pertama sesuai pembahasan pada bab 2 yaitu bagian 3.2 Metode Ubah Nilai RGB ke Grayscale. Dari pembahasan sebelumnya terdapat 3 persamaan umum dalam mengubah gambar RGB ke bentuk grayscale. Berikut kode untuk mengubah RGB ke grayscale.



Gambar 3. 9 RGB to Grayscale metode Lightnes

Pada kode di atas, penulis menggunakan library numpy dengan memanggil import numpy as np. Kemudian dilakukan perhitungan untuk mengubah citra warna menjadi citra keabuan menggunakan metode Lightness. Berdasarkan pembahasan sebelumnya, metode ini mengambil nilai maksimum (max) dan nilai minimum (min) dari setiap piksel dalam citra pada setiap saluran warna (merah, hijau, biru), kemudian menjumlahkan kedua nilai tersebut dan membaginya dengan 2. Hasil perhitungan tersebut kemudian dikonversi menjadi tipe data unsigned integer 8-bit (np.uint8). Dalam tipe data ini, rentang nilai yang dapat direpresentasikan adalah dari 0 hingga 255. Setiap piksel dalam citra keabuan yang dihasilkan menggunakan metode Lightness direpresentasikan dalam format ini, di mana setiap nilai piksel berkisar antara 0 (kecerahan minimum) hingga 255 (kecerahan maksimum). Selanjutnya, citra keabuan yang dihasilkan ditampilkan menggunakan fungsi cv2\_imshow. Selanjutnya untuk metode kedua sebagai berikut.



Gambar 3. 10 RGB to Grayscale metode Average

Pada gambar di atas, penulis menerapkan metode Average untuk mengubah citra menjadi grayscale. Sesuai pembahasan sebelumnya, metode Average mengambil rata-rata nilai dari setiap piksel dalam citra dengan menjumlahkan nilai-nilai R, G, dan B pada setiap piksel dan membaginya dengan 3. Selanjutnya, hasil citra grayscale yang dihasilkan dengan metode Average ditampilkan menggunakan `cv2_imshow(gray_average)`. Dengan demikian, kita dapat melihat citra grayscale yang dihasilkan menggunakan metode Average. Selanjutnya untuk metode ketiga sebagai berikut.

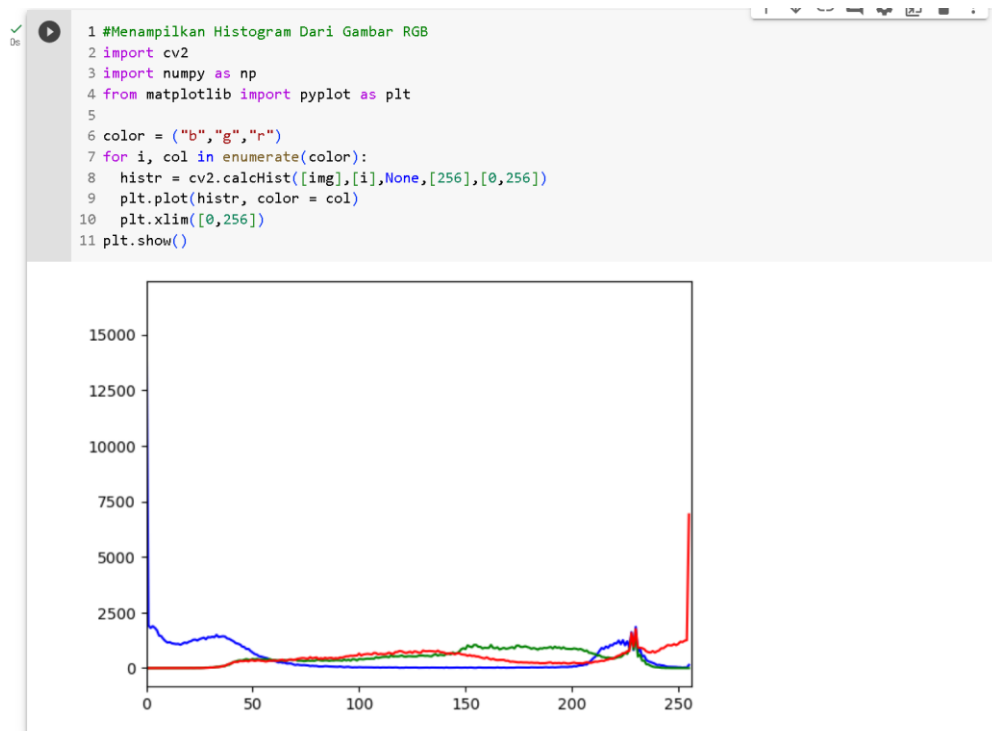


Gambar 3. 11 RGB to Grayscale metode Luminosity

Pada gambar kode di atas, penulis menggunakan metode Luminosity untuk mengubah citra menjadi grayscale. Berdasarkan pembahasan sebelumnya, metode Luminosity menghitung nilai grayscale dari setiap piksel dengan menggunakan bobot tertentu untuk masing-masing komponen warna (R, G, B). Pada metode ini,

kita mengalikan nilai piksel pada kanal R dengan bobot 0.3, nilai piksel pada kanal G dengan bobot 0.59, dan nilai piksel pada kanal B dengan bobot 0.11. Kemudian, kita menjumlahkan ketiga hasil perkalian tersebut untuk mendapatkan nilai grayscale yang akhir. Terakhir, hasil citra grayscale menggunakan metode Luminosity ditampilkan dengan menggunakan `cv2_imshow(gray_luminosity)`. Dengan demikian, kita dapat melihat citra grayscale yang dihasilkan menggunakan metode Luminosity.

Kemudian lanjut pada operasi kedua sesuai pembahasan pada bab 2 yaitu bagian 3.3 Histogram Citra. Dari pembahasan sebelumnya kita dapat menampilkan histogram dari gambar RGB maupun dari citra grayscale. Berikut kode- kode untuk menampilkan gambar histogram dari citra RGB dan grayscale.

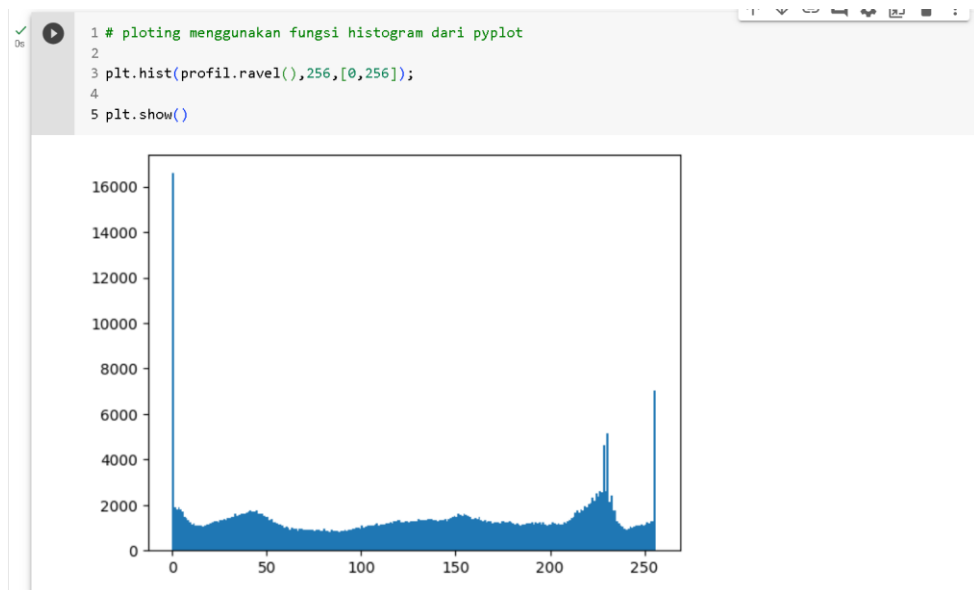


Gambar 3. 12 Menampilkan Histogram pada gambar RGB

Pada tampilan di atas, penulis menggunakan fungsi `cv2.calcHist()` untuk menghitung histogram dari gambar RGB. Pertama, penulis melakukan iterasi untuk setiap komponen warna (biru, hijau, merah) dengan menggunakan `enumerate(color)`. Di setiap iterasi, kita menghitung histogram untuk komponen warna tersebut dengan memanggil `cv2.calcHist([img], [i], None, [256], [0,256])`, di mana `img` adalah gambar input dan `i` adalah indeks komponen warna yang sedang diproses. Fungsi `cv2.calcHist()` akan menghasilkan histogram sebagai array numpy.



Selanjutnya, kita menggunakan `plt.plot()` untuk menggambar histogram dengan menggunakan warna yang sesuai dengan komponen warna yang sedang diproses. Kita juga mengatur batas sumbu x menggunakan `plt.xlim([0,256])` agar histogram ditampilkan dalam rentang nilai piksel yang valid. Terakhir, kita memanggil `plt.show()` untuk menampilkan histogram yang dihasilkan. Dengan demikian, kita dapat melihat distribusi intensitas piksel pada setiap komponen warna dalam gambar RGB. Gambar RGB yang dimaksud disini adalah gambar `sun.jpg` tadi yang gambarnya bisa dilihat di bagian 3.6

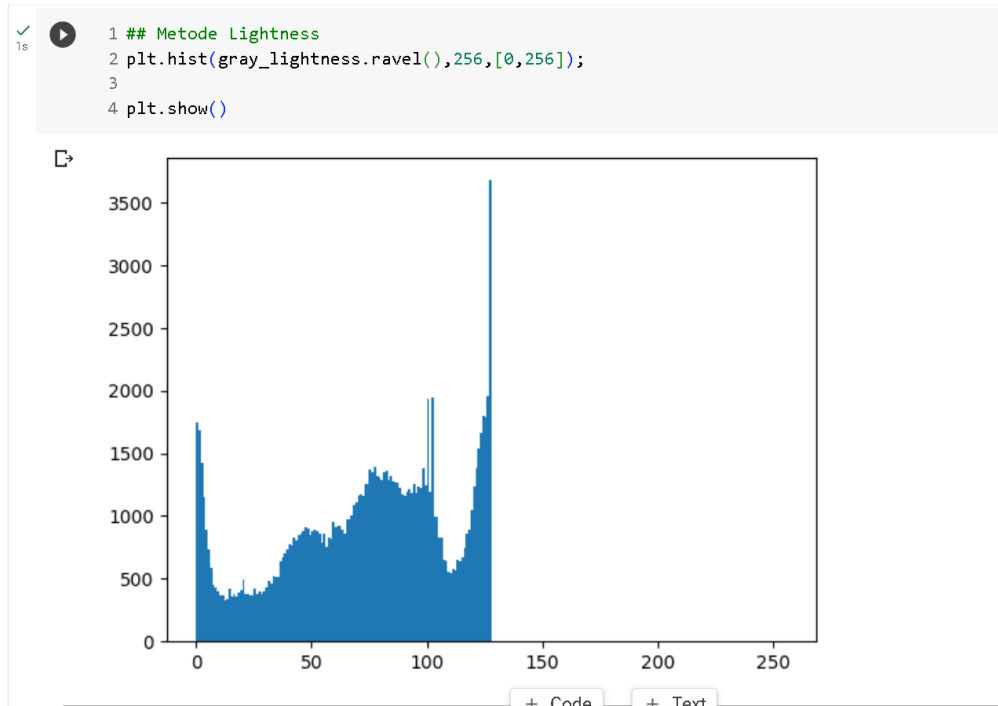


Gambar 3. 13 Menampilkan histogram dari fungsi pyplot

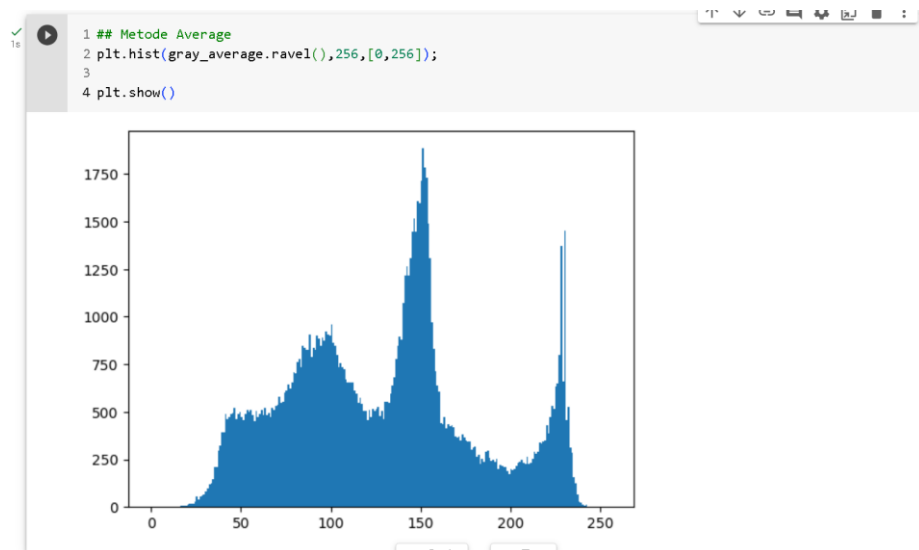
Kemudian Pada tampilan gambar di atas, penulis menggunakan fungsi `'plt.hist()'` dari pyplot untuk menghasilkan histogram dari gambar RGB sebelumnya. Kita memanggil `'plt.hist(profil.ravel(),256,[0,256])'` untuk menghitung dan menampilkan histogram dari gambar dengan menggunakan 256 bin pada rentang nilai piksel 0 hingga 255. Fungsi `'ravel()'` digunakan untuk meratakan array gambar menjadi satu dimensi sebelum menghitung histogramnya. Setelah itu, kita memanggil `'plt.show()'` untuk menampilkan histogram yang dihasilkan.

Perbedaan utama antara pendekatan ini dengan pendekatan sebelumnya adalah cara perhitungan histogram dan visualisasi yang digunakan. Pada pendekatan sebelumnya, kita menggunakan `'cv2.calcHist()'` untuk menghitung histogram menggunakan OpenCV, sedangkan pada pendekatan ini kita

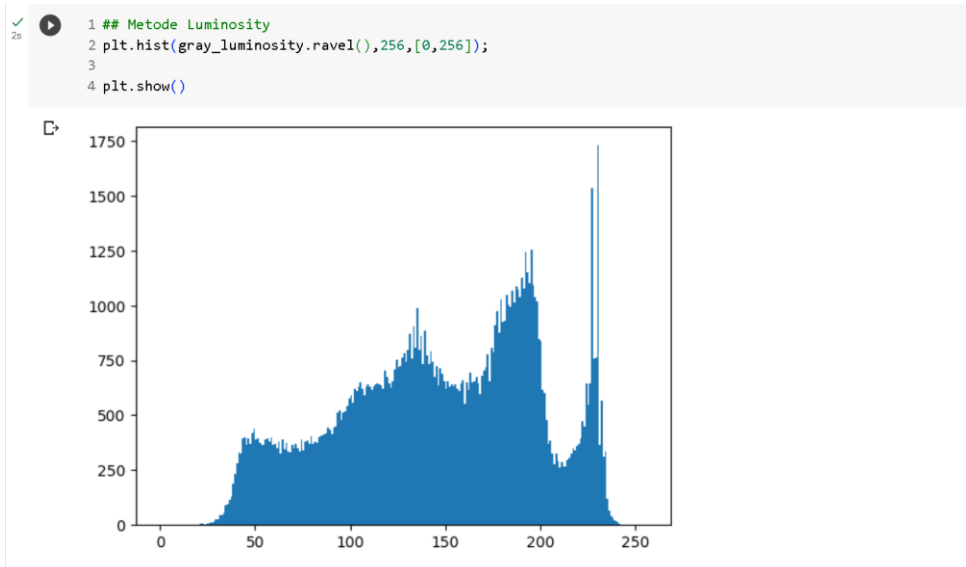
menggunakan `plt.hist()` dari pyplot untuk menghitung dan menampilkan histogram menggunakan Matplotlib. Selain itu, kita juga menggunakan fungsi `ravel()` untuk meratakan array gambar menjadi satu dimensi sebelum menghitung histogramnya. Kemudian lanjut menampilkan histogram dari gambar grayscale sebelumnya.



Gambar 3. 14 Menampilkan histogram dari gambar grayscale metode Lightnes



Gambar 3. 15 Menampilkan histogram dari gambar grayscale metode Average



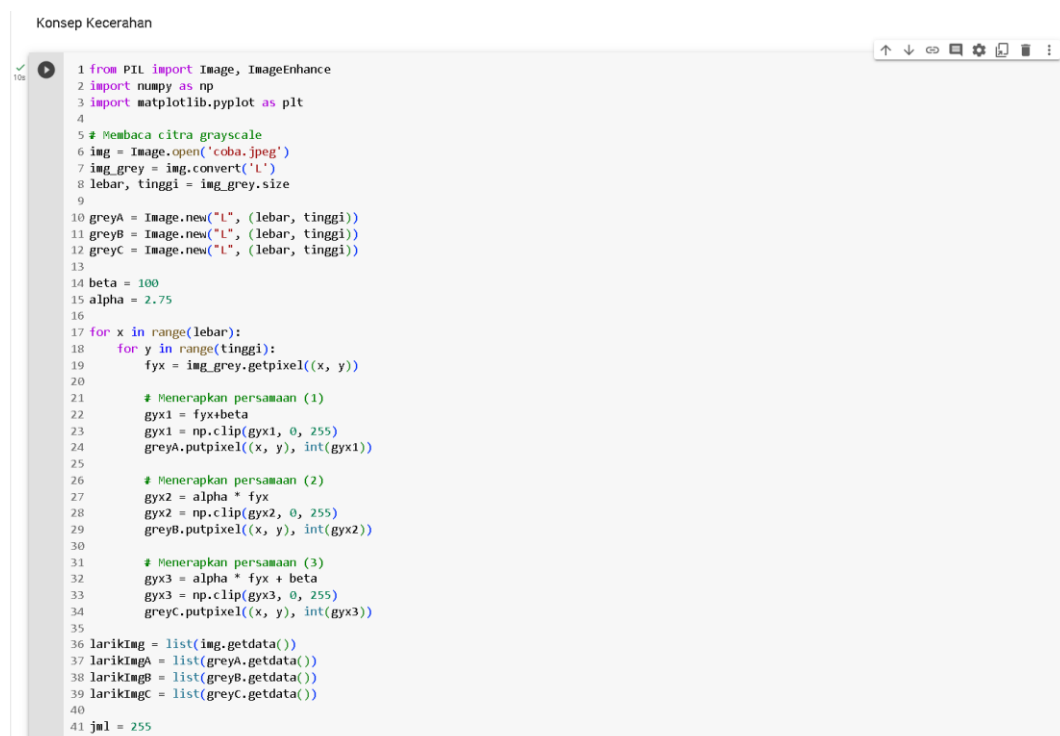
Gambar 3. 16 Menampilkan histogram dari gambar grayscale metode Luminosity

Sesuai gambar di atas penulis menggunakan `plt.hist(gray_lightness.ravel(),256,[0,256])` untuk menghitung dan menampilkan histogram dari citra keabuan yang dihasilkan menggunakan metode Lightness, Average, dan Luminosity. Fungsi `ravel()` digunakan untuk meratakan array citra keabuan menjadi satu dimensi sebelum menghitung histogramnya. Rentang nilai piksel yang digunakan adalah 0 hingga 255 dengan 256 bin. "256 bin" mengacu pada citra dipecah menjadi 256 interval atau kelompok nilai piksel. Setiap interval mewakili rentang nilai piksel tertentu, seperti 0 hingga 1, 2 hingga 3, dan seterusnya. Jumlah bin yang lebih besar memberikan representasi yang lebih detail tentang intensitas piksel dalam citra. Dengan menggunakan 256 bin, kita dapat melihat perubahan intensitas piksel dengan lebih jelas dan mendapatkan informasi yang lebih rinci tentang distribusi intensitas dalam citra tersebut. Setelah itu, kita memanggil `plt.show()` untuk menampilkan histogram yang dihasilkan.

Berdasarkan gambar 2.14 bisa dilihat bahwa, tampilan histogram untuk metode Lightness untuk nilai sumbu x nya berada disekitar rentang antara 0-125. Artinya, gambar grayscale pada metode Lightness mengandung banyak unsur hitam pekat dan ke abu-abuan. Dengan presentase nilai x yang berada disekitar nilai 125 memiliki nilai sumbu y tertinggi hingga sekitar 3.500 ke atas. Sedangkan untuk gambar 2.15 bisa dilihat bahwa, tampilan histogram untuk metode Average untuk nilai sumbu x nya berada disekitar rentang 25-240. Dengan nilai pada sumbu y

memiliki nilai tertinggi yaitu sekitar 1750 ke atas, pada rentang nilai sumbu x pada angka sekitar 150. Artinya pada gambar grayscale dengan metode Average mengandung banyak unsur warna keabuan, dengan mengandung nilai yang sedikit untuk warna hitam, serta warna putih mengandung sekitar setengah dari warna keabuan. Kemudian lanjut pada gambar 3.16 untuk metode Luminosity, bisa dilihat bahwa tampilan histogramnya hampir sama dengan pada gambar 3.15 pada metode Average, yang membedakan hanya pada sumbu y nya, dimana pada metode Average nilai keabuan pada sumbu x sekitar di angka 150an, memiliki sumbu y dengan nilai tertinggi hingga di atas 1750an, di metode Luminosity pada sumbu x nya di angka 150an, memiliki sumbu y tertinggi hingga di atas 1750an. Artinya pada metode Luminosity paling banyak mengandung warna keputihan, disusul dengan warna keabu-abuan dan nilai yang sedikit pada warna hitam.

Kemudian lanjut pada operasi ketiga sesuai pembahasan pada bab 2 yaitu bagian 3.4 Meningkatkan Kecerahan. Dari pembahasan sebelumnya kita dapat menampilkan gambar hasil pencerahan dari gambar grayscale. Berikut kode untuk mengimplementasikannya.



```

Konsep Kecerahan
1 from PIL import Image, ImageEnhance
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 # Membaca citra grayscale
6 img = Image.open('coba.jpeg')
7 img_grey = img.convert('L')
8 lebar, tinggi = img_grey.size
9
10 greyA = Image.new("L", (lebar, tinggi))
11 greyB = Image.new("L", (lebar, tinggi))
12 greyC = Image.new("L", (lebar, tinggi))
13
14 beta = 100
15 alpha = 2.75
16
17 for x in range(lebar):
18     for y in range(tinggi):
19         fyx = img_grey.getpixel((x, y))
20
21         # Menerapkan persamaan (1)
22         gyx1 = fyx + beta
23         gyx1 = np.clip(gyx1, 0, 255)
24         greyA.putpixel((x, y), int(gyx1))
25
26         # Menerapkan persamaan (2)
27         gyx2 = alpha * fyx
28         gyx2 = np.clip(gyx2, 0, 255)
29         greyB.putpixel((x, y), int(gyx2))
30
31         # Menerapkan persamaan (3)
32         gyx3 = alpha * fyx + beta
33         gyx3 = np.clip(gyx3, 0, 255)
34         greyC.putpixel((x, y), int(gyx3))
35
36 larikImg = list(img.getdata())
37 larikImgA = list(greyA.getdata())
38 larikImgB = list(greyB.getdata())
39 larikImgC = list(greyC.getdata())
40
41 jul = 255

```

Gambar 3. 17 Kode untuk meningkatkan kecerahan dari gambar grayscale

Kode di atas menggambarkan proses pengolahan untuk menerapkan konsep kecerahan dari gambar dengan citra grayscale dengan menerapkan tiga persamaan

yang berbeda pada setiap piksel citra. Citra grayscale awal dibaca menggunakan library PIL (Python Imaging Library) dengan ImageEnhance adalah modulnya untuk meningkatkan atau mengubah kualitas citra. Kemudian gambar awal dikonversi menjadi citra grayscale dengan fungsi `'convert('L')'`, digunakan untuk memastikan bahwa citra dalam format grayscale, meskipun citra awalnya mungkin sudah dalam format grayscale.. Setelah itu, dilakukan inisialisasi tiga citra baru yaitu `'greyA'`, `'greyB'`, dan `'greyC'` dengan ukuran yang sama seperti citra awal.

Selanjutnya, dilakukan iterasi pada setiap piksel citra grayscale menggunakan nested loop. Pada setiap iterasi, nilai piksel asli (`'fyx'`) diambil menggunakan fungsi `'getpixel()'` dari citra awal. Kemudian, tiga persamaan yang berbeda diterapkan pada nilai piksel tersebut:

1. Persamaan (1):  $\text{'gyx1'} = \text{'fyx'} + \text{'beta'}$ , dengan `'beta'` sebagai nilai konstanta. Hasil dari persamaan ini adalah `'gyx1'` yang kemudian dibatasi pada rentang 0 hingga 255 menggunakan fungsi `'np.clip()'`. Nilai yang dihasilkan kemudian disimpan pada citra `'greyA'` dengan menggunakan fungsi `'putpixel()'`.
2. Persamaan (2):  $\text{'gyx2'} = \alpha * \text{'fyx'}$ , dengan `'alpha'` sebagai nilai konstanta. Hasil dari persamaan ini adalah `'gyx2'` yang juga dibatasi pada rentang 0 hingga 255. Nilai yang dihasilkan kemudian disimpan pada citra `'greyB'`.
3. Persamaan (3):  $\text{'gyx3'} = \alpha * \text{'fyx'} + \text{'beta'}$ , dengan `'alpha'` dan `'beta'` sebagai nilai konstanta. Hasil dari persamaan ini adalah `'gyx3'` yang juga dibatasi pada rentang 0 hingga 255. Nilai yang dihasilkan kemudian disimpan pada citra `'greyC'`.

Selanjutnya, dilakukan konversi nilai piksel pada setiap citra menjadi larik menggunakan fungsi `'getdata()'` dari library PIL. Larik tersebut kemudian digunakan untuk melakukan visualisasi histogram pada citra grayscale menggunakan library matplotlib. Jumlah interval histogram ditentukan sebagai 255, yang menghasilkan representasi distribusi intensitas piksel dalam citra grayscale

yang dihasilkan dari tiga persamaan yang berbeda. Sehingga untuk menampilkan gambar hasil konsep pencerahannya adalah sebagai berikut.

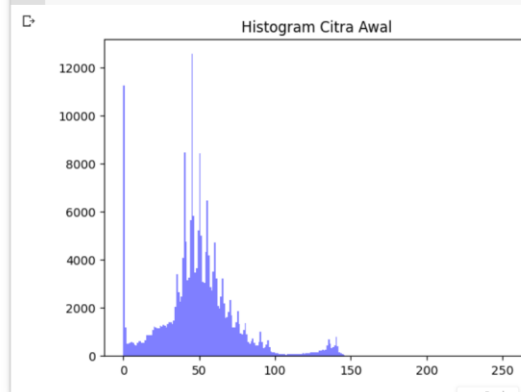
```
1 plt.subplot(2, 2, 1), plt.imshow(img, cmap='gray')
2 plt.title('Citra Awal'), plt.xticks([]), plt.yticks([])
3 plt.subplot(2, 2, 2), plt.imshow(greyA, cmap='gray')
4 plt.title('Penerapan Persamaan (1)'), plt.xticks([]), plt.yticks([])
5 plt.subplot(2, 2, 3), plt.imshow(greyB, cmap='gray')
6 plt.title('Penerapan Persamaan (2)'), plt.xticks([]), plt.yticks([])
7 plt.subplot(2, 2, 4), plt.imshow(greyC, cmap='gray')
8 plt.title('Penerapan Persamaan (3)'), plt.xticks([]), plt.yticks([])
9 plt.show()
```



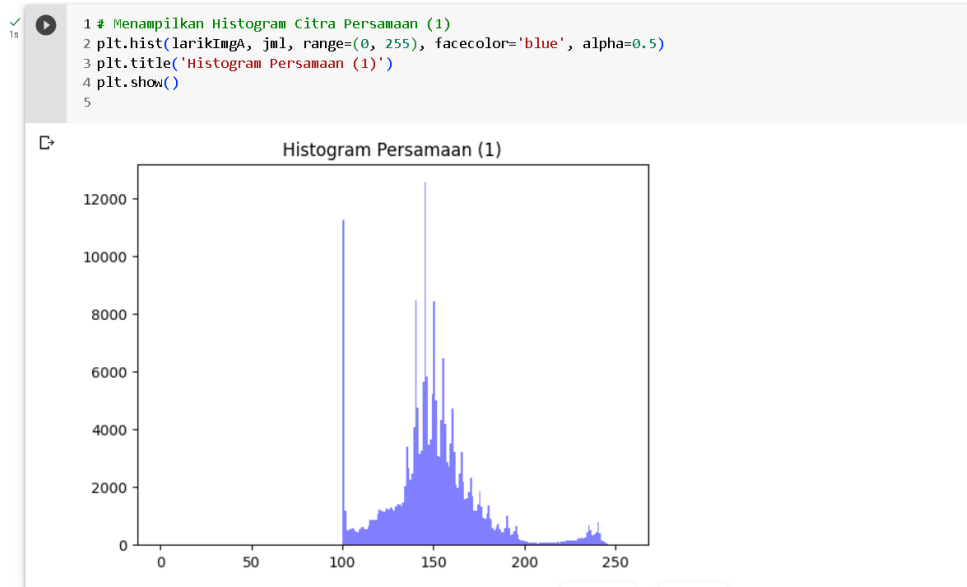
Gambar 3. 18 Menampilkan gambar hasil pencerahan

Pada bagian ini, kita menggunakan fungsi `plt.subplot` untuk membuat tata letak gambar dalam bentuk grid 2x2. Setiap sub-plot dalam grid akan menampilkan citra dengan penerapan persamaan yang berbeda. Terakhir, `plt.show()` digunakan untuk menampilkan semua sub-plot dalam satu tampilan. Kemudian lanjut, untuk menampilkan hasil histogramnya dari gambar di atas.

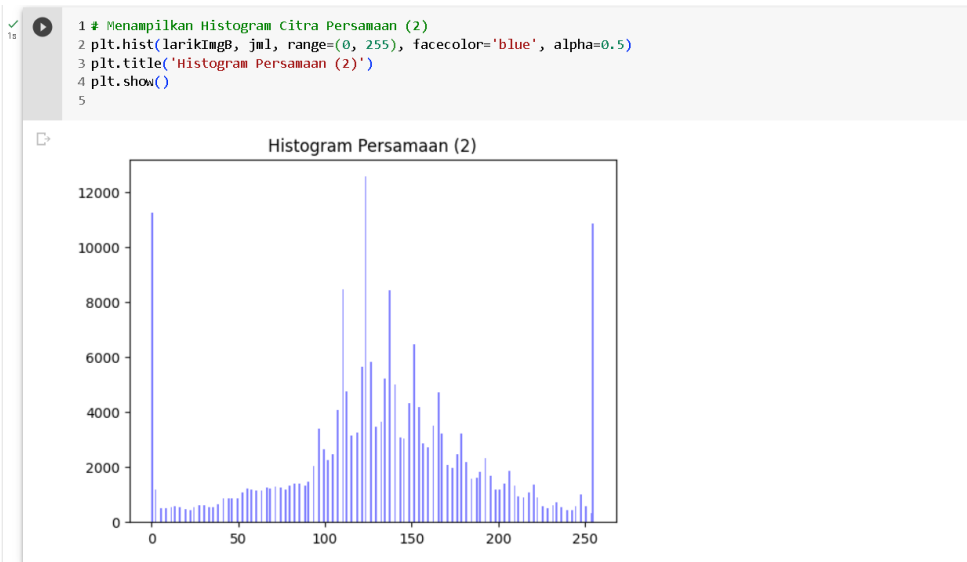
```
1 # Menampilkan Histogram Citra Awal
2 plt.hist(larikImg, jul, range=(0, 255), facecolor='blue', alpha=0.5)
3 plt.title('Histogram Citra Awal')
4 plt.show()
5
```



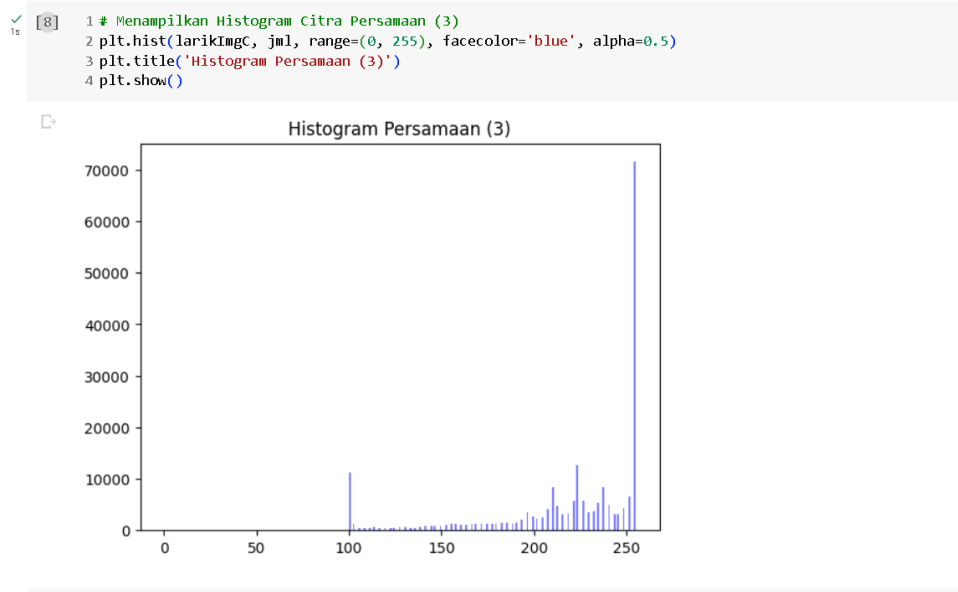
Gambar 3. 19 Histogram untuk citra awal



Gambar 3. 20 Histogram untuk persamaan pertama konsep kecerahan



Gambar 3. 21 Histogram untuk persamaan kedua konsep kecerahan



Gambar 3. 22 Histogram untuk persamaan ketiga konsep kecerahan

Pada hasil histogram di atas, kita menggunakan fungsi `plt.hist` untuk menghitung dan menampilkan histogram dari masing-masing citra. Parameter `larikImg` merupakan larik data piksel dari citra awal yang telah diambil menggunakan metode `getdata()`. Parameter `jml` digunakan untuk menentukan jumlah interval atau bin dalam histogram yang berdasarkan pada gambar 3.17 tadi dimana `jml=255`. Kemudian untuk parameter `range=(0, 255)` digunakan untuk mengatur rentang nilai piksel yang akan digunakan dalam histogram, yaitu 0 hingga 255. Lalu, `facecolor='blue'` dan `alpha=0.5` digunakan untuk mengatur warna dan transparansi dari histogram. Selanjutnya, `plt.title` digunakan untuk memberikan judul pada histogram. Terakhir, `plt.show()` digunakan untuk menampilkan histogram tersebut.

Berdasarkan hasil analisis penulis, untuk gambar 3.19 pada histogram untuk citra awal, bisa dilihat bahwa nilai x hanya berada pada rentang 0-150 dimana berarti pada gambar grayscale citra awal tadi banyak mengandung unsur gelap. Kemudian untuk gambar 3.20 pada histogram persamaan (1), bisa dilihat bahwa untuk nilai x nya sudah mulai bergeser mulai dari 100-250, hal ini memndandakan bahwa unsur paling gelap atau hitam pekat tidak ada, hanya dipenuhi dengan unsur warna keabuan yang berada tepat pada sumbu x sekitar 150an, memiliki nilai tertinggi di sumbunya ,hingga dapat tembus di nilai 12000an ke atas. Kemudian kita lanjut pada gambar histogram di persamaan (2), bisa dilihat pada gambar 3.21,



berdasarkan histogram tersebut, kita dapat melihat bahwa intensitas piksel dalam gambar persamaan (2) terdistribusi. Histogram yang dihasilkan terfokus pada rentang intensitas yang luas dan terdistribusi secara merata, hingganya gambar yang dihasilkan cenderung memiliki kontras yang baik. Sedangkan pada gambar histogram untuk persamaan (3), bisa dilihat pada gambar 3.22, berdasarkan histogram tersebut pada sumbu x nilainya hanya berada pada rentang 100-255, dengan nilai sumbu y tertinggi pada sumbu x yang bernilai 255 yang mencapai angka 7000. Sehingga pada gambar 3.18 pada persamaan (3), bisa kita lihat gambar dari citra grayscale yang dicerahkan memiliki struktur yang sangat cerah dengan gambar hampir diliputi dengan warna putih.

## **BAB IV**

### **PENUTUP**

#### **2.1 Kesimpulan**

Citra digital adalah representasi visual dari objek atau scene yang ditangkap menggunakan perangkat elektronik seperti kamera atau scanner. Citra digital terdiri dari piksel-piksel yang membentuk matriks dengan setiap piksel memiliki nilai intensitas atau warna tertentu. Citra digital berbeda dengan citra analog yang merekam informasi secara kontinu, karena citra digital terdiri dari nilai-nilai diskret yang dapat direpresentasikan secara numerik.

Jenis-jenis citra digital dapat dibagi menjadi beberapa kategori. Pertama, citra biner adalah citra yang hanya memiliki dua tingkat intensitas, yaitu hitam dan putih. Kedua, citra grayscale adalah citra dengan tingkat keabuan yang beragam, tetapi tidak memiliki warna. Ketiga, citra warna adalah citra dengan tingkat keabuan dan warna yang berbeda pada setiap pikselnya.

Pengolahan citra digital adalah proses manipulasi dan analisis terhadap citra menggunakan algoritma dan teknik tertentu. Metode pengolahan citra digital melibatkan operasi seperti filtrasi, transformasi, segmentasi, dan analisis fitur. Tujuan dari pengolahan citra digital dapat bervariasi, termasuk meningkatkan kualitas citra, menghapus noise, mendeteksi objek, mengenali pola, dan banyak lagi. Metode pengolahan citra digital terus berkembang dengan kemajuan teknologi dan kebutuhan yang semakin kompleks dalam memanfaatkan informasi visual.

#### **2.2 Saran**

Saran bagi pembaca terhadap makalah ini adalah untuk membaca dengan cermat dan menjaga konsentrasi. Aljabar linear dalam pengolahan citra digital merupakan topik yang kompleks, oleh karena itu, penting bagi pembaca untuk memahami setiap konsep dan aplikasi yang dijelaskan dalam makalah. Dalam membaca makalah ini, disarankan untuk membaca dengan teliti agar tidak melewatkan informasi penting. Jika pembaca mengalami kesulitan dalam memahami beberapa bagian, disarankan untuk membaca ulang atau mencari referensi tambahan untuk mendapatkan pemahaman yang lebih baik.

## DAFTAR PUSTAKA

Antoniadis Panagiotis .2022. How to Convert an RGB Image to a Grayscale.

<https://www.baeldung.com/cs/convert-rgb-to-grayscale>

Diakses pada 7 Juni 2023

Adhisusanto & Abdul Kadir. 2013. Pengolahan Citra Teori dan Aplikasi.

Yogyakarta: Andi Offset.

Budi Rahardjo .2019. Pemrograman Phyton Bandung.Informatika.

Hanna Arini Parhusip. 2019. Pemrograman Pyton. Yogyakarta: Andi Offset.

Kadir, Abdul. 2019. Pemrograman OpenCV & Phyton. PT.Gramedia. Jakarta.

Munir Rinaldi.2006. “Aplikasi Image Thresholding Untuk Segmentasi Objek”,  
Seminar Nasional Aplikasi Teknologi Informasi 2006

Putra Darma. 2010. Pengolahan Citra Digital. Yogyakarta: Andi Offset.

R. C. Gonzalez and R. E. Woods.2008., Digital Image Processing, Third Edition.

Rianto Sugeng.2019. Dasar-Dasar Pengelolaan Citra Menggunakan Python.  
Universitas Brawijaya. Malang.