Pengolahan Citra Dan Visi Komputer

PCVK

RTI235007

Analisis Intensitas dan Histogram Citra

TEACHING TEAM

MATA KULIAH PENGOLAHAN CITRA DAN VISI KOMPUTER

Analisis Intensitas dan Histogram Citra

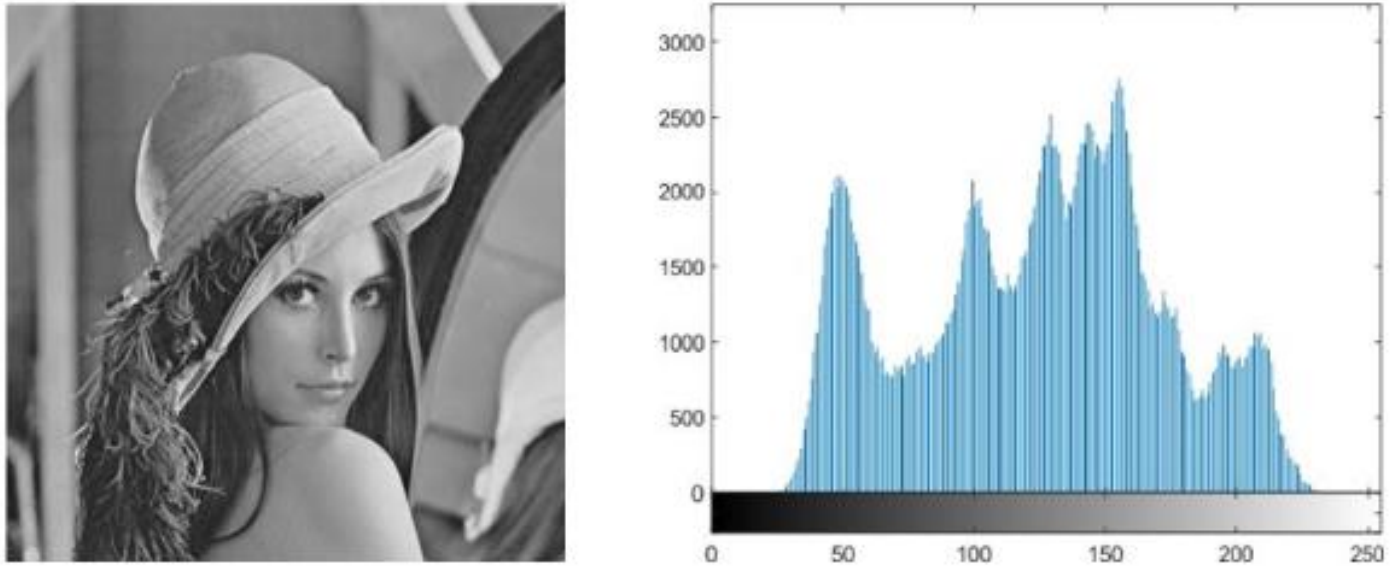
- Konsep histogram citra
- Histogram equalization
- Dithering

Konsep Histogram Citra

Distribusi frekuensi nilai intensitas piksel dalam suatu citra

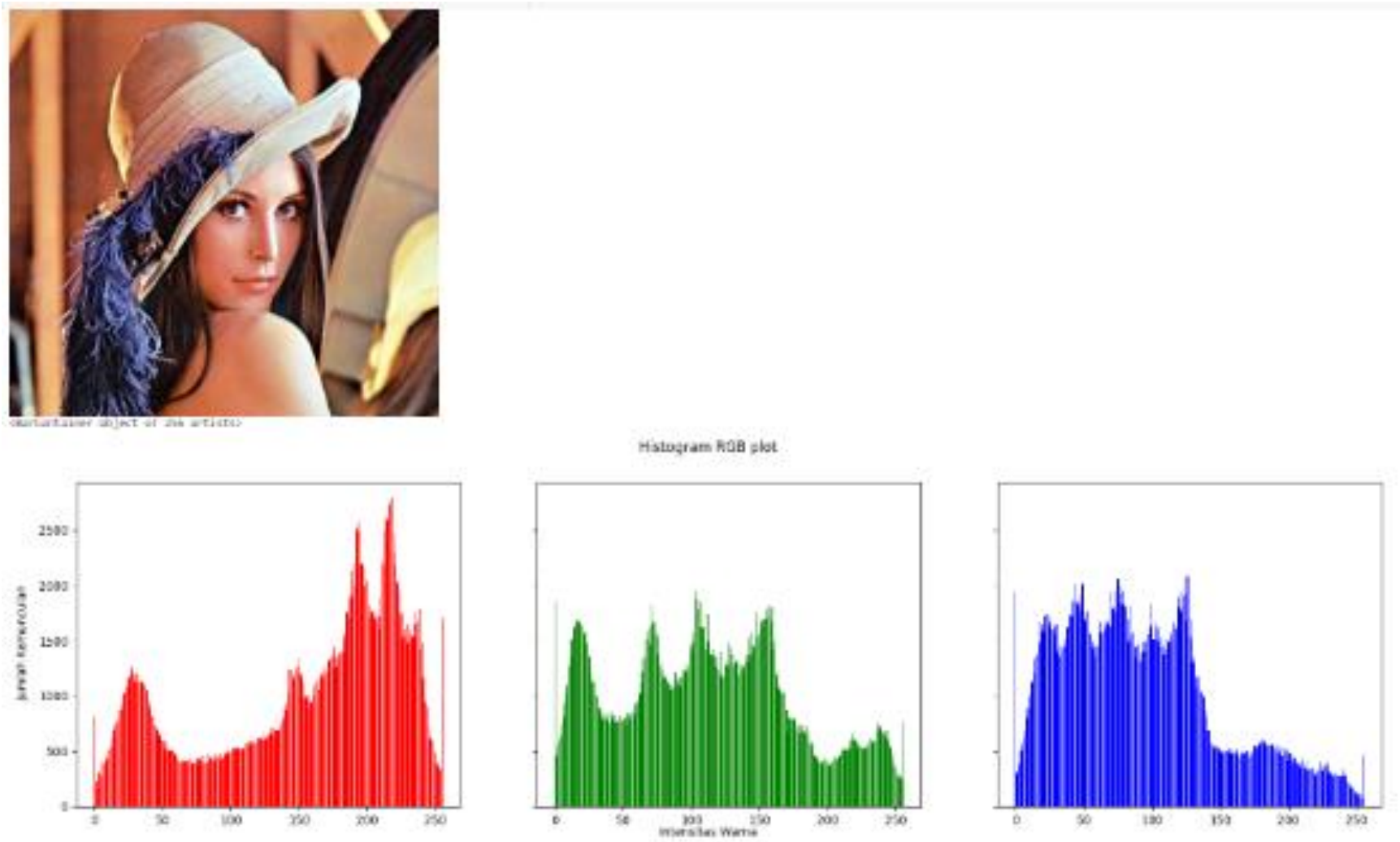
Definisi Histogram Citra

- merupakan diagram yang menggambarkan distribusi frekuensi nilai intensitas piksel dalam suatu citra. Sumbu horizontal merupakan nilai intensitas piksel sedangkan sumbu vertikal merupakan frekuensi/jumlah piksel.



(a) Citra Grayscale (b) Histogram Citra

Histogram dari sebuah citra Grayscale



Histogram dari sebuah citra berwarna

Menghitung histogram :

- Misalkan sebuah citra mempunyai L level nilai keabuan, [0, L-1].
- Hitung frekuensi kemunculan setiap nilai keabuan j dengan cara menghitung jumlah pixel yang mempunyai nilai keabuan tersebut.
- Perhitungan ini dilakukan untuk j = 0, 1, 2, …, L – 1.

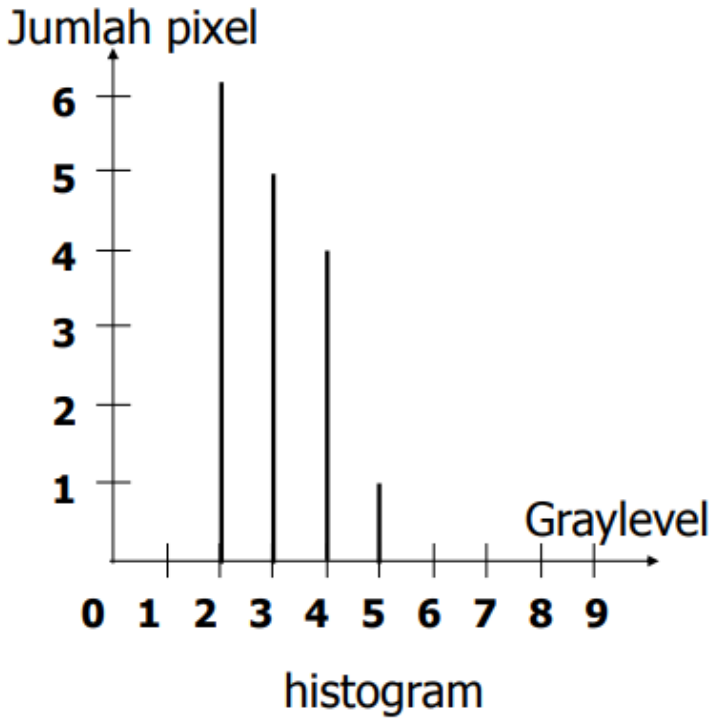
Sumber: ALI JAVED, Digital Image Processing, Chapter # 3, Image Enhancement in Spatial Domain

Contoh:

2	3	3	2
4	2	4	3
3	2	3	5
2	4	2	4

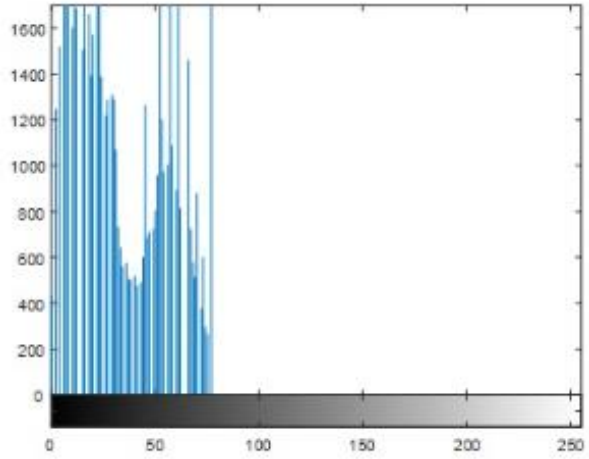
4x4 image

Grayscale = [0,9]



histogram

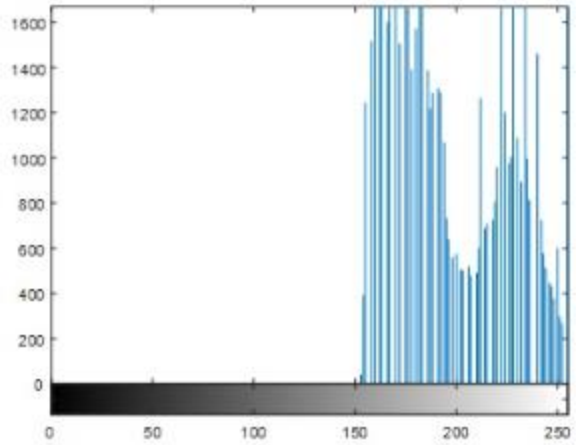
(a) Citra gelap



(b) Histogram citra gelap

Citra gelap merupakan citra yang memiliki banyak piksel dengan nilai intensitas mendekati 0. Distribusi nilai intensitas citra gelap cenderung berada pada daerah sebelah kiri histogram.

(a) Citra terang



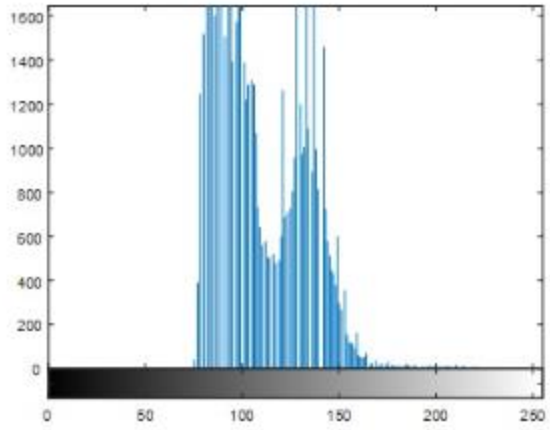
(b) Histogram citra terang

Citra terang merupakan citra yang memiliki banyak piksel dengan nilai intensitas mendekati 255. Distribusi nilai intensitas citra terang cenderung berada pada daerah sebelah kanan histogram.

Sumber: https://pemrogramanmatlab.com/2017/07/26/histogram-citra/

Citra dengan kontras rendah merupakan citra yang memiliki range nilai intensitas yang sempit. Histogram citra pada disamping ini menunjukkan bahwa citra berada pada range nilai intensitas 74-224.

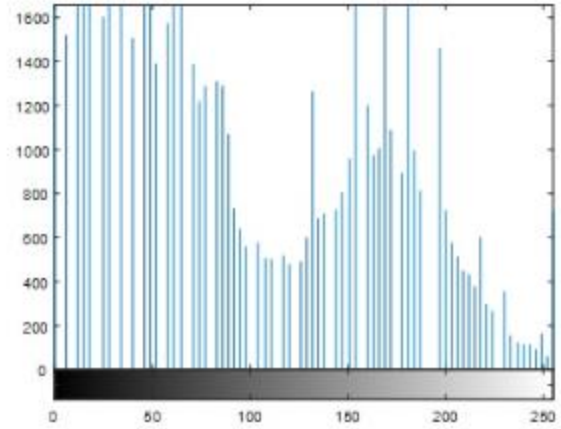
(a) Citra dengan kontras rendah



(b) Histogram citra dengan kontras rendah

Citra dengan kontras tinggi merupakan citra yang memiliki range nilai intensitas yang lebar. Histogram citra pada citra kontras tinggi menunjukkan bahwa citra berada pada range nilai intensitas 0-255.

(a) Citra dengan kontras tinggi



(b) Histogram citra dengan kontras tinggi

Sumber: https://pemrogramanmatlab.com/2017/07/26/histogram-citra/

Normalisasi Histogram

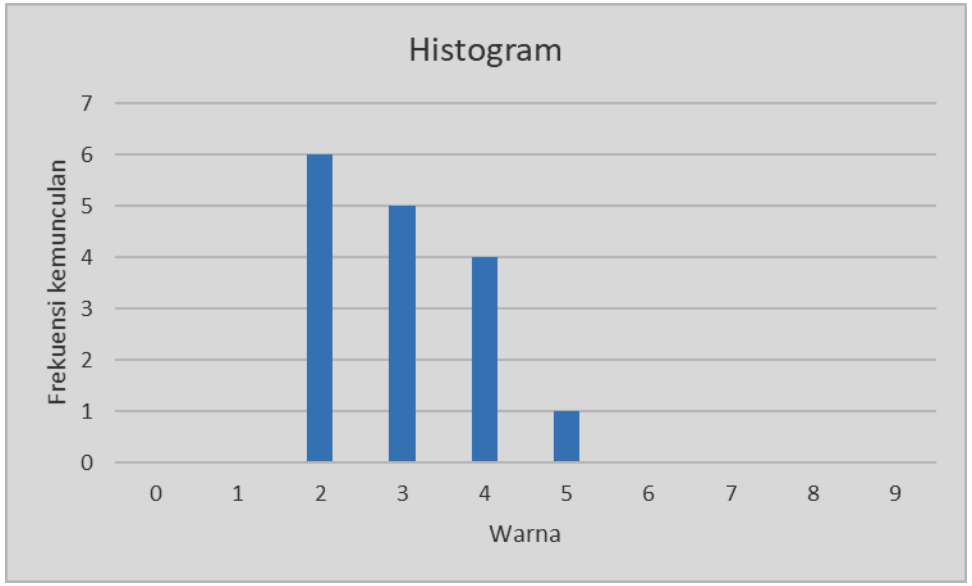
Mengubah nilai frekuensi kemunculan warna menjadi probabilitas kemunculan warna

Histogram

Citra Grayscale 9 warna

2	3	3	2
4	2	4	3
3	2	3	5
2	4	2	4

Warna	Frekuensi
0	0
1	0
2	6
3	5
4	4
5	1
6	0
7	0
8	0
9	0



Rumus

$$p_i = \frac{n_i}{MxN}$$

p_i= Peluang kemunculan warna i

n_i = Frekuensi kemunculan warna i

M = Jumlah baris pada citra digital

N = Jumlah kolom pada citra digital

NORMALISASI HISTOGRAM

Citra Grayscale 9 warna

2	3	3	2
4	2	4	3
3	2	3	5
2	4	2	4

$n_2 = 6$

$M = 4$

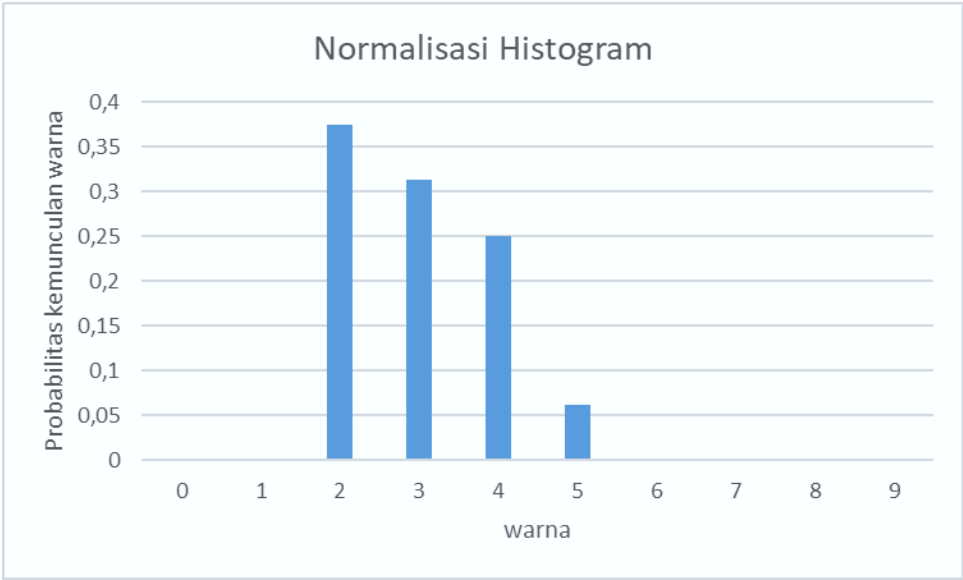
$N = 4$

$$p_2 = \frac{n_2}{MxN}$$

$$= \frac{6}{4x4}$$

$$= 0{,}375$$

Warna	Frekuensi	Probabilitas	
0	0	0/16	0
1	0	0/16	0
2	6	6/16	0,375
3	5	5/16	0,3125
4	4	4/16	0,25
5	1	1/16	0,0625
6	0	0/16	0
7	0	0/16	0
8	0	0/16	0
9	0	0/16	0



Histogram Equalization

Pemrosesan citra yang digunakan untuk ***meningkatkan kontras dalam gambar*** *dengan* ***meratakan distribusi warna pada citra digital***

Definisi Histogram Equalization

- Histogram Equalization adalah sebuah teknik dalam pemrosesan citra yang digunakan untuk **meningkatkan kontras dalam gambar**. Tujuan utama dari histogram equalization adalah untuk membuat distribusi intensitas piksel dalam gambar menjadi lebih merata sehingga gambar menjadi lebih kontras dan lebih tajam
- Walaupun tidak dapat dibuktikan bahwa bentuk histogram-nya akan seragam namun dengan *Histogram Equalization* dapat dipastikan histogram-nya akan lebih merata.
- Perataan histogram diperoleh dengan mengubah derajat keabuan sebuah piksel (r) dengan derajat keabuan yang baru (s) dengan sebuah fungsi transformasi (T) (Gonzalez & Woods, 2002). *Histogram Equalization* dilakukan dengan membuat transform function dari histogram image yang telah dinormalisasi. Berikut adalah persamaan untuk transform function:

$$s_k = (L-1)\sum_{i=0}^{k} p_i$$

$s_k = warna\ baru\ dari\ warna\ k$

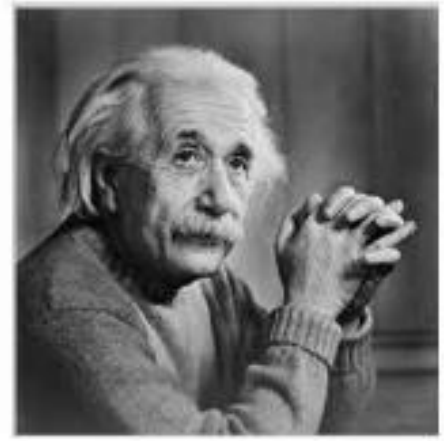
$L = Jumlah\ variasi\ warna\ pada\ citra$

$k = 0, 1, 2, 3 \ldots (L-1)$

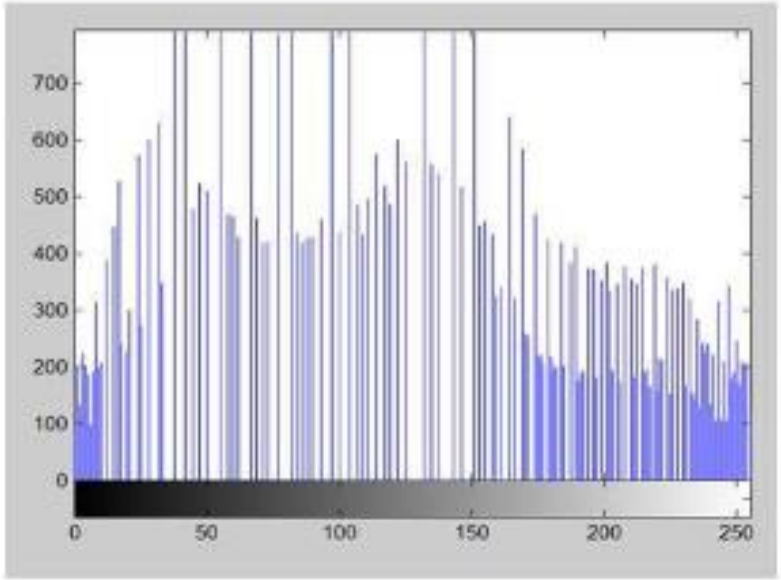
$p_i = probabilitas\ warna\ i$

New Image

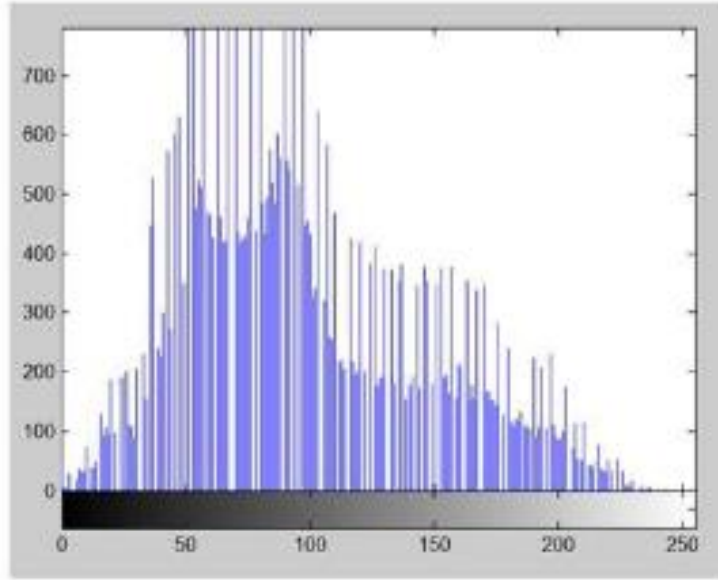
Old image



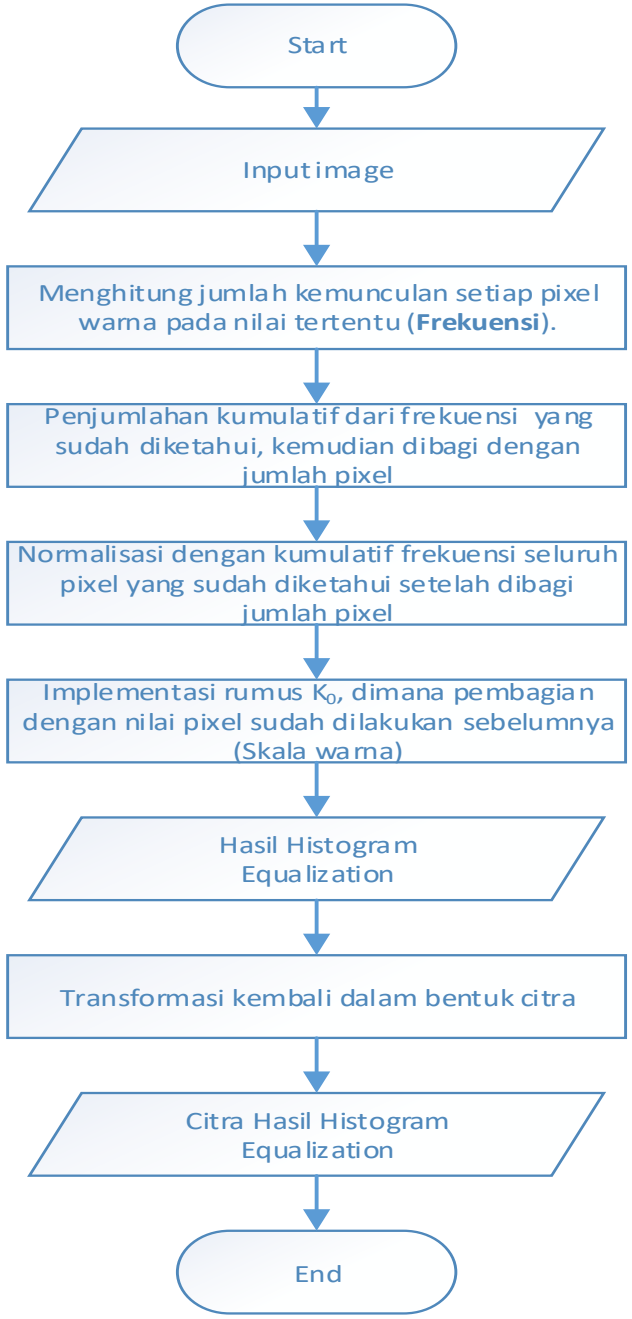
New Histogram



Old Histogram



Flowchart perbaikan citra digital dengan Histogram Equalization



HISTOGRAM EQUALIZATION

Warna	Probabilitas	Warna Baru
0	0	0
1	0	0
2	0,375	3
3	0,3125	5,5=6
4	0,25	7,5 =7
5	0,0625	8
6	0	8
7	0	8
8	0	8

$$s_k = (L-1)\sum_{i=0}^{k} p_i$$

$k = 0$

$$s_0 = (9-1)\sum_{i=0}^{0} p_i$$
$$= 8x0$$
$$= 0$$

$k = 1$

$$s_1 = (9-1)\sum_{i=0}^{1} p_i$$
$$= 8x(p_0 + p_1)$$
$$= 8x(0+0)$$
$$= 0$$

$k = 2$

$$s_2 = (9-1)\sum_{i=0}^{2} p_i$$
$$= 8x(p_0 + p_1 + p_2)$$
$$= 8x(0+0+0,375)$$
$$= 8$$

k =3

$$s_3 = (9-1)\sum_{i=0}^{3} p_i$$
$$= 8x(p_0 + p_1 + p_2 + p_3)$$
$$= 8x(0+0+0,375+0,3125)$$
$$= 5,5$$

k =4

$$s_4 = (9-1)\sum_{i=0}^{4} p_i$$
$$= 8x(p_0 + p_1 + p_2 + p_3 + p_4)$$
$$= 8x(0+0+0,375+0,3125+0,25)$$
$$= 7,5$$

k =5

$$s_5 = (9-1)\sum_{i=0}^{5} p_i$$
$$= 8x(p_0 + p_1 + p_2 + p_3 + p_4 + p_5)$$
$$= 8x(0+0+0,375+0,3125+0,25+0,0625$$
$$= 8$$

k =6 s_6 =8

k =7 s_6 =8

k =8 s_6 =8

HISTOGRAM EQUALIZATION

Warna	Warna Baru
0	0
1	0
2	3
3	6
4	7
5	8
6	8
7	8
8	8

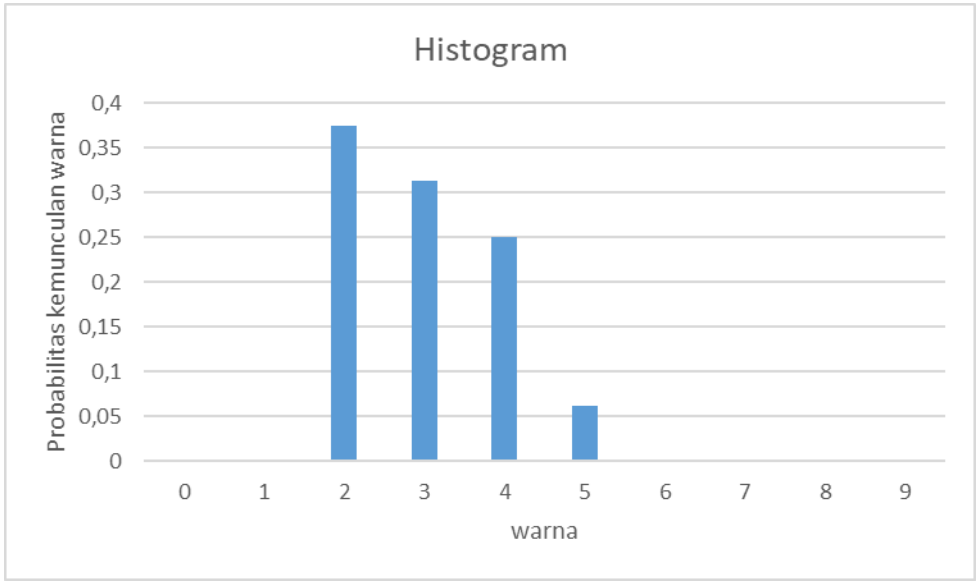
Citra Grayscale 9 warna

2	3	3	2
4	2	4	3
3	2	3	5
2	4	2	4

Citra Grayscale baru

3	6	6	3
7	3	7	6
6	3	6	5
3	7	3	7

Warna	Frekuensi	Probabilitas	
0	0	0/16	0
1	0	0/16	0
2	0	0/16	0
3	6	6/16	0,375
4	0	0/16	0
5	1	1/16	0,0625
6	5	5/16	0,3125
7	4	4/16	0,25
8	0	0/16	0
9	0	0/16	0



Dithering & Error Difussion

Pengolahan citra digital khususnya dalam konteks konversi citra dengan palet warna yang terbatas

Dithering dan Error Diffusion

- **Dithering** dan **Error Diffusion** adalah dua teknik yang berkaitan dalam pengolahan citra digital, khususnya dalam konteks konversi citra dengan palet warna yang terbatas atau pengurangan bit.
- **Dithering** adalah teknik yang digunakan untuk mengurangi jumlah warna yang digunakan dalam citra digital. Ini menciptakan efek warna yang lebih halus dengan menggunakan pola titik atau pola lainnya. Dalam dithering, citra kontinu dikonversi menjadi citra diskrit dengan memilih warna terdekat dari palet warna yang terbatas dan mengatur pola titik atau tekstur sehingga menciptakan ilusi warna dan nuansa yang lebih banyak daripada yang sebenarnya tersedia.
- **Error Diffusion** adalah salah satu algoritma dithering yang digunakan untuk mengurangi jumlah warna dalam citra. Ini bekerja dengan cara mendiffusikan kesalahan kuantisasi (perbedaan antara warna asli dan warna yang dipilih) ke piksel-piksel tetangga. Ini menghasilkan citra yang lebih akurat secara visual dengan mengurangi efek "terdistorsi" yang sering terlihat dalam dithering yang lebih sederhana.

What is Dithering in Image Processing and How it Maintains Image Quality?

- Ada beberapa keadaan dalam pengolahan citra digital yang memerlukan pengurangan ukuran pixel gambar. Pada keadaan ini, dithering adalah proses yang dapat membantu dalam pengurangan ukuran tanpa kehilangan kualitas dan banyak informasi dari data serta membantu meminimalkan kesalahan kuantisasi.
- Contohnya pada beberapa teknologi rendering yang ada saat ini hanya mampu menampilkan sejumlah tingkat keabuan yang terbatas. Salah satu contoh peralatannya adalah printer, alat ini hanya mampu menampilkan warna hitam dan putih untuk gambar monokrom. Hal ini akan mempengaruhi kualitas gambar yang dihasilkan saat melakukan proses cetak dari gambar berwarna ke gambar monokrom. Metode Dithering mampu mengubah citra grayscale dengan 256 tingkat keabuan menjadi citra monokrom/citra biner dengan tetap mempertahankan kualitas citra yang dihasilkan. Metode ini dilakukan dengan mengeksploitasi sifat dari sistem visual manusia dalam memberikan kesan bahwa citra tersebut sifatnya kontinu pada semua detail citra meskipun hanya memiliki dua tingkat dalam rendering.

https://digital-photography-school.com/5-reasons-why-converting-to-black-and-white-may-improve-your-image/

Sumber: https://digital-photography-school.com/5-reasons-why-converting-to-black-and-white-may-improve-your-image/

Proses mengubah suatu gambar menjadi gif memerlukan pengurangan warna gambar menjadi 256. Gambar di bawah ini merupakan representasi proses kuantisasi pada suatu gambar.

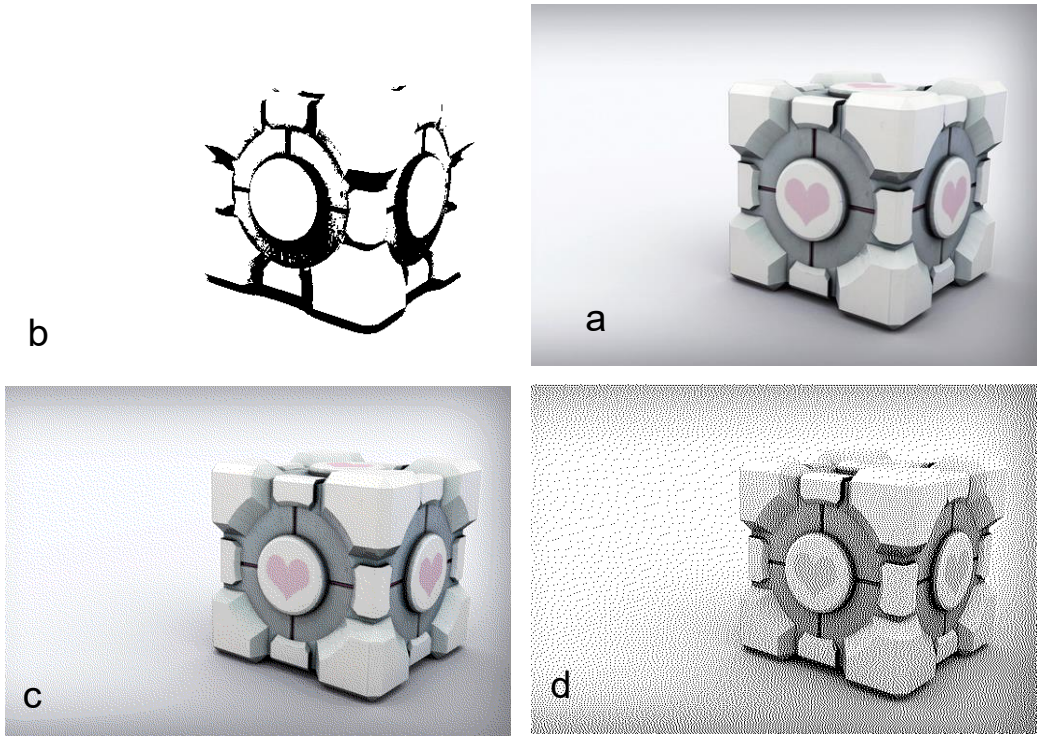
Dapat melihat ukuran gambar dan kualitas representasi gambar dikurangi. Di sini kita telah melihat bagaimana kuantisasi gambar menyebabkan hilangnya informasi dan ini dapat dianggap sebagai kesalahan kuantisasi.

Sumber: https://www.cs.princeton.edu/courses/archive/fall00/cs426/lectures/dither/dither.pdf

Dithering

- adalah suatu teknik dalam pemrosesan citra digital yang digunakan untuk mengurangi ketidaksempurnaan atau efek aliasing yang terjadi saat mengonversi citra berkontinu ke dalam citra berpiksel. Ketika citra berkontinu, seperti foto, diubah menjadi citra berpiksel, masalah aliasing dapat muncul, menghasilkan efek serrated atau meruncing pada tepi objek, terutama saat mengurangkan resolusi.
- Dithering bekerja dengan cara menyisipkan noise atau pola piksel berbeda dalam citra untuk mengaburkan batas-batas tajam yang mungkin terbentuk saat mengonversi citra ke citra berpiksel. Ini menciptakan ilusi perbedaan warna atau nuansa pada tingkat piksel yang lebih rendah daripada yang sebenarnya ada dalam palet warna.
- Penggunaan dithering bisa bermanfaat dalam beberapa konteks, seperti mencetak citra berwarna pada printer monokrom, mengonversi citra berwarna tinggi menjadi citra berwarna rendah, atau mengurangi efek serrated pada grafis komputer.
- Dithering membantu mengatasi keterbatasan dalam representasi warna dan detail pada citra berpiksel dengan cara menciptakan ilusi visual yang lebih baik.

Contoh Dithering



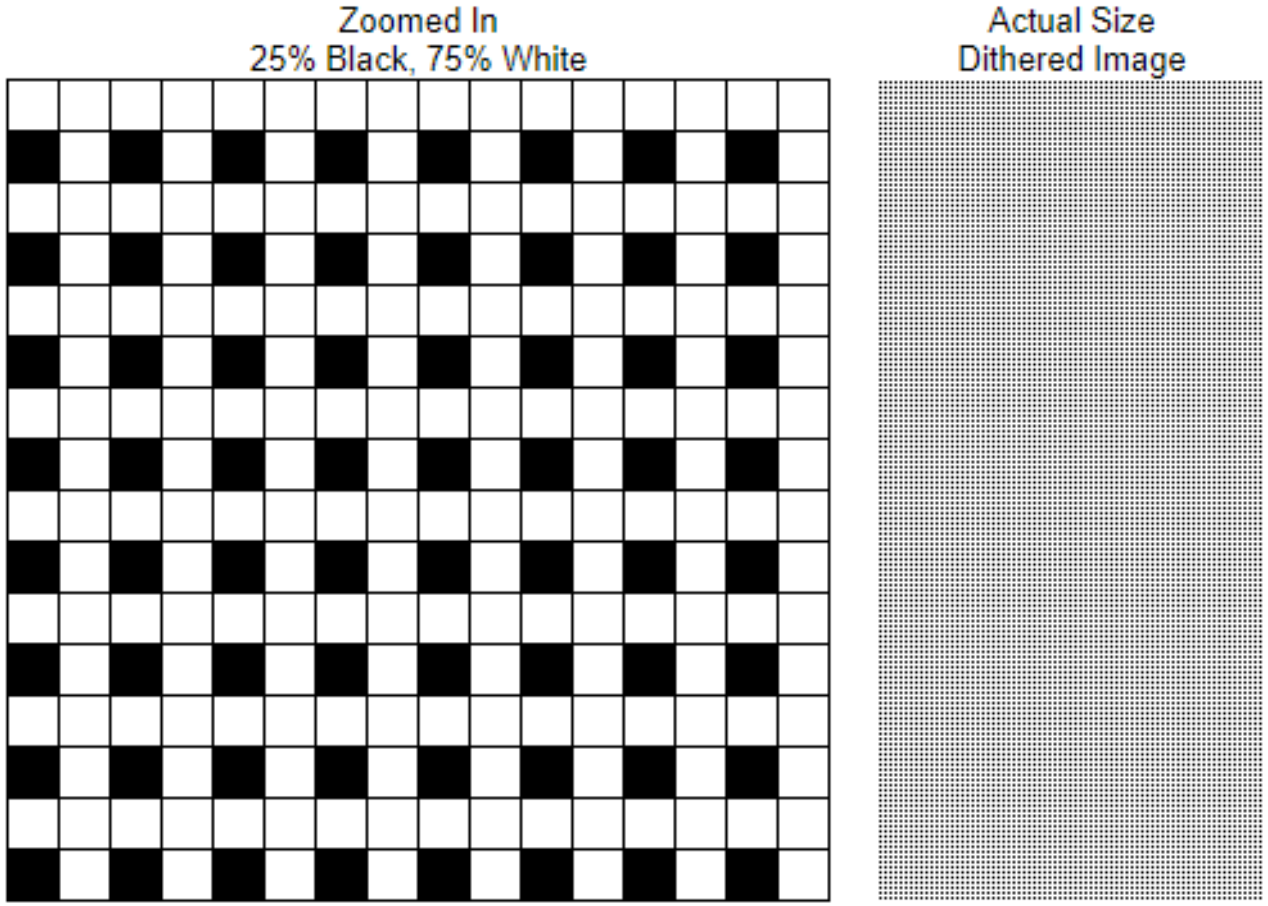
- a. original image
- b. black-and-white printer
- c. dithering RGB
- d. dithering black-and-white version

Sumber : https://tannerhelland.com/2012/12/28/dithering-eleven-algorithms-source-code.html

Macam-macam Teknik Dithering

1. **Ordered Dithering**: Teknik ini menggunakan matriks titik teratur untuk memutuskan apakah piksel pada gambar akan dinyatakan sebagai warna yang lebih terang atau lebih gelap berdasarkan perbandingan dengan elemen-elemen dalam matriks tersebut.
2. **Error Diffusion Dithering**: Dalam teknik ini, kesalahan dari konversi warna piksel sebelumnya disebarkan ke piksel-piksel berikutnya. Ini menghasilkan gambar yang lebih halus dan lebih mirip dengan gambar asli.
3. **Floyd-Steinberg Dithering**: Ini adalah salah satu metode error diffusion yang paling terkenal. Ini membagi kesalahan ke piksel-piksel tetangga dengan bobot tertentu untuk menghasilkan hasil yang lebih baik.
4. **Random Dithering**: Dalam teknik ini, nilai piksel dinyatakan secara acak sebagai warna yang lebih terang atau lebih gelap dengan peluang tertentu. Ini memberikan hasil yang acak tetapi kurang halus.
5. **Cluster Dithering**: Metode ini mengelompokkan piksel-piksel berdasarkan kesamaan warna dan kemudian mewakili setiap kelompok dengan satu warna dari palet yang tersedia.
6. **Halftone Dithering**: Teknik ini mengubah gambar menjadi pola titik yang mirip dengan titik-titik yang digunakan dalam cetakan halftone tradisional.
7. **Atkinson Dithering**: Ini adalah metode error diffusion yang membagi kesalahan dengan bobot tertentu ke piksel-piksel tetangga.

- Setiap teknik dithering memiliki kelebihan dan kekurangan, dan pemilihan teknik tergantung pada tujuan akhir dari penggunaan gambar tersebut dan kualitas yang diinginkan.



Representasi ilusi cahaya abu-abu dengan menggunakan warna hitam putih.

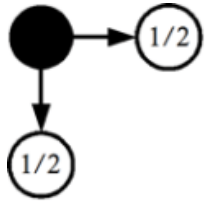
Sumber gambar: https://www.graphicsacademy.com/grid2.gif

Error Difussion

- Error Diffusion adalah salah satu tipe halftoning dimana proses kuantisasi didistribusi pada piksel tetangga yang belum diproses. Penggunaan utamanya adalah untuk mengubah kedalaman image menjadi lebih kecil, tetapi kualitas gambar tidak terlalu buruk.
- Proses ini biasanya digunakan pada teknologi cetak, karena jumlah warna tinta yang tidak terlalu banyak sehingga ilusi warna terbentuk dari teknik ini.
- Error diffusion digolongkan sebagai area operation, karena yang dilakukan oleh error diffusion pada satu pixel akan mempengaruhi hasil perhitungan operasi pixel lainnya.
- Error diffusion juga memiliki kemampuan untuk memperkuat tepi dari citra. Hal ini dapat membuat teks dalam citra menjadi lebih mudah terbaca daripada teknik halftoning yang lain.

Bentuk Sederhana Error Difussion

- Error diffusion bekerja dengan membandingkan warna asli pixel dengan warna terdekat yang ditentukan dan menghitung selisihnya. Selisih ini dinamakan dengan error.
- Porsi error ini kemudian dibagikan ke pixel tetangga sehingga menyebabkan errornya terdifusi dan dinamakan “Error Diffusion”.



- Setengah error dari pixel yang diproses (ditunjukkan dengan titik hitam) terdifusi setengahnya ke pixel sebelah kanan dan setengahnya lagi ke pixel bawahnya.
- Jumlah total error yang terdifusi tidak boleh melebihi satu.
- Hal penting lain yang perlu dicatat adalah ketika porsi error terdifusi ke tetangganya, tetangga tersebut memiliki nilai pixel antara 0 – 255.
- Jika nilainya diluar 0 – 255, maka nilai akhir perlu di truncate.

Pseudocode Error Difusi Sederhana

1	**Error = warna asli – warnaTerdekat**
2	SetPixel(x+1, y) = Truncate(GetPixel(x+1,y) + 0.5 * error)
3	SetPixel(x, y+1) = Truncate(GetPixel(x, y+1) + 0.5 * error)

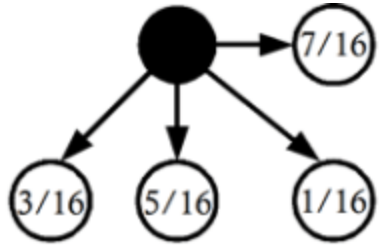
Beberapa metode Error Diffusion

1. Floyd and Steinberg
2. Jarvis, Judice dan Ninke dari Bell Labs

Floyd and Steinberg

- Floyd dan Steinberg mendesripsikan system pembentuk error diffusion pada citra digital dengan menggunakan kernel sederhana:

$$\frac{1}{16}\begin{bmatrix} - & \# & 7 \\ 3 & 5 & 1 \end{bmatrix}$$



Dimana – menunjukkan pixel yang sudah diproses, # menunjukkan pixel saat ini yang sedang diproses. Atau dalam bentuk diagram node ditunjukkan sbb:

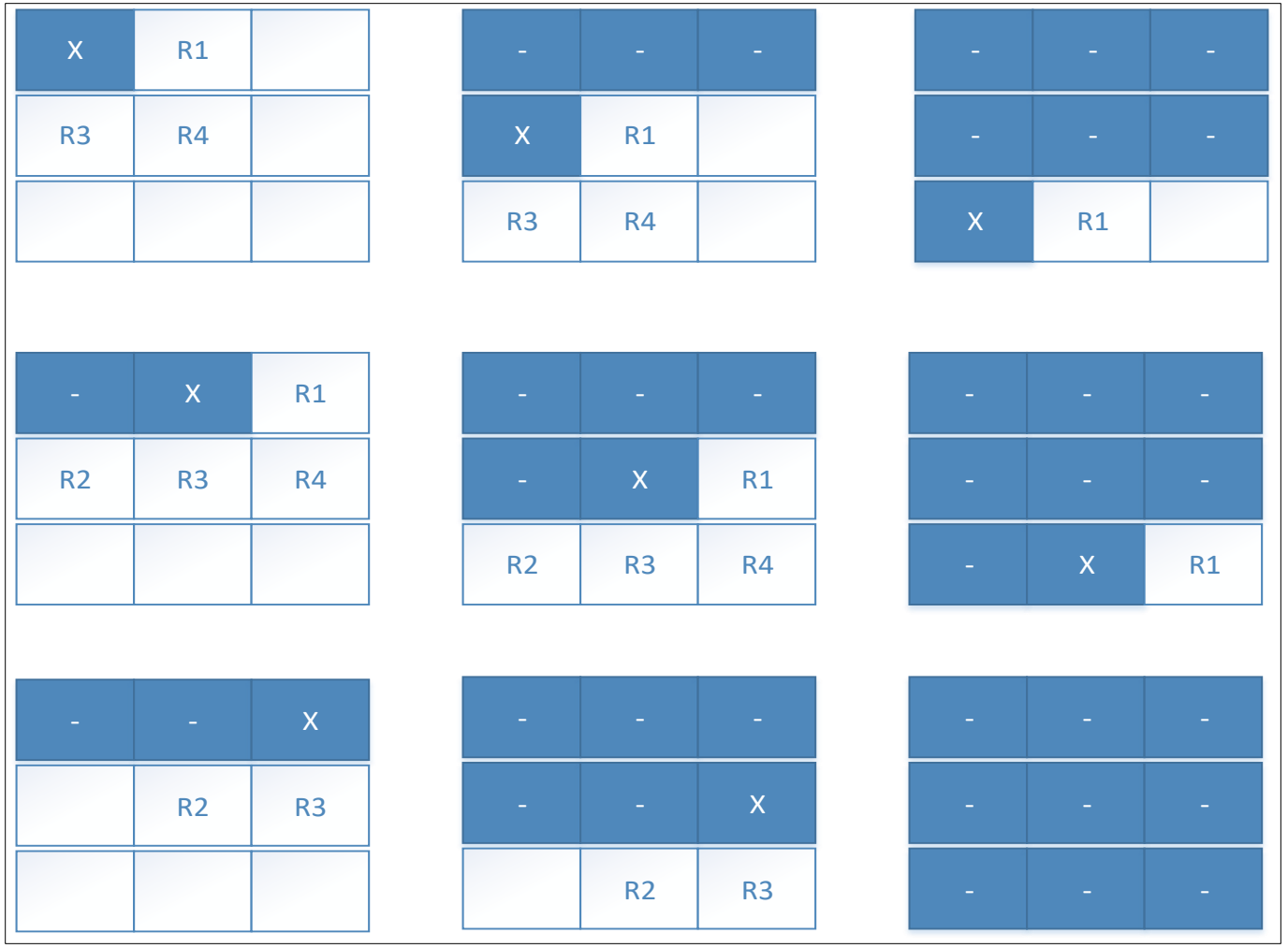
Rumus Dithering Floyd and Steinberg

- Rumus pertama (R1) → $I_{acc}(i+1,j) = I_{acc}(i+1,j) + \frac{7}{16}e$
- Rumus pertama (R2) → $I_{acc}(i+1,j+1) = I_{acc}(i+1,j+1) + \frac{3}{16}e$
- Rumus pertama (R3) → $I_{acc}(i,j+1) = I_{acc}(i,j+1) + \frac{5}{16}e$
- Rumus pertama (R4) → $I_{acc}(i-1,j+1) = I_{acc}(i-1,j+1) + \frac{1}{16}e$

- Dimana :
- I_{acc} = Target pixel (pixel yang sedang diproses)
- e = error

Contoh Penggunaan Rumus Dithering Floyd and Steinberg

- Citra berukuran 3 x 3 pixel



Pseudocode Difusi Floyd and Steinberg

1	**Error = warna asli – warnaTerdekat**
2	SetPixel(x+1, y) = Truncate(GetPixel(x+1, y) + 7/16 * error)
3	SetPixel(x-1, y+1) = Truncate(GetPixel(x-1, y+1) + 3/16 * error)
4	SetPixel(x , y+1) = Truncate(GetPixel(x , y+1) + 5/16 * error)
5	SetPixel(x+1, y+1) = Truncate(GetPixel(x+1, y+1) + 1/16 * error)

Jarvis, Judice dan Ninke dari Bell Labs

- Jarvis, Judice dan Ninke dari Bell Labs mendeskripsikan system yang sama yang mereka istilahkan “minimized average error”, menggunakan kernel dengan ukuran yang lebih besar:

$$\frac{1}{48}\begin{bmatrix} - & - & \# & 7 & 5 \\ 3 & 5 & 7 & 5 & 3 \\ 1 & 3 & 5 & 3 & 1 \end{bmatrix}$$

Cara lain penyederhaan jumlah warna

- Jika ingin menyederhanakan jumlah warna tanpa error diffusion, maka cara yang termudah adalah dengan mencari warna terdekatnya, Jika input warna lebih dekat ke merah, maka warnanya akan diubah kemerah.
- Jika lebih dekat ke cyan, maka warnanya diubah ke cyan, dan seterusnya. Hasilnya terlihat pada gambar pada sub bab warna terdekat.

Contoh : Penyederhanaan Gambar Menjadi 8 Warna
Memetakan 16 Juta warna RGB ke dalam 8 warna saja yaitu hitam, merah, hijau, kuning, biru, cyan, magenta, putih

Referensi

- https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2019-2020/07-Image-Histogram.pdf
- https://pemrogramanmatlab.com/2017/07/26/histogram-citra/
- https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_histograms/py_histogram_equalization/py_histogram_equalization.html
- https://analyticsindiamag.com/what-is-dithering-in-image-processing-and-how-it-maintains-image-quality/
- https://tannerhelland.com/2012/12/28/dithering-eleven-algorithms-source-code.html
- https://www.ronja-tutorials.com/post/042-dithering/
- https://surma.dev/things/ditherpunk/
- https://www.sciencedirect.com/topics/engineering/error-diffusion
- https://scipython.com/blog/floyd-steinberg-dithering/
- https://www.youtube.com/watch?v=9zsaegcL_84
- https://www.youtube.com/watch?v=48DAvO7j3zQ

Terimakasih