GLCM
(Gray Level Co-occurrence Matrix)

Prof. Dr. Eng. Fitri Utaminingrum,ST,MT

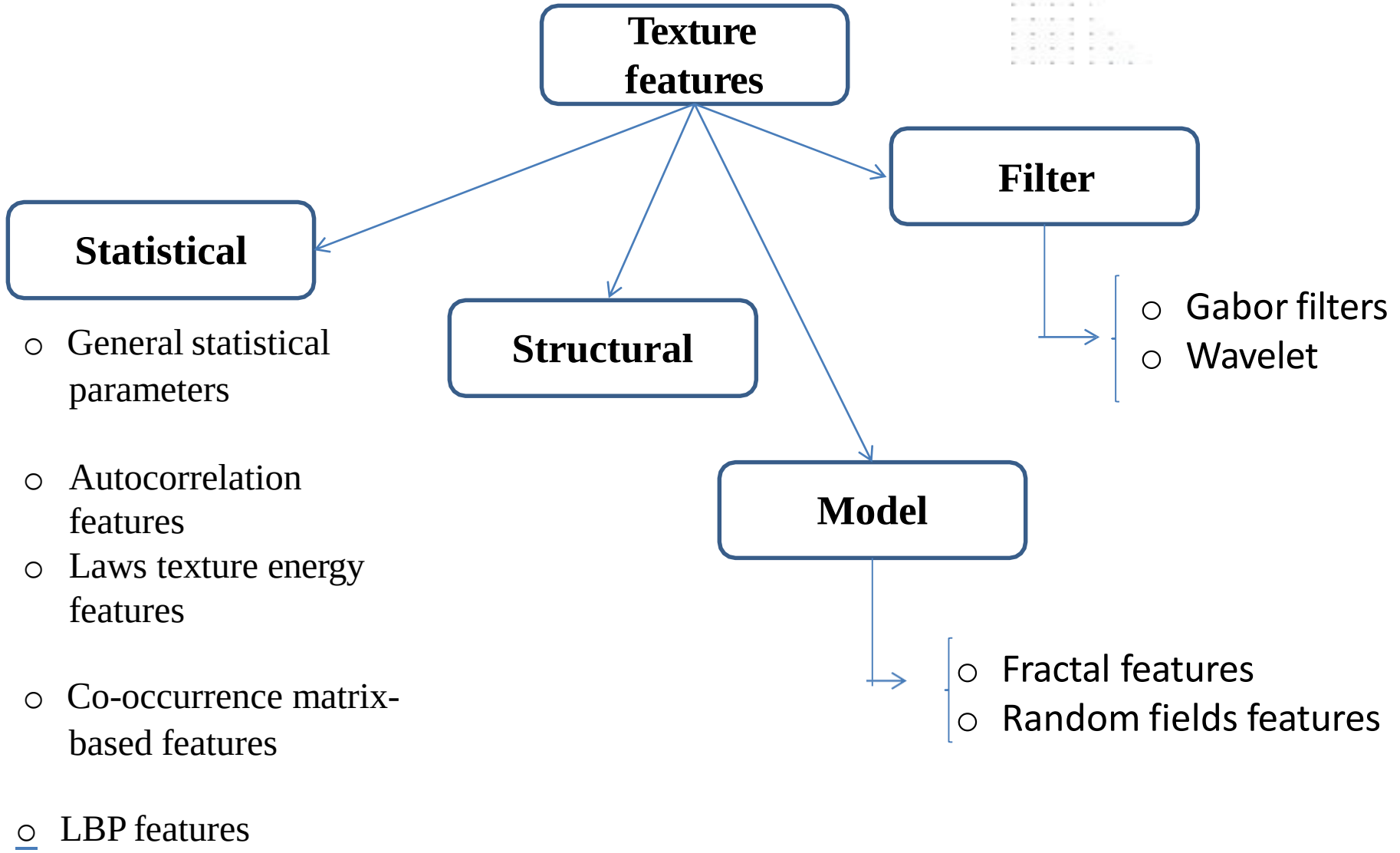
Texture Feature Extraction

Texture Feature Extraction

- Textures can be rough or smooth, vertical or horizontal etc
- Generally they capture patterns in the image data (or lack of them), e.g. repetitiveness and granularity

- ✓ Texture features:
  - ✓ Statistical measures:
    - ✓ Entropy
    - ✓ Homogeneity
    - ✓ Contrast
  - ✓ Wavelets
  - ✓ Fractals

Texture Feature Extraction



Gray Level Co-occurrence Matrix (GLCM)

- *GLCM* merupakan sebuah matriks statistik yang merepresentasikan hubungan spasial antara piksel-piksel dengan intensitas keabuan yang berdekatan dalam citra.
- GLCM berguna dalam analisis tekstur citra dan umumnya digunakan dalam pengolahan citra digital dan pengenalan pola.
- GLCM dihitung dengan cara menghitung kemunculan bersama-sama (co-occurrence) dari pasangan intensitas keabuan pada jarak dan arah tertentu dalam citra.
- Matriks GLCM ini kemudian dapat digunakan untuk mengekstrak berbagai fitur tekstur seperti kontras, kehalusan, dan arah.

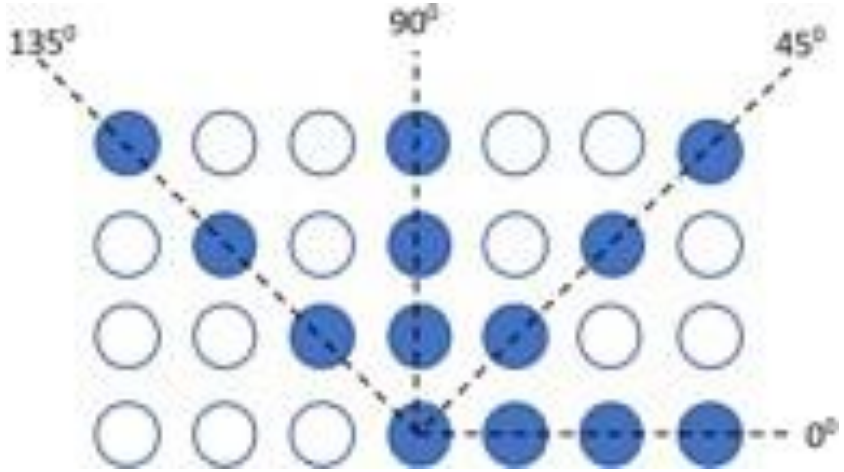
GLCM

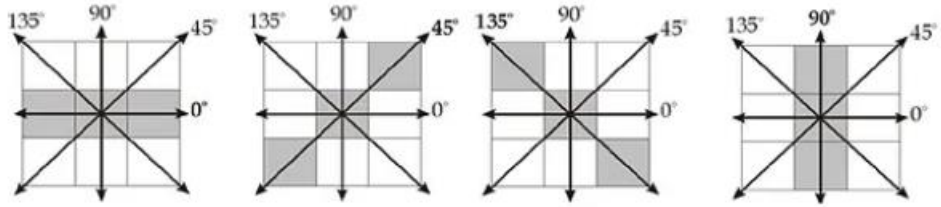
1. Langkah pertama ekstraksi fitur dengan *GLCM* adalah merubah gambar RGB menjadi gambar dalam skala keabuan (*grayscale*), dengan Persamaan :

$$Grayscale = 0{,}299 * R + 0{,}587 * G + 0{,}114 * B$$

GLCM

2. Langkah berikutnya adalah membentuk matriks *GLCM* dari gambar skala keabuan.





d → Relative **distance** between the pixel pair (measured in pixel number. e.g., 1, 2, 3, 4, ……)

θ → Relative **orientation** / rotational angle. (e.g., 0º, 45º, 90º, 135º)

Computation of Co-ocurence Matrix

Image matrix

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

Pixel values: 0,1,2,3. **So,** $N = 4$

So, *size* of CM = 4x4

$d = 1$

θ = horizontal (0°)

Find the **number of co-occurrences** of pixel *i* to the neighboring pixel value *j*

i/j	0	1	2	3
0	#(0,0)	#(0,1)	#(0,2)	#(0,3)
1	#(1,0)	#(1,1)	#(1,2)	#(1,3)
2	#(2,0)	#(2,1)	#(2,2)	#(2,3)
3	#(3,0)	#(3,1)	#(3,2)	#(3,3)

$d = 1$ θ = horizontal (0°)

i/j	0
0	#(0,0)

$d = 1$ θ = horizontal (0°)

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

2			

Computation of Co-ocurence Matrix

$d = 1$ θ = horizontal (0°)

Image

0	0 → 1	1	
0	0 → 1	1	
0 →	2	2	2
2	2	3	3

2	**2**	**1**	0

CM for the Image

i/j	0	1	2	3
0		#(0,1)	#(0,2)	#(0,3)
1				
2				
3				

Computation of Co-ocurence Matrix

$d = 1$ θ = **horizontal**(0°)

Image

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

2	2	1	0
0	2	0	0
0	0	3	1
0	0	0	1

CM for the Image

i/j	0	1	2	3
0	#(0,0)	#(0,1)	#(0,2)	#(0,3)
1	#(1,0)	#(1,1)	#(1,2)	#(1,3)
2	#(2,0)	#(2,1)	#(2,2)	#(2,3)
3	#(3,0)	#(3,1)	#(3,2)	#(3,3)

$d = 1$ θ = vertical(90°)

Image

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

3	0	2	0
0	2	2	0
0	0	1	2
0	0	0	0

CM for the Image

i/j	0	1	2	3
0	#(0,0)	#(0,1)	#(0,2)	#(0,3)
1	#(1,0)	#(1,1)	#(1,2)	#(1,3)
2	#(2,0)	#(2,1)	#(2,2)	#(2,3)
3	#(3,0)	#(3,1)	#(3,2)	#(3,3)

$d = 1 \quad \theta = 0$

1	1	2	3
0	0	1	2
0	1	1	2
2	2	0	1

Contoh gambar

(i,j)	0	1	2	3
0	(0,0)	(0,1)	(0,2)	(0,3)
1	(1,0)	(1,1)	(1,2)	(1,3)
2	(2,0)	(2,1)	(2,2)	(2,3)
3	(3,0)	(3,1)	(3,2)	(3,3)

Pasangan piksel

1	3	0	0
0	2	3	0
1	0	1	1
0	0	0	0

Matriks GLCM dengan d=1 dan θ=0^0

GLCM

3. Matriks *GLCM* diubah menjadi matriks *GLCM* yang simetris dengan cara menjumlahkan dengan *transpose* matriksnya

$$matriks\ GLCM + transpose\ matriks\ GLCM = matriks\ GLCM\ simetris$$

$$\begin{bmatrix} 1 & 3 & 0 & 0 \\ 0 & 2 & 3 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 1 & 0 \\ 3 & 2 & 0 & 0 \\ 0 & 3 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 0 \\ 3 & 4 & 3 & 0 \\ 1 & 3 & 2 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

GLCM

4. Matriks *GLCM* yang simetris tersebut dinormalisasi

$$P_{i,j} = \frac{V_{i,j}}{\sum_{i,j=0}^{N-1} V_{i,j}}$$

$Matriks\ GLCM\ simetris$

$$\begin{bmatrix} 2 & 3 & 1 & 0 \\ 3 & 4 & 3 & 0 \\ 1 & 3 & 2 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \rightarrow Matriks\ GLCM\ yang\ dinormalisasi = \begin{bmatrix} \frac{1}{12} & \frac{1}{8} & \frac{1}{24} & 0 \\ \frac{1}{8} & \frac{1}{6} & \frac{1}{8} & 0 \\ \frac{1}{24} & \frac{1}{8} & \frac{1}{12} & \frac{1}{24} \\ 0 & 0 & \frac{1}{24} & 0 \end{bmatrix}$$

Ekstraksi Fitur GLCM

5. Dari matrik *GLCM* yang dinormalisasi tersebut, kemudian bisa digunakan untuk ekstraksi fitur.

1. **Kontras**: Mencerminkan variasi intensitas antara piksel yang berdekatan dalam citra. Semakin tinggi kontrasnya, semakin besar perbedaan antara nilai intensitas piksel-piksel tersebut. Kontras dapat dihitung dengan rumus:
   $\text{Kontras} = \sum_{i,j}(i-j)^2 \cdot \text{GLCM}(i,j)$
   di mana i dan j adalah intensitas piksel yang berdekatan, dan GLCM(i, j) adalah nilai dalam matriks GLCM yang sesuai.
2. **Kekasaran (Roughness)**: Mengukur ketinggian dan kedalaman tekstur dalam citra. Semakin tinggi nilai kekasarannya, semakin kasar tekstur pada citra tersebut. Kekasaran dapat dihitung dengan rumus:
   $\text{Kekasaran} = \sum_{i,j}(\text{GLCM}(i,j) \cdot (i-\mu)^2)$
   di mana μ adalah nilai rata-rata intensitas dalam citra.
3. **Kesesuaian (Homogeneity)**: Mengukur seberapa dekat distribusi intensitas dalam citra dengan distribusi yang sama. Semakin tinggi nilai kesesuaian, semakin seragam tekstur pada citra tersebut. Keseragaman dapat dihitung dengan rumus:
   $\text{Kesesuaian} = \sum_{i,j} \frac{\text{GLCM}(i,j)}{1+(i-j)^2}$
4. **Energi (Energy)**: Mencerminkan homogenitas atau keragaman intensitas dalam citra. Semakin tinggi nilai energinya, semakin homogen citra tersebut. Energi dapat dihitung dengan rumus:
   $\text{Energi} = \sum_{i,j} \text{GLCM}(i,j)^2$

Ekstraksi Fitur GLCM

1. *Contrast*

Contrast adalah ukuran intensitas atau variasi tingkat abu-abu antara piksel referensi dan tetangganya (antara pasangan piksel). Nilai fitur *contrast* dirumuskan pada Persamaan

$$Contrast = \sum_{n=0}^{L-1} n^2 \left\{ \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} P_{\vec{r}}(i,j) \right\}$$

Dimana $|i - j| = n$

2. *Dissimilarity*

Nilai fitur *dissimilarity* dirumuskan pada Persamaan

$$Dissimilarity = \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} P_{\vec{r}}(i,j)|i - j|$$

Ekstraksi Fitur GLCM

3. *Homogeneity*

Homogeneity atau disebut juga *Inverse Difference Moment (IDM)* mengukur kesamaan dari *coocurrence matrix* dan *diagonal matrix*. Nilai fitur *homogeneity* dirumuskan pada Persamaan $Homogeinity = \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \frac{P_{\vec{r}}(i,j)}{1+(i-j)^2}$

4. *Angular Second Moment* (*ASM*)

ASM mengukur homogenitas suatu gambar. Nilai ASM tinggi jika nilai piksel-pikse sangat mirip. Nilai fitur *ASM* dirumuskan pada Persamaan

- $ASM = \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} P_{\vec{r}}(i,j)^2$
- Dimana $P_{\vec{r}}(i,j)$ adalah nilai piksel pada matriks *GLCM yang dinornalisasi,* L *adalah ukuran matriks GLCM.*

5. *Energy*

Nilai fitur *energy* dirumuskan pada Persamaan

$$Energy = \sqrt{ASM}$$

6. *Correlation*

Korelasi menghitung ketergantungan linear dari nilai-nilai *gray level* dalam matriks GLCM. Korelasi menunjukkan bagaimana piksel referensi terkait dengan tetangganya. Nilai fitur *correlation* dirumuskan pada Persamaan $Correlation = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} P_{\vec{r}}(i,j)\frac{(i-\mu_i)(j-\mu_j)}{\sqrt{\sigma_i{}^2\sigma_j{}^2}}$

Dimana:

$$\mu_i = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} iP_{\vec{r}}(i,j)$$

$$\mu_j = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} iP_{\vec{r}}(i,j)$$

$$\sigma_i = \sqrt{\sum_{i=0}^{L-1}\sum_{j=0}^{L-1} P_{\vec{r}}(i,j)\,(i-\mu_i)^2}$$

$$\sigma_j = \sqrt{\sum_{i=0}^{L-1}\sum_{j=0}^{L-1} P_{\vec{r}}(i,j)\,(j-\mu_j)^2}$$

Original Image

GLCM Contrast

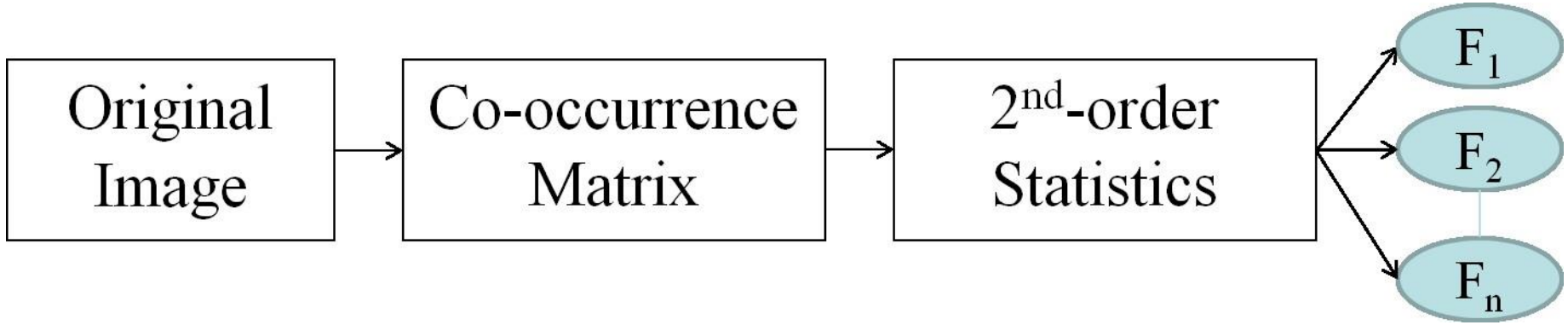
Original Image

GLCM Homogeneity

Features on Coocurence Matrix

- Co-occurrence matrices capture properties of a texture
- But they are *not directly useful* for further analysis

(e.g., comparison of two textures)



11 Numeric features are computed from a matrix

Features on Coocurence Matrix

F1 → Angular Second Moment (ASM) feature

F2 → Contrast feature

F3 → Entropy feature

F4 → Variance feature

F5 → Correlation feature

F6 → Inverse Difference Moment (IDM) feature

F7 → Sum Average feature

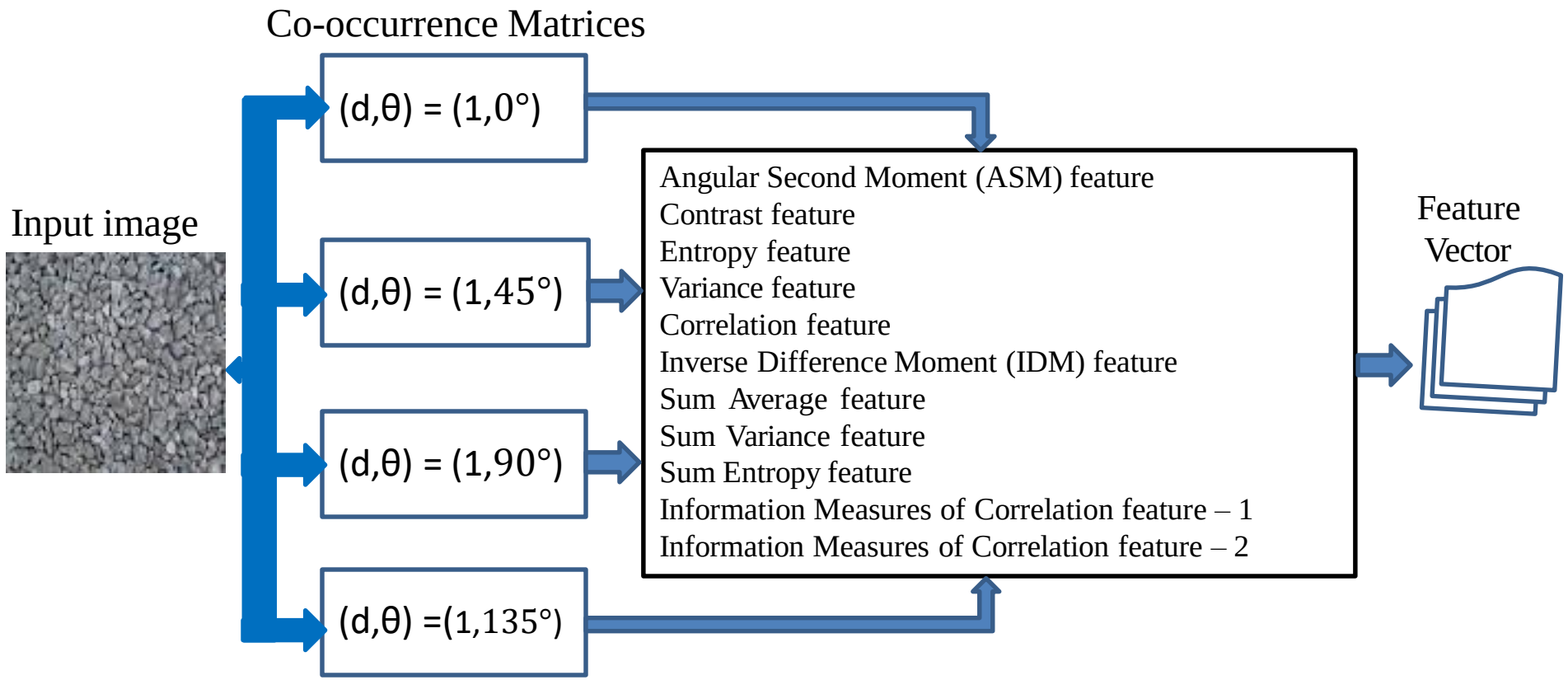
F8 → Sum Variance feature

F9 → Sum Entropy feature

F10 → Information Measures of Correlation feature – 1 (IMC1)

F11 → Information Measures of Correlation feature – 2 (IMC2)

Features on Coocurence Matrix



Features on Coocurence Matrix

Feature	Comment
F2: Contrast	- Have discriminating ability. - Rotationally-variant.
F3: Entropy	- Have strong discriminating ability. - Almost rotational-invariant.
F4: Variance	- Have discriminating ability. - Rotational-invariant.
F5: Correlation	- Have strong discriminating ability. - Rotational-dependent feature.
F7: Sum average	- Characteristics are similar to 'variance'/F4 - Rotational-invariant.
F10: Information Measure of Correlation–1	-It has almost similar pattern of 'sum average'/F7 but vary for various classes - Varies significantly with rotation
F11: Information Measure of Correlation–2	- It is computationally expensive compare to others. - Rotation-variant

Feature	Comment
F1: Angular Second Moment / Energy	- No distinguishing ability
F6: Inverse Different Moment	- Similar to 'angular second moment'/F1
F8: Sum Variance	- Similar to 'variance'/F4
F9: Sum Entropy	- Similar to 'entropy'/F3

K-Nearest Neighbors (KNN)

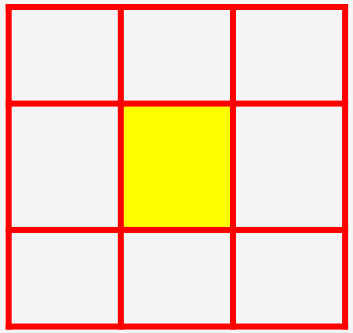
- *Nearest Neighbor (NN)* merupakan algoritme sederhana yang didasarkan pada pembelajaran yang diawasi (*supervised learning*).
- Tujuannya adalah menemukan titik yang paling dekat dengan data uji.
- Ukuran kedekatan antar data diukur menggunakan fungsi jarak *euclidean*
- Misalkan sebuah data x memiliki vektor fitur $\{a_1(x), a_2(x), \ldots, a_n(x)\}$

$$d(x_i, y_i) = \sqrt{\sigma_{r=}^{n} (a_r(x_i) - a_r(y_i))^2}$$

dimana $a_r(x)$ adalah nilai dari atribut ke r dari data x latih dan $a_r(y_i)$ dimana y_i dari data testing. Maka jarak antara dua data x_i dan yi adalah $d(x_i, y_i)$

Langkah menghitung Haralick features

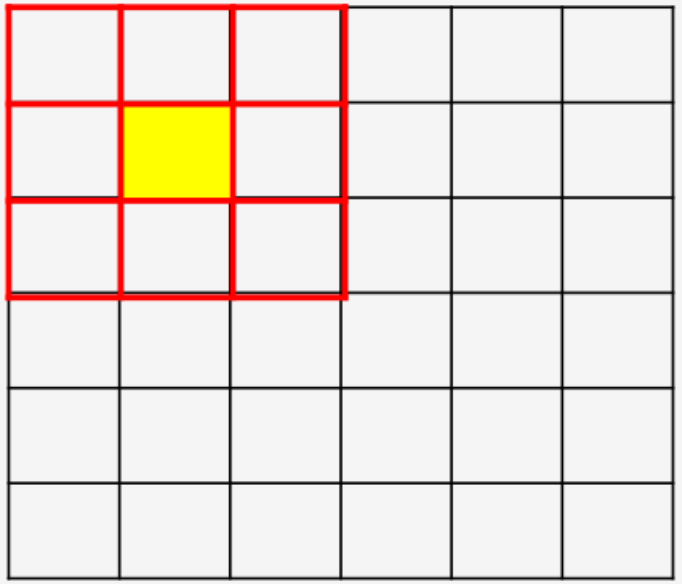
1. Menentukan ukuran window.
   Pada contoh disamping menggunakan window 3 x 3



Window

Langkah menghitung Haralick features

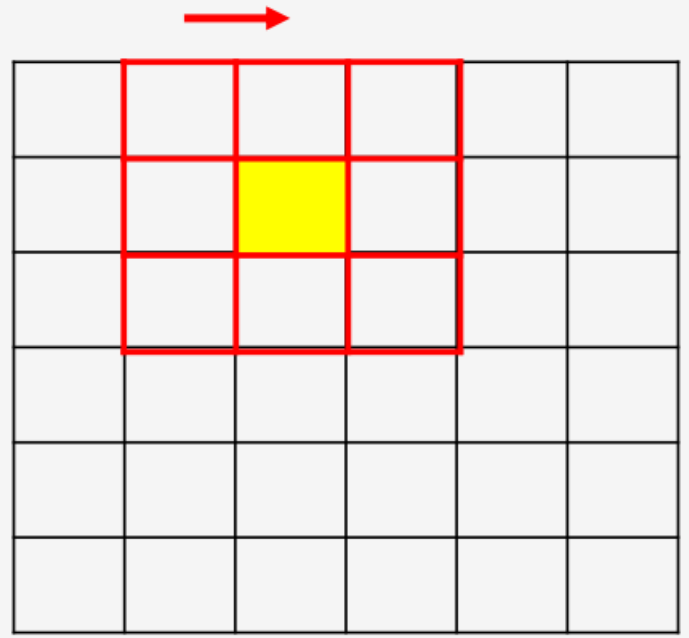
1. Menentukan ukuran window.
   Pada contoh disamping menggunakan window 3 x 3
2. Tempatkan jendela di posisi pertama di kiri atas gambar.
3. Untuk piksel dalam jendela ini di posisi ini hitung GLCM
4. Hitung ukuran tekstur / Haralick fitur



Gambar keseluruhan

Langkah menghitung Haralick features

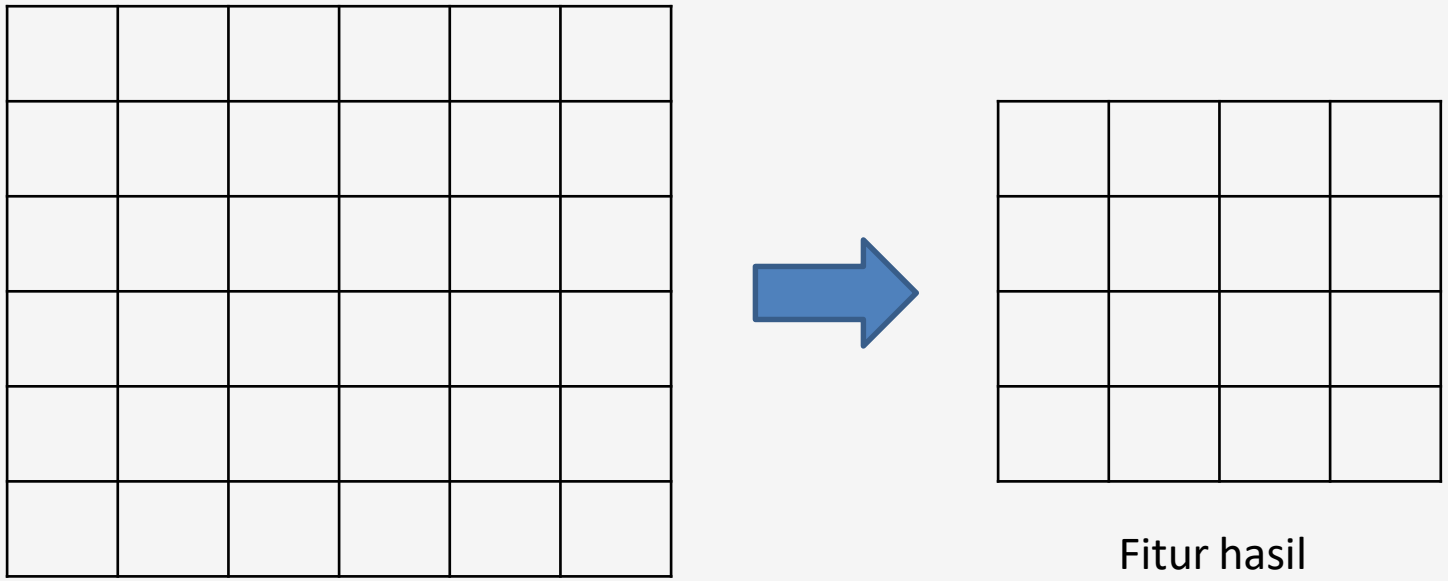
1. Menentukan ukuran window.
   Pada contoh disamping menggunakan window 3 x 3
2. Tempatkan jendela di posisi pertama di kiri atas gambar.
3. Untuk piksel dalam jendela ini di posisi ini hitung GLCM
4. Hitung ukuran tekstur / Haralick fitur
5. Geser jendela sebanyak satu piksel, dan ulangi langkah 3 dan 4.



Gambar keseluruhan

Langkah menghitung Haralick features

6 Lanjutkan langkah diatas hingga window telah mencakup semua posisi yang mungkin sampai gambar tekstur lengkap.



Gambar keseluruhan

Fitur hasil

Haralick feature

Haralick mengekstraksi tiga belas fitur tekstur dari GLCM untuk sebuah gambar. Fitur-fitur ini adalah sebagai berikut:

1. **Angular Second Moment**-Ukuran keseragaman tekstur pada gambar.

$$f_1 = \sum_{i,j=0}^{N-1} (p_{i,j})^2$$

Contoh perhitungan ASM pada GLCM horizontal (0°) yang telah dihitung sebelumnya

$$\sum_{i,j=0}^{N-1} (p_{i,j})^2 = (p_{0,0})^2+(p_{0,1})^2+(p_{0,2})^2+(p_{0,3})^2+(p_{1,0})^2+ \ldots +(p_{3,3})^2$$

$$= 0{,}166^2+0{,}083^2+0{,}042^2+0{,}000^2$$
$$+0{,}083^2+0{,}166^2+0{,}000^2+0{,}000^2$$
$$+0{,}042^2+0{,}000^2+0{,}249^2+0{,}042^2$$
$$+0{,}000^2+0{,}000^2+0{,}042^2+0{,}083^2$$
$$= 0{,}145$$

0,166	0,083	0,042	0
0,083	0,166	0	0
0,042	0	0,250	0,042
0	0	0,042	0,083

Haralick feature

2 Contrast

Ukuran intensitas atau variasi tingkat abu-abu antara referensi pixel dan tetangga.

$$f_2 = \sum_{i,j=0}^{N-1} |i-j|^2 P_{i,j}$$

0,166	0,083	0,042	0
0,083	0,166	0	0
0,042	0	0,250	0,042
0	0	0,042	0,083

Contoh perhitungan ASM pada GLCM horizontal (0°) yang telah dihitung sebelumnya

$$\sum_{i,j=0}^{N-1} |i-j|^2 p_{i,j} = |0-0|^2 p_{0,0} + |0-1|^2 p_{0,1} + |0-2|^2 p_{0,2} + |0-3|^2 p_{0,3} + |1-0|^2 p_{1,0} + \ldots + |3-3|^2 p_{3,3}$$

$$\begin{aligned} &= |0-0|^2 * 0{,}166 + |0-1|^2 * 0{,}083 + |0-2|^2 * 0{,}042 + |0-3|^2 * 0{,}000 \\ &+|1-0|^2 * 0{,}083 + |1-1|^2 * 0{,}166 + |1-2|^2 * 0{,}000 + |1-3|^2 * 0{,}000 \\ &+|2-0|^2 * 0{,}042 + |2-1|^2 * 0{,}000 + |2-2|^2 * 0{,}249 + |2-3|^2 * 0{,}042 \\ &+|3-0|^2 * 0{,}000 + |3-1|^2 * 0{,}000 + |3-2|^2 * 0{,}042 + |3-3|^2 * 0{,}083 \\ &= 0{,}586 \end{aligned}$$

Haralick feature

3 Correlation

Menunjukkan ketergantungan linear tingkat abu-abu nilai-nilai dalam matriks co-occurrence.

$$f_3 = \frac{\sum_{i,j=0}^{N-1}(ij)\, p_{i,j} - \mu_x \mu_y}{\sigma_x \sigma_y}$$

Dimana μ_x, μ_y, σ_x dan σ_y adalah mean dan standar deviasi dari p_x dan p_y

$$\mu_i = \sum_{i,j=0}^{N-1} i(p_{i,j}) \qquad \mu_j = \sum_{i,j=0}^{N-1} j(p_{i,j})$$

$$\sigma_i = \sqrt{\sigma_i{}^2} \qquad \sigma_j = \sqrt{\sigma_j{}^2}$$

Haralick feature

4 Variance

$$f_4 = \sum_{i,j=0}^{N-1} (i-\mu)^2 p_{i,j}$$

Contoh perhitungan ASM pada GLCM horizontal (0°)

yang telah dihitung sebelumnya

0,166	0,083	0,042	0
0,083	0,166	0	0
0,042	0	0,250	0,042
0	0	0,042	0,083

$$\sum_{i,j=0}^{N-1} (1-\mu)^2 p_{i,j} = (0-\mu)^2 p_{0,0} + (0-\mu)^2 p_{0,1} + (0-\mu)^2 p_{0,2} + (0-\mu)^2 p_{0,3} + (1-\mu)^2 p_{1,0} + \ldots + (3-\mu)^2 p_{3,3}$$

$$= (0-1{,}292)^2 * 0{,}166 + (0-1{,}292)^2 * 0{,}083 + (0-1{,}292)^2 * 0{,}042 + (0-1{,}292)^2 * 0{,}000$$
$$+(1-1{,}292)^2 * 0{,}083 + (1-1{,}292)^2 * 0{,}166 + (1-1{,}292)^2 * 0{,}000 + (1-1{,}292)^2 * 0{,}000$$
$$+(2-1{,}292)^2 * 0{,}042 + (2-1{,}292)^2 * 0{,}000 + (2-1{,}292)^2 * 0{,}249 + (2-1{,}292)^2 * 0{,}042$$
$$+(3-1{,}292)^2 * 0{,}000 + (3-1{,}292)^2 * 0{,}000 + (3-1{,}292)^2 * 0{,}042 + (3-1{,}292)^2 * 0{,}083$$
$$= 1{,}039$$

Haralick feature

5 Homogenity (Inverse Difference Moment)

Ukuran seberapa dekat distribusi elemen dalam GLCM dengan diagonal GLCM.

$$f_5 = \sum_{i,j=0}^{N-1} \frac{p_{i,j}}{1+(i-j)^2}$$

0,166	0,083	0,042	0
0,083	0,166	0	0
0,042	0	0,250	0,042
0	0	0,042	0,083

Contoh perhitungan ASM pada GLCM horizontal (0°) yang telah dihitung sebelumnya

$$\sum_{i,j=0}^{N-1} \frac{p_{i,j}}{1+(i-j)^2} = \frac{p_{0,0}}{1+(0-0)^2} + \frac{p_{0,1}}{1+(0-1)^2} + \frac{p_{0,2}}{1+(0-2)^2} + \frac{p_{0,3}}{1+(0-3)^2} + \frac{p_{1,0}}{1+(1-0)^2} + \ldots + \frac{p_{3,3}}{1+(3-3)^2}$$

$$= 0{,}166/1 + 0{,}083/3 + 0{,}042/5 + 0{,}000/10$$
$$+ 0{,}083/3 + 0{,}166/1 + 0{,}000/3 + 0{,}000/5$$
$$+ 0{,}042/5 + 0{,}000/3 + 0{,}250/1 + 0{,}042/3$$
$$+ 0{,}000/10 + 0{,}000/5 + 0{,}042/3 + 0{,}083/1$$
$$= 0{,}765$$

Haralick feature

6 **Sum Average**

$$f_6 = \sum_{i=1}^{2N-1} i p_{x+y}(i)$$

7 **Sum Variance**

$$f_7 = \sum_{i=1}^{2N-1} (i - f_8)^2 \, p_{x+y}(i)$$

8 **Sum Entropy**

$$f_8 = -\sum_{i=1}^{2N-1} p_{x+y}(i) \log\{p_{x+y}(i)\}$$

Dimana

$$p_{x+y}(k) = \sum_{i,j=0}^{N-1} p_{i,j} \quad , i + j = k$$

Haralick feature

9 Entropy

Mengukur tingkatan keacakan nilai pada gambar.

$$f_9 = -\sum_{i,j=0}^{N-1} p_{i,j} \log(p_{i,j})$$

0,166	0,083	0,042	0
0,083	0,166	0	0
0,042	0	0,250	0,042
0	0	0,042	0,083

Contoh perhitungan ASM pada GLCM horizontal (0°) yang telah dihitung sebelumnya

$$-\sum_{i,j=0}^{N-1} p_{i,j} \log(p_{i,j}) = -\{p_{0,0} \log(p_{0,0}) + p_{0,1} \log(p_{0,1}) + p_{0,2} \log(p_{0,2}) + p_{0,3} \log(p_{0,3}) + p_{1,0} \log(p_{1,0}) + \ldots + p_{3,3} \log(p_{3,3})\}$$

$$= -\{0{,}166 \log(0{,}166) + 0{,}083 \log(0{,}083) + 0{,}042 \log(0{,}042) + 0{,}000$$
$$+0{,}083 \log(0{,}083) + 0{,}166 \log(0{,}166) + 0{,}000 + 0{,}000$$
$$+0{,}042 \log(0{,}042) + 0{,}000^2 + 0{,}249 \log(0{,}249) + 0{,}042 \log(0{,}042)$$
$$+0{,}000 + 0{,}000 + 0{,}042 \log(0{,}042) + 0{,}083 \log(0{,}083)\}$$

Haralick feature

10 Difference Variance

$$f_{10} = variance\ of\ p_{x-y}$$

11 Difference Etropy

$$f_{11} = -\sum_{i=0}^{N-1} p_{x-y}(i) \log\{p_{x-y}(i)\}$$

Dimana

$$p_{x-y}(k) = \sum_{i,j=0}^{N-1} p_{i,j} \quad , |i-j| = k$$

Haralick feature

12 Information Measures of Correlatio$_{Dim}$n$_{ana}$Feature 1

$$f_{12} = \frac{f_9 - HXY1}{\max(HX, HY)}$$

$HX = -\sum_i p_x(i) \log(p_x(i))$ = entropy of p_x

$HY = -\sum_j p_y(j) \log(p_y(j))$ = entropy of p_y

$$HXY = -\sum_i \sum_j p(i,j) \log(p(i,j))$$

$$HXY1 = -\sum_i \sum_j p(i,j) \log(p_x(i) p_y(j))$$

$$HXY2 = -\sum_i \sum_j p_x(i) p_y(j) \log(p_x(i) p_y(j))$$

13 Information Measures of Correlation Feature 2

$$f_{13} = [1 - \exp(-2(HXY2 - f_9))]^{1/2}$$

14 Maximal Correlation Coefficient

$$f_{14} = (\text{second largest eigenvalue of } Q)^{1/2}$$

Dimana

$$Q(i,j) = \sum_k \frac{p(i,k)p(j,k)}{p_x(i)p_y(k)}$$

Contoh Python

1
```python
# mengubah gambar menjadi grayscale
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

2
```python
def extract_features(image):
    # menghitung haralic feature
    textures = mt.features.haralick(image)

    # mengambil nilai mean
    ht_mean = textures.mean(axis=0)
    return ht_mean
```

3
```python
# Inisialisasi classifier
print("Melakukan inisialisasi classifier..")
clf_svm = LinearSVC(random_state=9)

# Memasukkan data training ke classifier
print("Memasukkan data/label kedalam model..")
clf_svm.fit(train_features, train_labels)
```

Gambar train

Contoh Python

Nilai haralic feature

```
0.0002, 727.7330, 0.8838, 3131.1616, 0.0917, 323.8067, 11796.9135, 8.4587, 13.5848, 0.0001, 5.4702, -0.1833, 0.9658
0.0001, 975.2848, 0.8660, 3638.8094, 0.0692, 252.5360, 13579.9529, 8.8136, 14.1824, 0.0001, 5.7904, -0.1965, 0.9760
0.0002, 294.7131, 0.9355, 2283.9284, 0.1112, 237.5983, 8841.0006, 8.4744, 13.2272, 0.0002, 4.9269, -0.2343, 0.9842
0.0001, 1609.3866, 0.6460, 2273.1086, 0.0463, 205.9179, 7483.0479, 8.4533, 14.6348, 0.0001, 6.2964, -0.0670, 0.7896
0.0001, 689.7359, 0.8097, 1811.8548, 0.0785, 144.3217, 6557.6834, 8.2769, 13.7503, 0.0001, 5.5985, -0.1261, 0.9128
0.0001, 1042.6606, 0.8168, 2844.9398, 0.0693, 168.6709, 10337.0988, 8.5354, 14.2522, 0.0001, 5.8865, -0.1205, 0.9091
0.0001, 760.7374, 0.8877, 3387.3833, 0.1025, 227.6467, 12788.7960, 8.4958, 13.7575, 0.0001, 5.4472, -0.1799, 0.9636
0.0001, 796.8241, 0.7563, 1634.9789, 0.0752, 183.0240, 5743.0916, 8.2157, 13.7587, 0.0001, 5.6579, -0.1182, 0.8969
0.0942, 430.9297, 0.9029, 2218.2925, 0.6645, 328.9716, 8442.2403, 5.1614, 6.6454, 0.0015, 2.7726, -0.5420, 0.9952
```

4

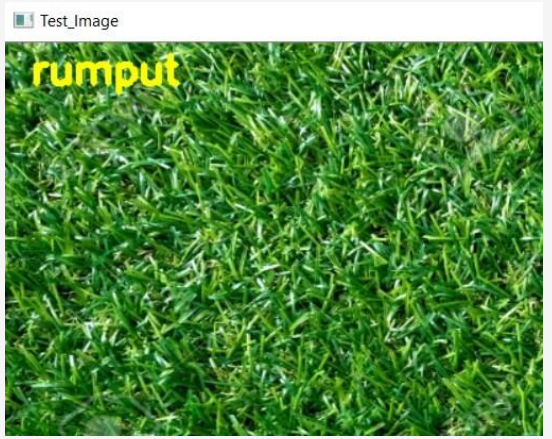
```python
# membaca gambar test
image = cv2.imread(file)

# mengubah gambar menjadi grayscale
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# mendapatkan fitur haralick dari gambar
features = extract_features(gray)

# memprediksi label kelas
prediction = clf_svm.predict(features.reshape(1, -1))[0]
```

Hasil akhir



TUGAS

Implementasikan program Haralick Feature untuk Texture Recognition menggunakan testing image anda sendiri.

https://gogulilango.com/software/texture-recognition