BAB X

EKSTRAKSI CIRI TEKSTUR

Materi:

- Konsep tekstur pada citra digital
- *Local Binary Pattern*

Tujuan Praktikum:

- Mahasiswa dapat memahami konsep tekstur pada citra digital
- Mahasiswa dapat menerapkan *Local Binary Pattern* pada citra digital

A. Penyajian

1. Konsep tekstur pada citra

Salah satu tahapan yang harus dilakukan dalam pengenalan pola citra digital, adalah ekstraksi ciri atau disebut juga fitur. Ciri-ciri umum yang dapat diekstrak dalam sebuah citra digital yakni, ekstraksi ciri bentuk, ekstraksi ciri ukuran dan geometri, ekstraksi ciri tekstur dan ekstraksi ciri warna. Pada modul ini, kita akan mempelajari salah satu metode ekstraksi ciri, yakni ciri tekstur.

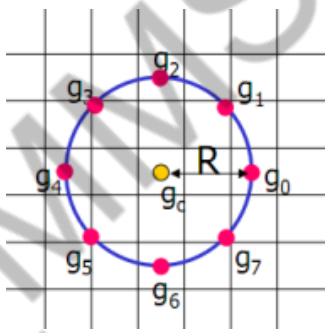
Ciri tekstur merupakan salah satu ciri penting dalam pengenalan pola. Misal anda diminta untuk mendeteksi sebuah bola berwarna merah di lapangan rumput yang berwarna hijau maka dengan mudah anda melakukannya dengan cara menggunakan ciri warna dari bola tersebut. Namun, jika anda diminta untuk mendeteksi seekor macan tutul melewati padang rumput yang kering maka ciri warna tidak cukup untuk dapat mendeteksi macan tutul tersebut dengan akurat. Oleh karena itu, dalam kasus ini, ciri tekstur menjadi penting karena tekstur dari kulit macan tutul yang berbintik berbeda dengan tekstur rumput yang bergaris. Pada modul ini kita akan membahas salah satu metode ekstraksi ciri tekstur pada citra digital yakni *Local Binary Pattern* (LBP).

2. *Local Binary Pattern*

Local Binary Pattern (LBP) merupakan salah satu *texture descriptor* yang sederhana dan efisien dengan cara memberikan label pada piksel dengan melakukan *thresholding* pada setiap piksel tetangga dan mempertimbangkan hasilnya sebagai bilangan biner.

LBP menggunakan citra *grayscale* dengan ketetanggaan 3x3, dimana nilai piksel pada posisi tengah digunakan sebagai ambang batas (*threshold*). Piksel-piksel tetangga kemudian dilakukan binerisasi dengan ketentuan apabila nilai piksel tetangga lebih kecil dibandingkan dengan ambang batas maka akan bernilai 0, sebaliknya akan bernilai 1. Setelah itu, nilai binerisasi pada setiap piksel dikalikan dengan bobot yang telah ditentukan pada piksel yang bersesuaian, yakni dikalikan dengan 2^n, dimana n adalah urutan piksel tetangga yang dimulai dari pojok kiri atas dengan mengitari piksel pada posisi tengah dengan arah dari kiri sampai dengan kanan bawah dengan urutan dimulai dari 0-7. Hasil perkalian pada seluruh piksel tetangga kemudian dilakukan penjumlahan.

LBP merupakan *descriptor* yang efisien yang mendeskripsikan pola tekstur lokal pada citra *grayscale*. LBP didefinisikan sebagai sekumpulan *pixel* ketetanggaan yang tersebar secara melingkar *(circular neighborhoods)* dengan *pixel* pusat berada di tengah seperti ditunjukkan Gambar 1. Notasi g_p merupakan nilai *pixel* tetangga ke-*p*. g_c merupakan *pixel* pusat yang digunakan sebagai nilai *threshold* agar *pixel* ketetanggaannya menjadi kode biner.



Gambar 1. *Circular neighborhood* delapan *sampling points*.

Secara matematis, operator LBP dapat ditulis menggunakan persamaan di bawah ini:

$$LBP_{P,R}(x_c, y_c) = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \qquad (1)$$

$$s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \qquad (2)$$

dimana,

x_c dan y_c = koordinat *pixel* pusat

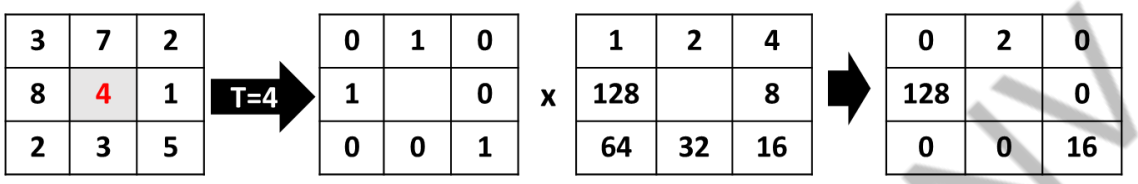
R = radius

P = titik sampling

g_p = nilai derajat keabuan pada piksel tetangga ke-p

g_c = nilai derajat keabuan pada piksel di posisi tengah

Contoh penggunaan LBP pada citra dapat dilihat pada Gambar 2.



Gambar 2. Contoh perhitungan nilai LBP.

Pola : 01001001 $= 0(2^0) + 1(2^1) + 0(2^2) + 0(2^3) + 1(2^4) + 0(2^5) + 0(2^6) + 1(2^7)$

$= 0 + 2 + 0 + 0 + 16 + 0 + 0 + 128 = 146$

Untuk keperluan klasifikasi/ pengenalan objek, nilai-nilai LBP selanjutnya direpresentasikan dalam bentuk histogram. Histogram menunjukkan frekuensi kejadian berbagai nilai LBP. Untuk ukuran citra *N*×*M*, keseluruhan nilai LBP dapat direpresentasikan dengan membentuk histogram sebagai berikut:

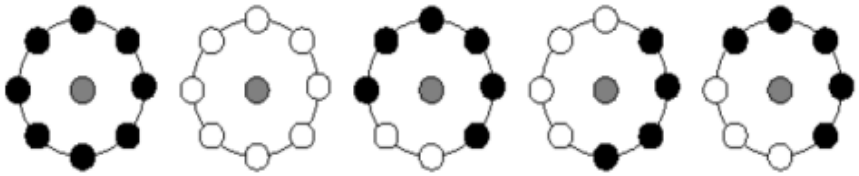
$$H(k) = \sum_{i=0}^{N}\sum_{j=1}^{M} f\big(LBP_{P,R}(i,j),k\big), k \in [0,K] \quad (3)$$

$$f(x,y) = \begin{cases} 1, & x = y \\ 0, & \text{selainnya} \end{cases} \quad (4)$$

dengan *k* merupakan nilai LBP maksimum.

Ojala *et al.* (2002) melakukan observasi bahwa beberapa pola LBP tertentu memiliki informasi penting dari suatu tekstur. Pola-pola yang memiliki informasi penting ini disebut *uniform patterns*. LBP dikatakan *uniform* jika *discontinuities* atau transisi bit 0/1 paling banyak adalah dua. Sebagai contoh 00000000 (0 transisi), 11001111 (2 transisi), dan 11001111 (2 transisi) merupakan *uniform patterns*, sedangkan 11001001 (4 transisi) dan 10101001 (6 transisi) bukan merupakan *uniform patterns*. *Uniform patterns* berfungsi untuk mengidentifikasi noda (*spot*), *flat area*

atau *dark spot*, sudut, dan tepi. Hampir 90 persen dari tekstur merupakan *uniform patterns* [1].



Spot *Spot/flat* *Line end* *Edge* *Corner*

Gambar 3. Tekstur *uniform patterns*.

Gambar 3 menunjukkan makna dari *uniform patterns*. Secara matematis, *uniform patterns* dapat diekspresikan sebagai berikut:

$$U(LBP_{P,R}) = |s(g_{P-1} - g_c) - s(g_0 - g_c)| + \sum_{p=1}^{P-1} |s(g_p - g_c) - s(g_{p-1} - g_c)| \quad (7)$$

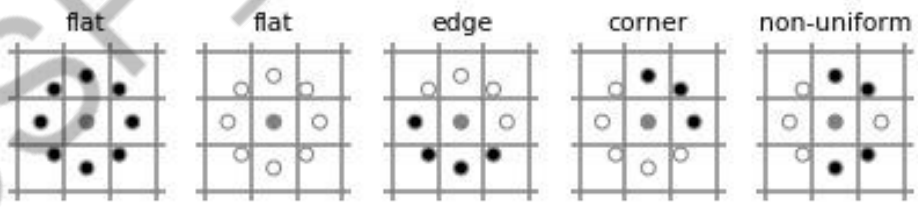
dengan $U(LBP_{P,R})$ merupakan *uniform patterns* dari jumlah *sampling points* (*P*) dan radius (*R*), p adalah *circular sampling point*, g_p adalah nilai keabuan dari p, dan g_c adalah nilai keabuan rata-rata seluruh *pixel neighborhood*. Jumlah pola yang dihasilkan *uniform patterns* adalah $P(P-1) + 2$ *bin*

Sebelum kita mengimplementasikan LBP pada sebuah citra digital, kode program di bawah ini akan menampilkan tekstur uniform pattern yang terdapat dalam LBP [2].

```
import numpy as np
import matplotlib.pyplot as plt

METHOD = 'uniform'
plt.rcParams['font.size'] = 9

def plot_circle(ax, center, radius, color):
    circle = plt.Circle(center, radius, facecolor=color, edgecolor='0.5')
    ax.add_patch(circle)

def plot_lbp_model(ax, binary_values):
    """Draw the schematic for a local binary pattern."""
    # Geometry spec
    theta = np.deg2rad(45)
    R = 1
    r = 0.15
    w = 1.5
    gray = '0.5'

    # Draw the central pixel.
```

```
    plot_circle(ax, (0, 0), radius=r, color=gray)
    # Draw the surrounding pixels.
    for i, facecolor in enumerate(binary_values):
        x = R * np.cos(i * theta)
        y = R * np.sin(i * theta)
        plot_circle(ax, (x, y), radius=r, color=str(facecolor))

    # Draw the pixel grid.
    for x in np.linspace(-w, w, 4):
        ax.axvline(x, color=gray)
        ax.axhline(x, color=gray)

    # Tweak the layout.
    ax.axis('image')
    ax.axis('off')
    size = w + 0.2
    ax.set_xlim(-size, size)
    ax.set_ylim(-size, size)

fig, axes = plt.subplots(ncols=5, figsize=(7, 2))

titles = ['flat', 'flat', 'edge', 'corner', 'non-uniform']

binary_patterns = [np.zeros(8),
                   np.ones(8),
                   np.hstack([np.ones(4), np.zeros(4)]),
                   np.hstack([np.zeros(3), np.ones(5)]),
                   [1, 0, 0, 1, 1, 1, 0, 0]]

for ax, values, name in zip(axes, binary_patterns, titles):
    plot_lbp_model(ax, values)
    ax.set_title(name)
```

Keluaran:



Gambar 4. Keluaran kode program untuk menampilkan tekstur *uniform pattern* LBP

Pada gambar di atas menunjukkan bahwa warna hitam atau putih mewakili pisel yang memiliki intensitas kurang atau lebih dibandingkan piksel pusatnya. Ketika piksel disekitar piksel pusat semuanya berwarna hitam atau putih, maka maknanya

wilayah tersebut datar (tidak ada tekstur pada wilayah tersebut). Sekumpulan piksel hitam atau putih yang muncul berturut-turut dianggap memiliki pola "*uniform*" yang diartikan bahwa terdapat sudut (*corner*) atau tepian (*edge*) pada wilayah tersebut. Namun jika piksel hitam dan putih muncul secara bergantian, maka wilayah tersebut memiliki pola "*non-uniform*".

Untuk implementasi LBP menggunakan python, Anda dapat menggunakan library scikit-image yang mana merupakan library pengolahan citra digital yang bersifat *open source*. Pada library ini terdapat modul/function LBP yang dapat anda gunakan langsung. Berikut definisi function LBP pada library scikit-image:

```
local_binary_pattern(image, P, R, method='default')
```

Terdapat 4 parameter pada function LBP, yakni:

image **:** (N, M) array, citra *grayscale*

P (int) : Titik sampling atau *Circular neighborhood sampling poin*

R (float) : Radius

method{'default', 'ror', 'uniform', 'var'}

Nama Method	Keterangan
'default'	local binary pattern biasa yang tidak bersifat rotation invariant.
'ror'	ekstensi dari LBP biasa yang bersifat rotation invariant.
'uniform'	memperbaiki LBP rotation invariance dengan menambahkan uniform patterns
'nri_uniform'	LBP dengan non rotation-invariant uniform patterns variant

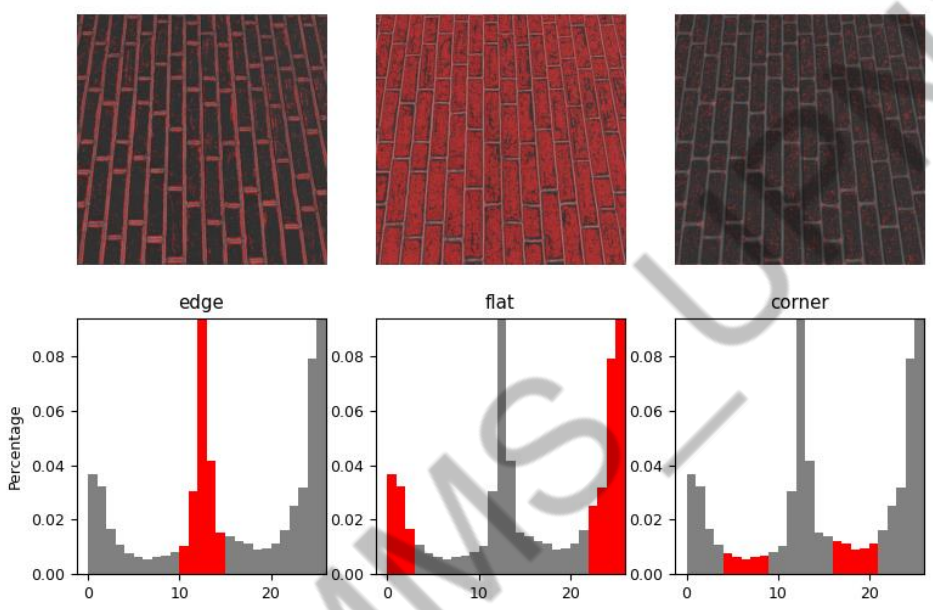
B. Latihan

Berikut contoh penerapan LBP pada citra tekstur bata.

```
from skimage.transform import rotate
from skimage.feature import local_binary_pattern
from skimage import data
from skimage.color import label2rgb

# settings for LBP
radius = 3
n_points = 8 * radius


def overlay_labels(image, lbp, labels):
    mask = np.logical_or.reduce([lbp == each for each in labels])
    return label2rgb(mask, image=image, bg_label=0, alpha=0.5)


def highlight_bars(bars, indexes):
    for i in indexes:
        bars[i].set_facecolor('r')


image = data.brick()
lbp = local_binary_pattern(image, n_points, radius, METHOD)


def hist(ax, lbp):
    n_bins = int(lbp.max() + 1)
    return ax.hist(lbp.ravel(), density=True, bins=n_bins, range=(0, n_bins), facecolor='0.5')


# plot histograms of LBP of textures
fig, (ax_img, ax_hist) = plt.subplots(nrows=2, ncols=3, figsize=(9, 6))
plt.gray()

titles = ('edge', 'flat', 'corner')
w = width = radius - 1
edge_labels = range(n_points // 2 - w, n_points // 2 + w + 1)
flat_labels = list(range(0, w + 1)) + list(range(n_points - w, n_points + 2))
i_14 = n_points // 4           # 1/4th of the histogram
i_34 = 3 * (n_points // 4)     # 3/4th of the histogram
corner_labels = (list(range(i_14 - w, i_14 + w + 1)) +
                 list(range(i_34 - w, i_34 + w + 1)))

label_sets = (edge_labels, flat_labels, corner_labels)

for ax, labels in zip(ax_img, label_sets):
    ax.imshow(overlay_labels(image, lbp, labels))
```

```
for ax, labels, name in zip(ax_hist, label_sets, titles):
    counts, _, bars = hist(ax, lbp)
    highlight_bars(bars, labels)
    ax.set_ylim(top=np.max(counts[:-1]))
    ax.set_xlim(right=n_points + 2)
    ax.set_title(name)

ax_hist[0].set_ylabel('Percentage')
for ax in ax_img:
    ax.axis('off')
```

Keluaran :



Gambar 4. Keluaran kode program pada Latihan No.1.

Penjelasan:

Pada gambar di atas, dengan menggunakan LBP, kita dapat mendeteksi area tepian, area datar dan sudut pada citra tekstur bata (lihat *highligh* merah pada gambar).

C. Lembar Kerja Praktikum

1. Batik merupakan warisan budaya Indonesia yang harus dilindungi. Kita mungkin sering sekali menggunakan batik baik dalam acara formal atau informal, namun kebanyakan dari Kita tidak mengetahui jenis atau motif batik yang kita kenakan. Batik di Indonesia memiliki berbagai macam jenis corak atau pola Batik. Maka, sebagai wujud cinta Tanah Air dan Bela Negara, Mahasiswa Fakultas Ilmu Komputer UPN Veteran Jakarta ingin menciptakan sebuah inovasi dalam bentuk aplikasi untuk mengidentifikasi jenis-jenis batik yang ada di Indonesia berbasis citra digital. Tahap pertama adalah mempelajari jenis-jenis batik yang ada di Indonesia dari berbagai sumber (minimal 2 jenis batik). Selanjutnya, sebelum ke tahap klasifikasi/identifikasi lakukan ekstraksi ciri tekstur dengan LBP menggunakan citra batik yang telah Anda pilih. Lakukan percobaan dengan berbagai macam jenis LBP dan lakukan analisis hasilnya.

Gambar 5. Contoh Motif Batik Kawung dan Sidomukti

Referensi

[1] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, 2002, doi: 10.1109/TPAMI.2002.1017623.

[2] the scikit-image development team, "Local Binary Pattern for texture classification," *https://scikit-image.org/*. https://scikit-image.org/docs/dev/auto_examples/features_detection/plot_local_binary_pattern.html (accessed Nov. 03, 2020).