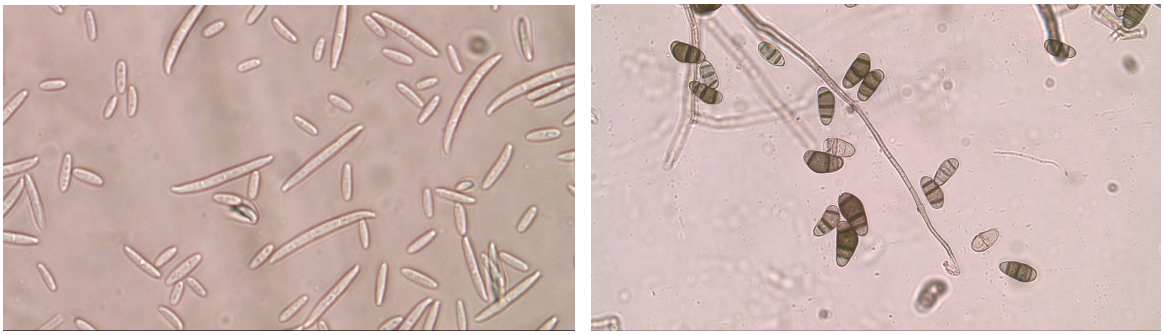
Nama : Fadia Ramadhana
NIM : G64170026

LKP 9
PENGOLAHAN CITRA DIGITAL

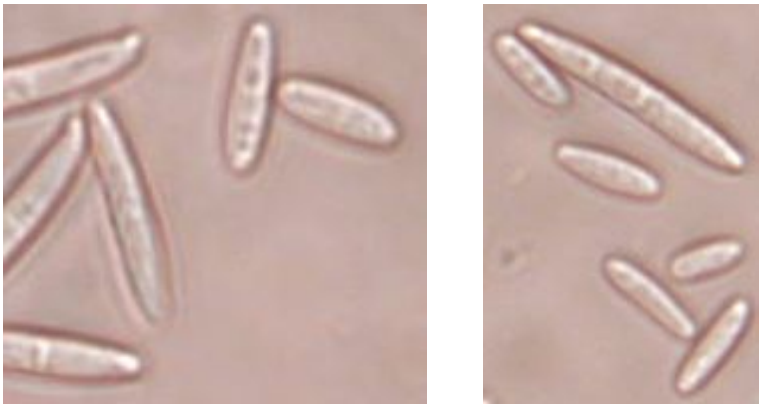


Gambar 1 *Fusarium sp.* (a) dan *Culvularia sp.* (b)

Image Preprocessing

a). Seleksi Citra dan Cropping

Gambar **fusarium-patogen.png** dan **culvularia-patogen.png** akan diseleksi kemudian dilakukan cropping untuk mendapatkan bagian gambar yang dan dapat dilanjutkan pada proses selanjutnya. Beberapa hasil dari seleksi dan cropping adalah sebagai berikut :



Gambar 2 *Fusarium sp.* Crop 1 (a) dan *Fusarium sp.* Crop 2 (b)

Gambar 2 *Culvularia sp.* Crop 1 (a) dan *Culvularia sp.* Crop 2 (b)

Bagian tersebut dipilih karena objek yang akan diproses terlihat lebih jelas dibandingkan beberapa bagian lain pada gambar yang terdapat objek lain atau bayangan yang sebenarnya merupakan objek patogen yang berada di belakang. Bayangan tersebut ada karena kamera/mikroskop hanya difokuskan pada objek yang berada di depannya saja kemudian menimbulkan efek bokeh pada objek. Selain itu, bagian-bagian tersebut dipilih karena mudah untuk diterapkan proses cropping untuk membuang objek yang tidak diperlukan. Gambar 1(b) dan Gambar 2(a) akan digunakan pada proses selanjutnya.

b). Smoothing

Smoothing atau penghalusan gambar dilakukan dengan melakukan konvolusi gambar dengan kernel filter low-pass. Disini, proses smoothing akan dilakukan dengan menggunakan Gaussian Filter dengan kernel 5*5. Gaussian Filter adalah filter linier yang biasanya digunakan untuk mengaburkan gambar atau mengurangi noise. Filter Gaussian sendiri akan mengaburkan tepi dan mengurangi kontras.

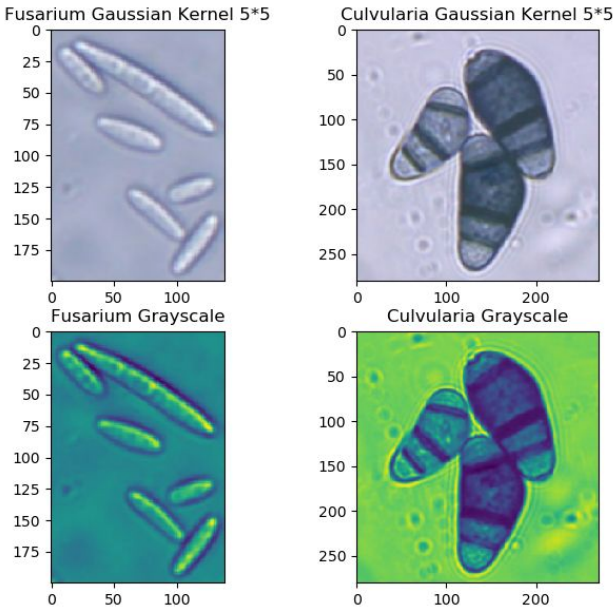
```
# Load image
img1 = cv2.imread("fusarium-patogen-cropped.png")
img2 = cv2.imread("culvularia-patogen-cropped.png")

# Smoothing
blurred_fusarium = cv2.GaussianBlur(img1,(5,5),0)
blurred_culvularia = cv2.GaussianBlur(img2,(5,5),0)
```

c). Grayscale Conversion

Gambar yang telah melalui proses smoothing selanjutnya akan dilakukan proses konversi color space kedalam format grayscale. Gambar hasil dari proses ini diperlukan untuk membuat *mask* pada segmentasi.

```
# Grayscale Convertion
fusarium_gray = cv2.cvtColor(blurred_fusarium, cv2.COLOR_BGR2GRAY)
culvularia_gray = cv2.cvtColor(blurred_culvularia, cv2.COLOR_BGR2GRAY)
```



Gambar 3 hasil smoothing dan grayscale pada gambar *Fusarium sp.* (a) dan hasil smoothing dan grayscale pada gambar *Culvularia sp.* (b)

c). Otsu Thresholding, Binary Inverse Thresholding, dan Segmentasi

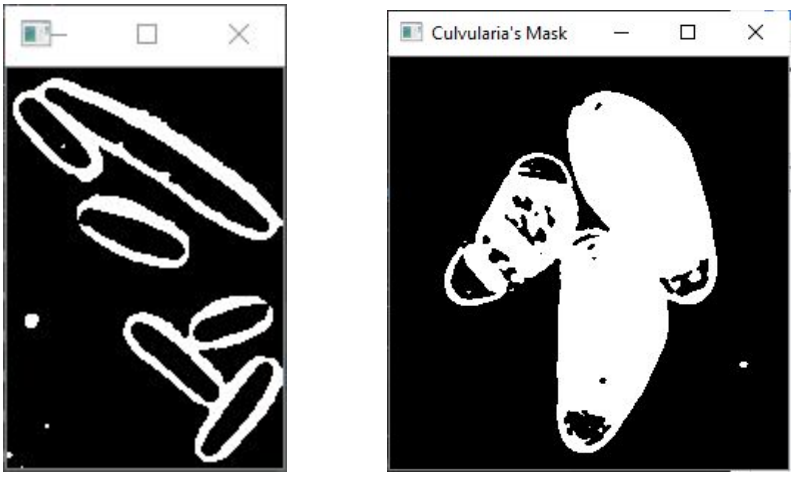
Thresholding adalah metode pemrosesan gambar yang digunakan untuk mengubah gambar *grayscale* (nilai piksel mulai dari 0-255) menjadi gambar biner (nilai piksel hanya dapat memiliki 2 nilai: 0 atau 1). Teknik thresholding terutama digunakan dalam segmentasi. Metode thresholding mengganti setiap piksel dalam gambar dengan piksel hitam jika intensitas piksel kurang dari beberapa T tetap konstan, selain itu diganti dengan piksel putih. Metode Otsu adalah cara threshold adaptif untuk binerisasi dalam pemrosesan gambar. Metode Otsu melakukan segmentasi berdasarkan daerah pemilihan threshold secara otomatis. Sehingga kita tidak perlu melakukan *tune in* nilai threshold secara manual.

$$\begin{aligned}
&\text{Within Class Variance} \quad \sigma_W^2 = W_b\,\sigma_b^2 + W_f\,\sigma_f^2 \quad \text{(as seen above)} \\
&\text{Between Class Variance} \quad \sigma_B^2 = \sigma^2 - \sigma_W^2 \\
&\qquad = W_b(\mu_b - \mu)^2 + W_f(\mu_f - \mu)^2 \quad (\text{where } \mu = W_b\,\mu_b + W_f\,\mu_f) \\
&\qquad = W_b\,W_f\,(\mu_b - \mu_f)^2
\end{aligned}$$

Inverse binary thresholding adalah kebalikan dari binary thresholding. Pixel tujuan di set menjadi nol jika piksel sumber yang sesuai lebih besar dari threshold, dan di set menjadi maxValue jika piksel sumber kurang dari threshold.

```
# Otsu's thresholding with Gaussian filtering
ret1,mask1 = cv2.threshold(fusarium_gray,0,255,cv2.THRESH_OTSU+cv2.THRESH_BINARY_INV)
cv2.imshow("Fusarium's Mask", mask1)
```

```
# Otsu's thresholding with Gaussian filtering
ret2,mask2 = cv2.threshold(culvularia_gray,0,255,cv2.THRESH_OTSU+cv2.THRESH_BINARY_INV)
cv2.imshow("Culvularia's Mask", mask2)
```



Gambar 4 hasil mask thresholding *Fusarium sp.* (a) dan
hasil mask thresholding *Culvularia sp.* (b)

Hasil dari thresholding adalah mask yang dapat dilihat seperti pada gambar 4(a) dan 4(b). Ternyata dari proses thresholding yang dilakukan, hasilnya tidak 100% dapat mengcover patogen. Seperti pada gambar 4(a) *Fusarium sp*. Karena badan *Fusarium sp*. Yang berada pada gambar asli memiliki ciri morfologi tubuh transparan sehingga warnanya sama dengan warna background. Ini menyebabkan ketika dilakukan thresholding, bagian tengah tubuh *Fusarium sp.* Akan ikut diberi nilai 0 (hitam). Kemudian pada gambar 4(b), hasil mask *Culvularia sp.* Hampir dapat mengcover objek yang dimaksud pada image asli walaupun ada beberapa bagian yang tidak terdeteksi.

Mask yang telah diperoleh akan digunakan untuk melakukan segmentasi citra. Mask tersebut akan diletakkan secara overlap pada gambar asli.

```
# Segmentasi Fusarium
segmented_fusarium = cv2.bitwise_and(img1, img1, mask = mask1)
cv2.imshow("Fusarium", segmented_fusarium)
```

```
# Segmentasi Culvularia
segmented_culvularia = cv2.bitwise_and(img2, img2, mask = mask2)
cv2.imshow("Culvularia", segmented_culvularia)
```

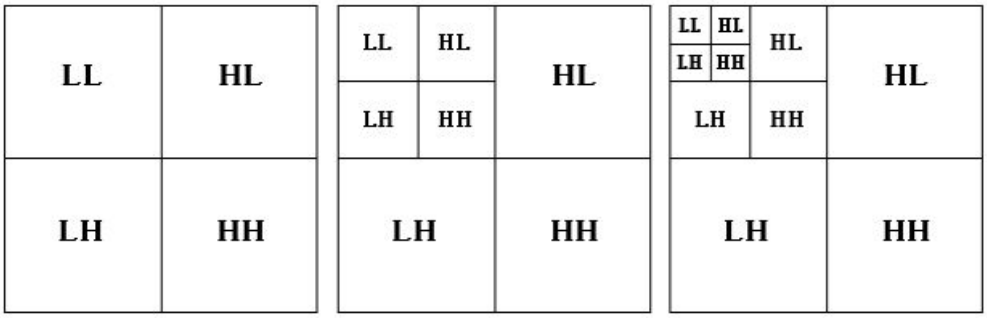
Hasil yang diperoleh dari proses segmentasi adalah sebagai berikut :

Gambar 5 hasil segmentasi *Fusarium sp.* (a) dan
hasil segmentasi *Culvularia sp.* (b)

Ekstraksi Ciri Discrete Wavelet Transform

Gambar yang telah diperoleh dari hasil pra-proses yaitu gambar 5(a) dan 5(b) akan diterapkan proses Discrete Wavelet Transform (DWT) untuk ekstraksi ciri. Sebuah gambar direpresentasikan sebagai array dua dimensi dari koefisien yang masing-masing koefisien tersebut mewakili tingkat kecerahan pada titik tersebut. Koefisien ini secara intuitif dapat dijelaskan seperti variasi kehalusan (*low frequency*) yang menyusun dasar gambar dan variasi ketajaman (*high frequency*) yang memberikan detail tambahan untuk membuat sebuah gambar semakin terperinci. Salah satu cara untuk melakukan penguraian variasi gambar adalah dengan menggunakan Discrete Wavelet Transform (DWT).

Prosedur dari DWT adalah dengan menerapkan *Low Pass Filter* (LPF) dan *High Pass Filter* (HPF) kemudian akan membagi menjadi dua rentang frekuensi. LPF diterapkan pada setiap baris data dan menghasilkan komponen dengan frekuensi rendah yang hanya berisi setengah. Kemudian, begitu juga HPF diterapkan pada setiap baris data dan menghasilkan komponen dengan frekuensi tinggi yang akan mengisi bagian setengahnya lagi. Selanjutnya, pemfilteran dilakukan untuk setiap kolom dari intermediate-data. Array dua dimensi yang dihasilkan berisi empat label (koefisien) data, masing-masing dilabeli sebagai LL (low-low), HL (high-low), LH (low-high) dan HH (high-high). Label LL dapat didekomposisi sekali lagi dengan cara yang sama, sehingga menghasilkan lebih banyak sub-label. Ini dapat dilakukan hingga level berapa pun, sehingga menghasilkan dekomposisi piramidal seperti yang ditunjukkan di bawah ini.



Gambar 6 Dekomposisi piramidal dari sebuah gambar, dekomposisi single-level (a), dekomposisi two-level (b), dan dekomposisi three-level (c)

Seperti disebutkan di atas, koefisien LL pada tingkat tertinggi dapat diklasifikasikan sebagai yang paling penting, dan koefisien 'detail' lainnya dapat diklasifikasikan sebagai kurang penting. Spectrum LH, HL dan HH menunjukan informasi keragaman suatu citra berdasarkan intensitas citra, LH menghasilkan tekstur dan edge dengan orientasi vertikal, HL menghasilkan tekstur dan edge dengan orientasi horizontal dan HH menghasilkan tekstur dan edge dengan orientasi diagonal. Tingkat kepentingan menurun dari puncak piramida ke koefisien yang berada di bawah. Pada python sendiri, untuk melakukan DWT pada image atau dalam hal ini dalam bentuk 2-D dapat dilakukan menggunakan package "**pywt**". Terdapat dua buah fungsi yang dapat digunakan yaitu pywt.dwt2() yang melakukan DWT pada single-level dan pywt.wavedec2() yang melakukan DWT dengan multilevel. Kedua fungsi tersebut sama-sama mengembalikan nilai berupa array dari keempat koefisien yang telah disebutkan diatas.

```
(cA, (cH, cV, cD))
```

```
[cAn, (cHn, cVn, cDn), ..., (cH1, cV1, cD1)]
```

Gambar 7 Nilai return fungsi pywt.dwt2() (a) dan nilai return fungsi pywt.wavedec2() (b).

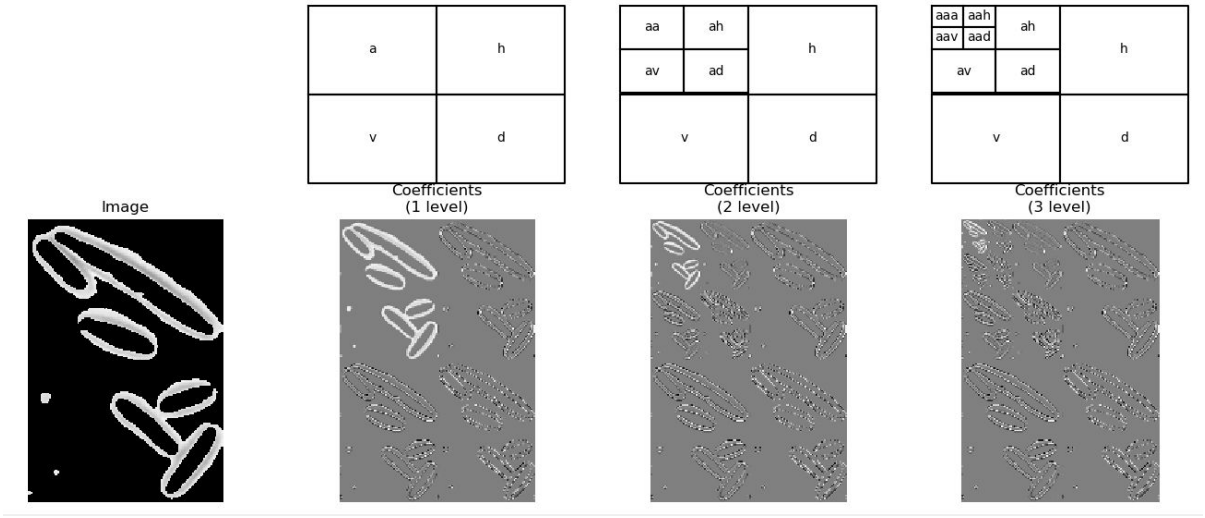
Pada kesempatan ini, akan digunakan fungsi pywt.wavedec2() untuk melakukan dekomposisi DWT dengan three-level.

```
# compute the 2D DWT
c = pywt.wavedec2(image, 'db2', mode='periodization', level=level)
```

```
# Normalisasi masing-masing koefisien pada array secara independen
# untuk hasil yang lebih baik
c[0] /= np.abs(c[0]).max()
for detail_level in range(level):
    c[detail_level + 1] = [d / np.abs(d).max() for d in c[detail_level + 1]]
# menampilkan hasil koefisien yang telah dinormalisasi
arr, slices = pywt.coeffs_to_array(c)
```

Fungsi pywt.coeffs_to_array() sendiri berfungsi untuk memecah masing-masing koefisien pada wavelet menjadi single-array.

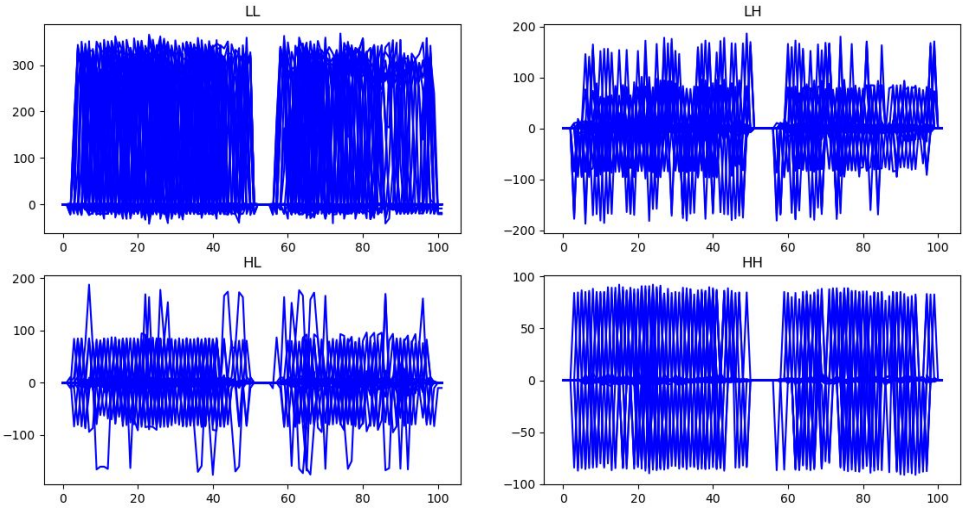
Berikut adalah hasil yang diperoleh menggunakan DWT dengan three-level :



Gambar 8 Hasil Discrete Wavelet Transform dengan berbagai macam level pada *Fusarium sp.*

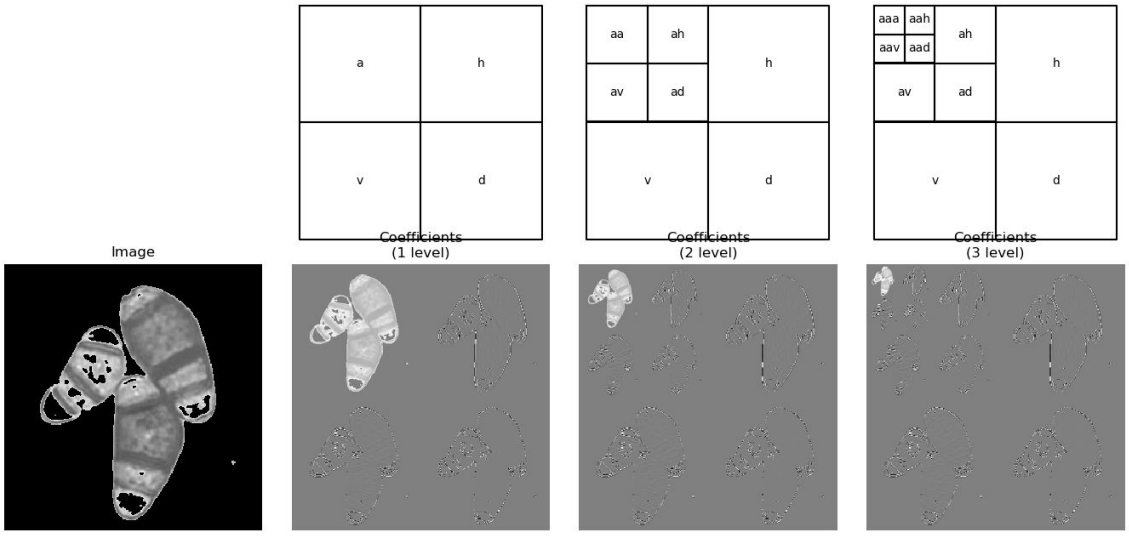
Koefisien LL menunjukan aproksimasi citra asli. Adapun hasil lain dari koefisien LL yaitu dapat mengurangi atau menghilangkan noises kecil pada citra, sehingga koefisien LL menghasilkan nilai suatu citra yang lebih bersih dan mendekati nilai murni suatu citra. Semakin naik level dekomposisi DWT suatu citra maka koefisien LL ukurannya semakin berkurang dan semakin mendekati aproksimasi citra.

Selanjutnya, koefisien-koefisien hasil dari DWT akan ditransformasikan dari yang bentuknya adalah 2-Dimensi menjadi 1-D signal (spectrum). Intensitas citra yang dominan akan ditunjukan dengan nilai peak spectrum yang tinggi pada spectrum LL (aproksimasi).



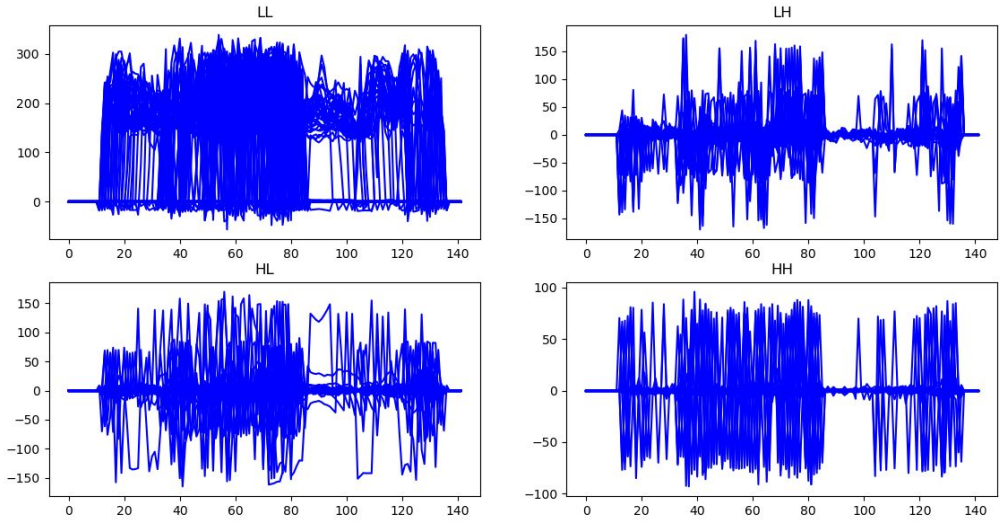
Gambar 9 Hasil transformasi masing-masing koefisien pada DWT *Fusarium sp.* menjadi 1-D signal (Spectrum) dengan tingkat single-level

Sama halnya dengan gambar *Fusarium sp.* , proses DWT three-level yang sama akan diterapkan pada gambar *Culvularia sp.* dan diperoleh hasil sebagai berikut :



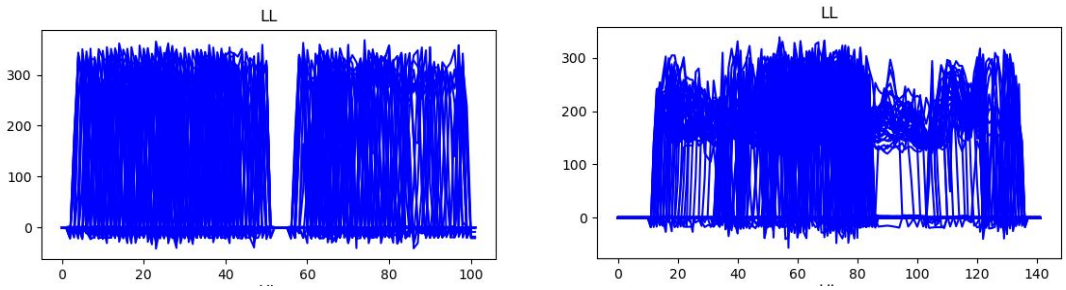
Gambar 10 Hasil Discrete Wavelet Transform (DWT) dengan berbagai macam level pada *Culvularia sp.*

Selanjutnya, koefisien-koefisien hasil dari DWT akan ditransformasikan dari yang bentuknya adalah 2-Dimensi menjadi 1-D signal (spectrum). Intensitas citra yang dominan akan ditunjukan dengan nilai peak spectrum yang tinggi pada spectrum LL (aproksimasi). Berikut adalah hasil spectrum yang diperoleh :



Gambar 11 Hasil transformasi masing-masing koefisien pada DWT *Culvularia sp.* menjadi 1-D signal (Spectrum) dengan tingkat single-level.

Kesimpulan



Gambar 12 Hasil plot koefisien LL pada *Fusarium sp.* (a) dan Hasil plot koefisien LL pada *Culvularia sp.* (b)

Berdasarkan spektrum wavelet yang dihasilkan seperti yang tampak pada Gambar 12(a) dan 12(b) , kedua pathogen ini dapat dibedakan. Kedua spektrum memiliki peak yang sama yaitu 0-300. Namun intensitas frekuensi di tiap posisi dapat dilihat berbeda-beda diantara dua patogen. Keragaman citra yang dominan akan ditunjukan dengan nilai peak spectrum yang tinggi. Peak spectrum LL mengidentifikasikan suatu intensitas yang dominan pada sebuah citra. Analisis menggunakan spectrum akan lebih terlihat ketika sebuah citra didekomposisi ke beberapa level karena dengan naiknya level, tekstur yang tidak dominan akan direduksi sehingga meninggalkan tekstur dominan dengan intensitas frekuensi yang tinggi. Namun, pada kedua hasil spektrum LL dari patogen, dengan hanya menggunakan single-level sudah dapat dilihat perbedaan dari keduanya.

Referensi

Basri, H, Herdiyeni, Y, dan Herliyana, N. (2018). Identifikasi Ciri Citra Mikroskopis Patogen Daun Jabon Menggunakan Discrete Wavelet Transform. Sekolah Pascasarjana. Institut Pertanian Bogor. Bogor (ID): IPB Pr..

http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html

https://link.springer.com/referenceworkentry/10.1007%2F978-0-387-78414-4_305

https://www.debugmode.com/imagecmp/classify.htm
https://pywavelets.readthedocs.io/en/latest/ref/dwt-coefficient-handling.html
https://pywavelets.readthedocs.io/en/0.2.2/ref/2d-dwt-and-idwt.html
https://github.com/nigma/pywt/tree/master/demo