

# **KATA PENGANTAR**

Kami panjatkan puji dan syukur kepada Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya sehingga dapat menyusun proposal ini. Kami juga mengucapkan terima kasih kepada semua pihak yang telah terlibat dalam pembuatan proposal ini.

Kami menyadari bahwa masih memiliki kekurangan dalam pembuatan tugas ini. Oleh Karena itu, kami mengharapkan segala kritik dan saran dari pembaca agar dapat memperbaikinya. Kami harap, tugas ini dapat memberikan manfaat dan dapat berguna bagi pembaca.

Bandung, Maret 2019

Penulis

# **DAFTAR ISI**

[**KATA PENGANTAR** i](#_Toc8114299)

[**DAFTAR ISI** ii](#_Toc8114300)

[**DAFTAR GAMBAR** iii](#_Toc8114301)

[**DAFTAR TABEL** iv](#_Toc8114302)

[**BAB I** 1](#_Toc8114303)

[**PENDAHULUAN** 1](#_Toc8114304)

[**1.1** **Latar Belakang** 1](#_Toc8114305)

[**1.2 Tujuan** 2](#_Toc8114306)

[**1.3 Rumusan Masalah** 2](#_Toc8114308)

[**1.4 Ruang Lingkup Aplikasi** 2](#_Toc8114309)

[**1.5 Tinjauan Pustaka** 2](#_Toc8114310)

[**1.6. Parameter dan Objek Citra yang digunakan** 3](#_Toc8114311)

[**BAB II** 4](#_Toc8114312)

[**TEORI DASAR** 4](#_Toc8114313)

[**2.1 Operasi yang digunakan dalam aplikasi pendeteksian pola daun** 4](#_Toc8114314)

[**2.2 Penjelasan Flowchart atau cara kerja operasi tersebut ditambah dengan pemberian contoh implementasi operasi menggunakan contoh matriks yang dicuplik dengan ukuran kecil** 5](#_Toc8114315)

[**2.3 Coding dan Penjelasan Coding** 6](#_Toc8114316)

[**2.4 Screenshoot dari masing-masing operasi image enhancement** 8](#_Toc8114317)

[**2.5 Analisis hasil operasi berdasarkan visual dan representasi data piksel dari proses pengolahan citra digital** 9](#_Toc8114318)

[**BAB III** 12](#_Toc8114319)

[**METODE PENGERJAAN** 12](#_Toc8114320)

[**3.1 Blok diagram program** 12](#_Toc8114321)

[**3.2 Proses ekstraksi ciri** 12](#_Toc8114322)

[**BAB IV** 13](#_Toc8114323)

[**HASIL DAN PEMBAHASAN** 13](#_Toc8114324)

[**4.1 Fitur yang tersedia** 13](#_Toc8114325)

[**4.2 Proses ekstraksi ciri dan analisanya** 13](#_Toc8114326)

[**4.3 Proses pendeteksian citra dan analisanya** 13](#_Toc8114327)

[**4.4 Perhitungan Proses Secara Manual** 14](#_Toc8114328)

[**4.5 Pengujian dan analisis** 18](#_Toc8114329)

[**KESIMPULAN** 19](#_Toc8114330)

[**DAFTAR PUSTAKA** 20](#_Toc8114331)

# **DAFTAR GAMBAR**

Gambar 1.1 Flowchart 5

Gambar 2.1 LoadImage 8

Gambar 2.2 Grayscale 8

Gambar 2.3 Brightness 8

Gambar 2.4 Sobel 8

Gambar 2.5 Template Matching 8

Gambar 2.6 Citra asli dan hasil 9

# **DAFTAR TABEL**

Tabel 1.1 Coding dan penjelasan 6

Tabel 2.1 Screenshoot Operasi Image 7

## **BAB I**

## **PENDAHULUAN**

### **Latar Belakang**

Tanaman merupakan bagian paling penting dari kehidupan di muka bumi. Tanaman bermanfaat sebagai pemasok oksigen untuk bernafas, sebagai bahan makanan, bahan bakar, obat-obatan, kosmetik dan lebih banyak lagi. Proses pengelompokkan tanaman dapat dilakukan dengan cara mengidentifikasi citra bentuk daun dari tanaman itu sendiri.

Daun merupakan organ tumbuhan yang paling penting, karena memiliki fungsi untuk menangkap cahaya matahari dan mengubahnya menjadi makanan melalui proses fotosintesis. Daun memiliki bentuk yang sangat bervariasi, tetapi dapat dibedakan melalui helai daun *(blade)* dan tangkainya *(petiola).* Maka dari itu, jenis daun dapat dibedakan berdasarkan bentuknya. Cara pengambilan gambar daun dari tanaman tersebut, maka dapat dilakukan langkah-langkah pengenalan pola daun dengan cara mengenali karakteristik struktural daun seperti bentuk dan tekstur daun tersebut. Struktur tulang daun sebagai salah satu fitur yang dapat membedakan daun dari berbagai jenis tanaman, karena tulang daun memiliki fitur yang unik pada setiap jenis tanaman.

Proses pengenalan pola daun dapat dilakukan dengan mengenali karakteristik struktural daun seperti bentuk dan tekstur sebuah daun. Metode untuk melakukan pemrosesan terhadap citra masukan dengan pemanfaatan teknik pengolahan citra digital dilakukan untuk menganalisa karakteristik struktural daun. Perkembangan teknologi untuk teknik pengolahan citra juga berkembang pesat. Berbagai teknik dikembangkan untuk mempermudah pekerjaan manusia, baik sebagai pengolah citra, analis citra maupun penggguna citra untuk berbagai tujuan dan keperluan. Seringkali citra yang digunakan tidak dalam kondisi yang ideal untuk dikaji dikarenakan banyaknya gangguan, dapat berupa bayangan, foto atau gambar kabur, kurang jelasnya kenampakan obyek sehingga dapat menimbulkan masalah dan mempengaruhi hasil interpelasi serta akan mempengaruhi analisa dan perencanaan yang akan dilakukan, maka diperlukan berbagai teknik pengolahan citra untuk memperoleh citra yang ideal. Salah satu teknik pengolahan citra yang biasa dilakukan adalah deteksi tepi.

Deteksi tepi merupakan proses penggalian informasi tepi dari sebuah gambar. Hal ini dianggap sebagai langkah dasar yang digunakan dalam sebagian besar aplikasi pengolahan citra. Tepi dalam sebuah gambar dapat dianggap sebagai batas antara dua daerah yang berbeda. Banyak pendekatan telah digunakan untuk melakukan deteksi tepi pada sebuah gambar. Beberapa metode yang umum digunakan adalah prewitt, roberts, dan sobel. Dalam Pengolahan Citra Digital (PCD) bentuk daun dapat dideteksi dengan melakukan ekstraksi ciri yang terdapat pada citra daun untuk mendapatkan ekstraksi ciri dapat dilakukan menggunakan deteksi tepi.

### **1.2 Tujuan**

## Berdasarkan latar belakang yang sudah dijelaskan, penelitian ini memiliki tujuan yaitu sebagai berikut:

1. Mengetahui proses pada saat mendeteksi pola daun menggunakan deteksi tepi.
2. Mengetahui cara menggunakan metode Template matching.
3. Cara mengetahui bentuk daun.

### **1.3** **Rumusan Masalah**

Dalam pendeteksian pola daun terdapat beberapa masalah yang ditemui yaitu sebagai berikut:

1. Bagaimana proses pada saat mendeteksi pola daun menggunakan deteksi tepi?
2. Bagaimana cara menggunakan metode Template matching?
3. Bagaimana cara mengetahui bentuk daun?

### **1.4 Ruang Lingkup Aplikasi**

Dalam pendeteksian yang akan dilakukan, dibatasi oleh ruang lingkup yang terdiri dari:

* 1. Aplikasi hanya berjalan pada perangkat desktop.
  2. Aplikasi berjalan secara offline.
  3. Implementasi deteksi tepi untuk mendapatkan ekstraksi ciri.

### **1.5 Tinjauan Pustaka**

Adapun tinjauan pustaka yang akan digunakan sebagai referensi dari aplikasi ini adalah sebagai berikut:

(Liantoni, 2015)**,** membubat penelitian dengan judul **Deteksi Tepi Citra Daun Mangga Menggunakan Algoritma Ant Colony Optimization,** dimana deteksi citra daundilakukan dengan deteksi tepi citra, lalu digunakan perhitungan ACO *(Ant Colony* *Optimization)* untuk melakukan pendekatan dengan dataset yang ada.

(Febri Liantoni, 2015) membuat penelitian dengan judul **Klasifikasi Daun Herbal Menggunakan Metode Naïve Bayes Classifier Dan K-Nearest Neighbor,** dimana untuk menghitung pendekatan dengan dataset yang ada, digunakan metode *Naïve Bayes Classifier* karena merupakan metode *Bayes Learning* yang paling cepat dan sederhana dengan hasil akurasi 75%, sedangkan *K-Nearest Neighbor (KNN)* melakukan klasifikasi data berdasarkan jarak terdekat diantara dataset, dengan hasil akurasi 70,83%, hal ini menunjukan bahwa, motede *Naïve Bayes Classifier* lebih baik digunakan dibandingkan dengan *KNN.*

(Setiyorini, 2017)**,** membuat penelitian dengan judul **Perbaikan Kualitas Citra Untuk Klasifikasi Daun Menggunakan *Metode Fuzzy K-Nearest Neighbor*,** darihasil pengujian yang telah dilakukan, Hasil klasifikasi menggunakan metode *Fuzzy K-Nearest Neighbor (KNN)* diperoleh nilai akurasi sebesar 93%, hal ini menunjukkan metodeyang diajukan mampu melakukkan klasifikasi daun dengan baik.

(Lestari, 2018)**,** membuat penelitian dengan judul **Deteksi Penyakit**

**Tanaman Jeruk Siam Berdasarkan Citra Daun Menggunakan Segmentasi Warna RGB-HSV,** dalam sistem menggunakan segmentasi warna guna mendapatkan hasil yangoptimal dalam diagnosis penyakit tanaman jeruk. Hasil dari segmentasi warna menggunakan dataset daun jeruk yang terkena penyakit adalah sebesar 78%.

(Anggoro, 2018)**,** membuat penelitian dengan judul **Implementasi Metode**

**Fuzzy K-Nearest Neighbor Untuk Klasifikasi Penyakit Tanaman Kedelai Pada Citra Daun,** Terdapat empat pengujian yaitu pengujian perbandingan data latih dan data ujidengan akurasi tertingi pada perbandingan 90:10 dengan jumlah 54 data latih dan 6 data uji sebesar 100%, pengujian terhadap nilai Threshold dengan T=10 menghasilkan akurasi 83,33%, pengujian terhadap nilai k=5 menghasilkan akurasi 83,33%, dan pengujian terhadap nilai m=2 dengan akurasi 83,33%.

Aplikasi ini memiliki dasar teori berdasarkan tinjauan pustaka yang telah dijabarkan untuk mempelajari bagaimana ekstraksi bentuk daun menggunakan metode *(KNN) K-Nearest Neighbor.*

### **1.6. Parameter dan Objek Citra yang digunakan**

Parameter yang digunakan pada aplikasi ini adalah bentuk daun, tepi daun, dan tulang daun. Sedangkan objek citra yang digunakan adalah citra daun.

## **BAB II**

## **TEORI DASAR**

### **2.1 Operasi yang digunakan dalam aplikasi pendeteksian pola daun**

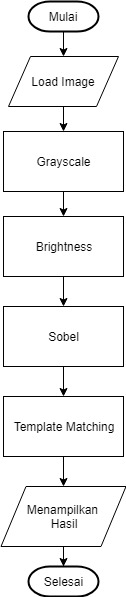
* Nama Operasi

1. GrayScale
2. Brightness atau Pencerahan Citra
3. Sobel
4. Template matching

* Tujuan Operasi

1. Operasi grayscale bertujuan untuk mengubah citra asli atau citra RGB ke dalam citra keabuan
2. Operasi brightness bertujuan untuk mengatur tingkat kecerahan citra yang sebelumnya sudah dilakukan operasi grayscale
3. Operasi sobel bertujuan untuk mengurangi noise sebelum melakukan perhitungan deteksi tepi
4. Operasi template Template bertujuan untuk menemukan bagian-bagian kecil dari gambar yang cocok dengan template gambar.

### **2.2 Penjelasan Flowchart atau cara kerja operasi tersebut ditambah dengan pemberian contoh implementasi operasi menggunakan contoh matriks yang dicuplik dengan ukuran kecil**



Penjelasan:

1. Mulai
2. Input Image atau load image
3. Proses mengubah citra RGB menjadi citra keabuan
4. Proses mengatur tingkat kecerahan citra atau brightnes
5. Proses untuk mendeteksi tepi menggunakan operasi Sobel
6. Proses dilakukannya template matching
7. Menampilkan citra gambar yang sudah di proses oleh operasi-operasi yang sudah dilakukan
8. Selesai

Gambar 1.1 Flowchart

### **2.3 Coding dan Penjelasan Coding**

Table 1.1 Coding dan Penjelasan

|  |  |
| --- | --- |
| @pyqtSlot() def loadClicked(self):  flname, filter = QFileDialog.getOpenFileName(self, 'Open File', 'C:\\Users\\MSI\\PycharmProjects\\ProjekAkhir\\',  "Image Files(\*.jpg)")  if flname:  self.loadImage(flname)  else:  print('Invalid Image') | Penjelasan:  Koding ini digunakan untuk membuat prosedur load clicked yang memuat gambar yang telah diimport |
| def loadImage(self, flname): #C:\Users\MSI\PycharmProjects\ProjekAkhir  self.image = cv2.imread(flname)  self.displayImage() | Penjelasan:  Koding ini digunakan untuk membuat prosedur load image atau memuat file image, membaca file gambar maupun memuat display imagenya. |
| img = cv2.cvtColor(images, cv2.COLOR\_BGR2GRAY) cv2.imshow("Image", img) cv2.waitKey(0) cv2.destroyAllWindows() | Penjelasan:  Koding ini digunakan untuk mengubah citra RGB menjadi citra keabuan agar nilai menjadi satu. |
| brightness = 100 h, w = img.shape[:2] for i in np.arange(h):  for j in np.arange(w):  a = img.item(i, j)  b = a + brightness  if b > 255:  b = 255  elif b < 0:  b = 0  else:  b = b  img.itemset((i, j), b)  cv2.imshow("Image", img) cv2.waitKey(0) cv2.destroyAllWindows() | Penjelasan:  Koding ini digunakan untuk mengatur tingkat kecerahan citra yang sebelumnya sudah dilakukan operasi grayscale. |
| delta = 0 scale = 1 ddepth = cv2.CV\_16S  # Gradient-X grad\_x = cv2.Sobel(img, ddepth, 1, 0, ksize=3, scale=scale, delta=delta, borderType=cv2.BORDER\_DEFAULT) # grad\_x = cv2.Scharr(gray,ddepth,1,0) # Gradient-Y grad\_y = cv2.Sobel(img, ddepth, 0, 1, ksize=3, scale=scale, delta=delta, borderType=cv2.BORDER\_DEFAULT) # grad\_y = cv2.Scharr(gray,ddepth,0,1)  abs\_grad\_x = cv2.convertScaleAbs(grad\_x) # converting back to uint8 abs\_grad\_y = cv2.convertScaleAbs(grad\_y)  dst = cv2.addWeighted(abs\_grad\_x, 0.5, abs\_grad\_y, 0.5, 0) # dst = cv2.add(abs\_grad\_x,abs\_grad\_y)  cv2.imshow('dst', dst) cv2.waitKey(0) cv2.destroyAllWindows()  self.image = dst self.displayImage(2) | Penjelasan:  Koding ini digunakan untuk membuat atau menampilkan bentuk tepi dari suatu citra |
| template\_data.append(dst)  cv2.imshow("Template", dst)  result = cv2.matchTemplate(dst, tmp, cv2.TM\_CCOEFF\_NORMED)  (\_, maxVal, \_, maxLoc) = cv2.minMaxLoc(result)  # Jika ketemu dengan variable berskala baru, maka simpan  if found is None or maxVal > found[0]:  found = (maxVal, maxLoc, r)  if maxVal >= 0.4:  rep = "daun"  imageP1 = imageP  imageP1 = imageP1.replace(rep, "")  imageP1 = imageP1.replace("\\", " ")  imageP1 = imageP1.replace(".jpg", "")  print("ini adalah", imageP1, "") | Penjelasan:  Koding disamping digunakan untuk melakukan proses template matching |

### **2.4 Screenshoot dari masing-masing operasi image enhancement**

Table 2.1 Screenshoot Operasi Image

|  |  |
| --- | --- |
| Gambar 2.1 Load Image Citra Asli | Ini adalah proses meload image |
| Gambar 2.2 Citra Grayscale | Gambar disampimg merupakan hasil dari operasi grayscale |
| Gambar 2.3 Citra Brightness | Gambar disamping merupakan hasil gambar dari citra brightness yang sudah dilakukan operasi grayscale sebelumnya. |
| Gambar 2.4 Citra Sobel | Gambar disamping merupakan gambar hasil dari citra sobel yang menghasilkan terlihat tepi dan structural daun tersebut. |
| Gambar 2.5 Citra Template Matching | Gambar disamping merupakan citra hasil dari template matching. |
| Gambar 2.6 Citra Asli dan Citra Hasil | Ini adalah citra sebelum dilakukan proses atau citra asli dan citra yang sudah di lakukan proses atau citra hasil. |

### **2.5 Analisis hasil operasi berdasarkan visual dan representasi data piksel dari proses pengolahan citra digital**

Analisis Visual :

Pada hasil implementasi diatas, dapat dilihat bahwa citra daun yang semula merupakan citra RGB sudah berubah menjadi citra dengan derajat keabuan.

Analisis Representasi data :

* Citra Asli

[[[22 61 40]

[23 62 41]

[23 64 43]

...

[28 59 44]

[28 59 44]

[31 60 45]]

[[23 62 40]

[25 64 42]

[28 67 45]

...

[28 59 44]

[28 59 44]

[31 60 45]]

[[22 62 37]

[25 65 40]

[30 69 44]

...

[28 59 44]

[28 59 44]

[31 60 45]]

...

[[12 51 26]

[14 53 28]

[16 55 30]

...

[21 57 40]

[18 54 37]

[14 50 33]]

[[10 50 25]

[13 52 27]

[16 55 30]

...

[21 57 40]

[18 54 37]

[14 50 33]]

[[15 54 32]

[18 56 34]

[17 55 33]

...

[ 9 45 28]

[11 47 30]

[19 55 38]]]

Perhitungan manual citra grayscale pada koordinat (0,0)

(40\*0.299)+(61\*0.587)+(22\*0.114) = 50

* Citra Grayscale

[[50 51 53 ... 51 51 52]

[51 53 56 ... 51 51 52]

[50 53 57 ... 51 51 52]

...

[39 41 43 ... 48 45 41]

[38 40 43 ... 48 45 41]

[43 45 44 ... 36 38 46]]

Brightness = ditambah 100

50+100 = 150

51+100 = 151

53+100 = 153

* Citra Brightness

[[150 151 153 ... 151 151 152]

[151 153 156 ... 151 151 152]

[150 153 157 ... 151 151 152]

...

[139 141 143 ... 148 145 141]

[138 140 143 ... 148 145 141]

[143 145 144 ... 136 138 146]]

* Citra Sobel

[[ 0 8 9 ... 0 2 0]

[ 2 14 17 ... 0 2 0]

[ 1 12 11 ... 2 3 1]

...

[ 3 10 8 ... 4 14 0]

[ 8 14 6 ... 23 16 2]

[ 0 6 1 ... 5 3 0]]

Pada hasil proses ini terlihat dari citra asli yang tadinya terdiri dari enam layer matrik diubah menjadi satu layer matriks dengan preprocessing grayscale,brightness dan sobel.

# **BAB III**

# **METODE PENGERJAAN**

## **3.1 Blok diagram program**



Gambar 3.1

Inputan citra yang akan diolah ini berupa citra yang objek citranya merupakan sebuah daun mangga yang masih muda berwarna hijau. Kemudian dilakukan proses grayscale agar nilai tiap pikselnya bernilai satu. Proses selanjutnya yaitu dilakukan proses brightness agar kecerahan citranya dapat diatur. Kemudian dilakukan proses sobel yang berfungsi untuk mendeteksi tepi daun yang sudah dilakukan beberapa proses sebelumnya. Setelah itu, dilakukan proses template matching agar menemukan bagian-bagian kecil dari gambar yang cocok dengan template gambar tersebut. Setelah melakukan seluruh proses tersebut, maka menghasilkan citra dengan tepi daun yang jelas dengan memberikan keterangan nama gambar jenis daun yang diinput.

## **3.2 Proses ekstraksi ciri**

Proses ekstraksi ciri yang digunakan ialah menggunakan metode sobel yang berfungsi mendeteksi tepi objek daun agar mendapatkan ekstraksi ciri pada daun.

# **BAB IV**

# **HASIL DAN PEMBAHASAN**

## **4.1 Fitur yang tersedia**

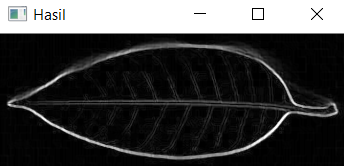
Fitur yang tersedia dalam aplikasi ini adalah menmasukan citra dan menampilkan hasil citra

telah diolah.

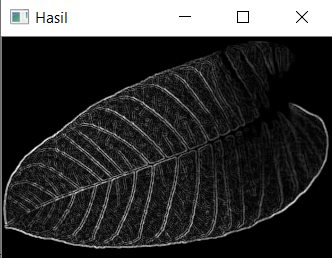
## **4.2 Proses ekstraksi ciri dan analisanya**

Proses ekstraksi ciri yang digunakan ialah menggunakan metode sobel yang berfungsi untuk:

* Mendeteksi tepi objek daun
* Mendapatkan ekstraksi ciri pada daun.



Gambar 4.1 Contoh citra daun mangga



Gambar 4.2 Contoh citra daun jambu

Berdasarkan hasil citra diatas diketahui bahwa setelah dilakukan metode sobel, citra tersebut terlihat jelas tepian-tepian dari objek daun. Pada contoh citra daun jambu terlihat lebih jelas tepiaan-tepian daunnya dibandingkan dengan citra daun mangga, hal ini disebabkan oleh structural dari daun tersebut.

## **4.3 Proses pendeteksian citra dan analisanya**

Proses pendeteksian citra dilakukan dengan menentukan tepi setiap daun. Alasan menggunakan metode sobel sebagai proses akhir yaitu untuk mendapatkan hasil deteksi tepi yang diinginkan. Setelah menggunakan deteksi tepi sobel, terlihat sangat jelas tepi, bentuk dan structural daun tersebut.

## **4.4 Perhitungan Proses Secara Manual**

Perhitungan Grayscale:

Contoh perhitungan pada citra daun mangga

* Citra Asli

B G R

[[[22 61 40]

[23 62 41]

[23 64 43]

...

[28 59 44]

[28 59 44]

[31 60 45]]

[[23 62 40]

[25 64 42]

[28 67 45]

...

[28 59 44]

[28 59 44]

[31 60 45]]

[[22 62 37]

[25 65 40]

[30 69 44]

...

[28 59 44]

[28 59 44]

[31 60 45]]

...

[[12 51 26]

[14 53 28]

[16 55 30]

...

[21 57 40]

[18 54 37]

[14 50 33]]

[[10 50 25]

[13 52 27]

[16 55 30]

...

[21 57 40]

[18 54 37]

[14 50 33]]

[[15 54 32]

[18 56 34]

[17 55 33]

...

[ 9 45 28]

[11 47 30]

[19 55 38]]]

* Perhitungannya manual citra grayscale:

(40\*0.299)+(61\*0.587)+(22\*0.114) = 50 (44\*0.299)+(59\*0.587)+(28\*0.114) = 51

(41\*0.299)+(62\*0.587)+(23\*0.114) = 51 (44\*0.299)+(59\*0.587)+(28\*0.114) = 51

(43\*0.299)+(64\*0.587)+(23\*0.114) = 53 (45\*0.299)+(60\*0.587)+(31\*0.114) = 52

(40\*0.299)+(62\*0.587)+(23\*0.114) = 51 (44\*0.299)+(59\*0.587)+(28\*0.114) = 51

(42\*0.299)+(64\*0.587)+(25\*0.114) = 53 (44\*0.299)+(59\*0.587)+(28\*0.114) = 51

(45\*0.299)+(67\*0.587)+(28\*0.114) = 56 (45\*0.299)+(60\*0.587)+(31\*0.114) = 52

(37\*0.299)+(62\*0.587)+(22\*0.114) = 50 (44\*0.299)+(59\*0.587)+(28\*0.114) = 51

(40\*0.299)+(65\*0.587)+(25\*0.114) = 53 (44\*0.299)+(59\*0.587)+(28\*0.114) = 51

(44\*0.299)+(69\*0.587)+(30\*0.114) = 57 (45\*0.299)+(60\*0.587)+(31\*0.114) = 52

(26\*0.299)+(51\*0.587)+(12\*0.114) = 39 (40\*0.299)+(57\*0.587)+(21\*0.114) = 48

(28\*0.299)+(53\*0.587)+(14\*0.114) = 41 (37\*0.299)+(54\*0.587)+(18\*0.114) = 45

(30\*0.299)+(55\*0.587)+(16\*0.114) = 43 (33\*0.299)+(50\*0.587)+(14\*0.114) = 41

(25\*0.299)+(50\*0.587)+(10\*0.114) = 38 (40\*0.299)+(57\*0.587)+(21\*0.114) = 48

(27\*0.299)+(52\*0.587)+(13\*0.114) = 40 (37\*0.299)+(54\*0.587)+(18\*0.114) = 45

(30\*0.299)+(55\*0.587)+(16\*0.114) = 43 (33\*0.299)+(50\*0.587)+(14\*0.114) = 41

(32\*0.299)+(54\*0.587)+(15\*0.114) = 43 (28\*0.299)+(45\*0.587)+(9\*0.114) = 36

(34\*0.299)+(56\*0.587)+(18\*0.114) = 45 (30\*0.299)+(47\*0.587)+(11\*0.114) = 38

(33\*0.299)+(55\*0.587)+(17\*0.114) = 44 (38\*0.299)+(55\*0.587)+(19\*0.114) = 46

* Hasil Citra Grayscale

[[50 51 53 ... 51 51 52]

[51 53 56 ... 51 51 52]

[50 53 57 ... 51 51 52]

...

[39 41 43 ... 48 45 41]

[38 40 43 ... 48 45 41]

[43 45 44 ... 36 38 46]]

Perhitungan Brightness:

Diketahui brightness = 100

* Citra Grayscale

[[50 51 53 ... 51 51 52]

[51 53 56 ... 51 51 52]

[50 53 57 ... 51 51 52]

...

[39 41 43 ... 48 45 41]

[38 40 43 ... 48 45 41]

[43 45 44 ... 36 38 46]]

* Proses perhitungan citra brightness

Hasil grayscale+100 = hasil brightness

[[50+100 51+100 53+100 ... 51+100 51+100 52+100]

[51+100 53+100 56+100 ... 51+100 51+100 52+100]

[50+100 53+100 57+100 ... 51+100 51+100 52+100]

...

[39+100 41+100 43+100 ... 48+100 45+100 41+100]

[38+100 40+100 43+100 ... 48+100 45+100 41+100]

[43+100 45+100 44+100 ... 36+100 38+100 46+100]]

* Hasil citra brightness

[[150 151 153 ... 151 151 152]

[151 153 156 ... 151 151 152]

[150 153 157 ... 151 151 152]

...

[139 141 143 ... 148 145 141]

[138 140 143 ... 148 145 141]

[143 145 144 ... 136 138 146]]

Perhitungan sobel:

* Citra brightness

[[150 151 153 ... 151 151 152]

[151 153 156 ... 151 151 152]

[150 153 157 ... 151 151 152]

...

[139 141 143 ... 148 145 141]

[138 140 143 ... 148 145 141]

[143 145 144 ... 136 138 146]]

* Perhitungan sobel menggunakan kernel:

Sx=[-1 0 1] Sy=[1 2 1]

[-2 0 2] [0 0 0]

[-1 0 1] [-1 -2 -1]

* Hasil Citra Sobel

[[ 0 8 9 ... 0 2 0]

[ 2 14 17 ... 0 2 0]

[ 1 12 11 ... 2 3 1]

...

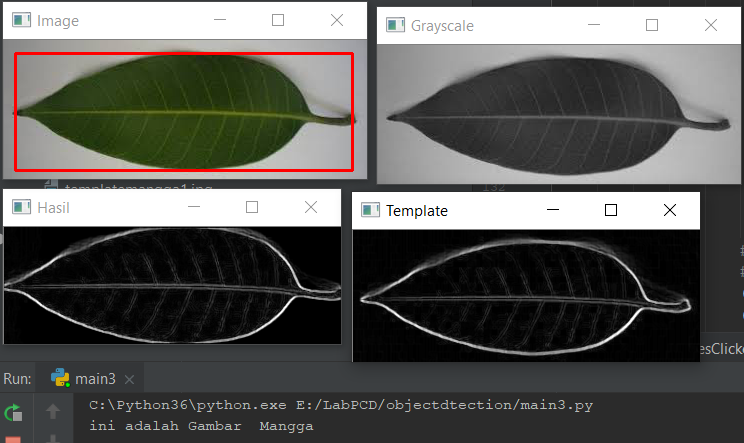
[ 3 10 8 ... 4 14 0]

[ 8 14 6 ... 23 16 2]

[ 0 6 1 ... 5 3 0]]

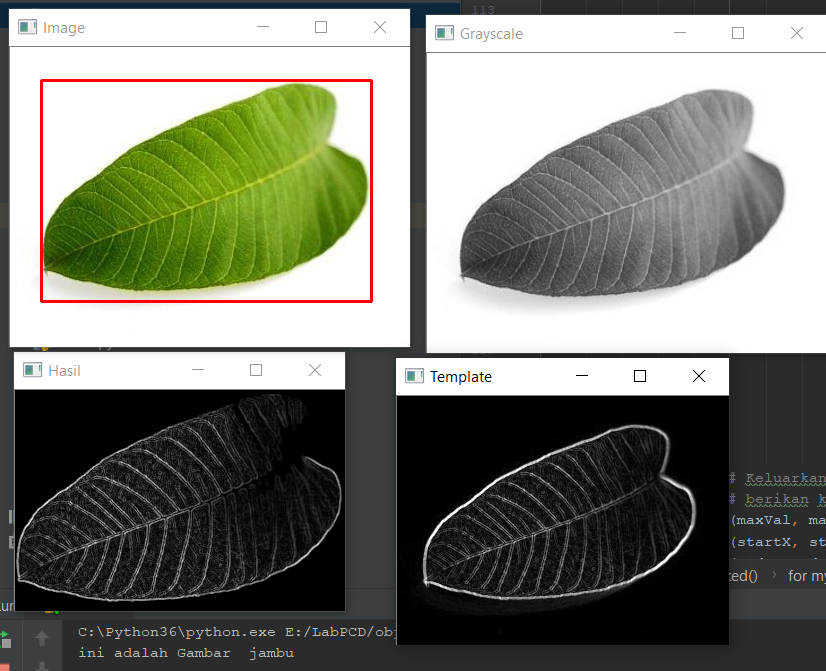
## **4.5 Pengujian dan analisis**

* Pengujian daun mangga



Gambar 4.3 Hasil Pengujian Daun Mangga

* Pengujian Daun Jambu



Gambar 4.4 Hasil Pengujian Daun Jambu

Berdasarkan gambar hasil pengujian diatas dianalis bahwa tepi dapat dilihat oleh kasat mata saat deteksi tepi sobel telah dilakukan. Dimana pada gambar hasil pengujian daun mangga tidak banyak noise atau bintik-bintik ppada objek, cenderung sedikit bersih di dalam objek tersebut. Sedangkan pada gambar hasil pengujian daun jambu terlihat banyak noise atau bintik-bintik pada objek, hal ini dikarenakan pada gambar hasil daun jambu memiliki struktural yang berbeda dnegan daun mangga yaitu daun jambu objek permukaannya lebih kasar yang timbul menyerupai bintik-bintik sehingga terdeteksi sebagai noise atau bintik-bintik yang tidak bersih.

# **KESIMPULAN**

Daun memang bisa terlihat oleh mata kita sendiri secara kasat mata, namun dengan perkembangan teknologi maka tepi dan structural daun dapat diklasifikasi atau diidentifikasi oleh teknologi yang ada. Dalam aplikasi ini citra asli akan diolah dengan menggunakan operasi grayscale, kemudian dilakukan operasi brightness untuk mengatur kecerahan citranya. Selanjutnya dilakukan deteksi tepi menggunakan metode sobel dan dilakukan template matching untuk menemukan bagian-bagian kecil dari gambar yang cocok dengan template gambar tersebut. Pada proses deteksi tepi menggunakan metode sobel, maka dihasilkan citra dengan pengklasifikasian nama jenis daun tersebut berdasarkan tepi dan struktural daunnya.

# **DAFTAR PUSTAKA**

Anggoro, Y. (2018). Anggoro, Y., dkk. 2018. Implementasi Metode Fuzzy K-Nearest Neighbor Untuk Klasifikasi Penyakit Tanaman Kedelai Pada Citra Daun. *. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol. 2, No. 6*, 2381-2389.

Lestari, F. R. (2018). Deteksi Penyakit Tanaman Jeruk Siam Berdasarkan Citra Daun Menggunakan Segmentasi Warna RGB-HSV. *Seminar Nasional Teknologi Terapan Berbasis Kearifan Lokal (SNT2BKL) ISSBN : 978-602-71928-1-2.*

Liantoni. (2015). Deteksi Tepi Citra Daun Mangga Menggunakan Algoritma Ant Colony Optimization. *Seminar Nasional Sains dan Teknologi Terapan III 2015 Institut Teknologi Adhi Tama Surabaya ISBN 978-602-98569-1-0*.

Liantoni, F., & dkk. (2015). Klasifikasi Daun Herbal Menggunakan Metode Naive Bayes Classifier dan K-Nearest Neighbor. *Jurnal SimanteC Vol. 5 No. 1*.

Setiyorini, A. (2017). Perbaikan Kualitas Citra Untuk Klasifikasi Daun Menggunakan MetodeFuzzy K-Nearest Neighbor. *ISSN 2085-4552.*