BAB III. METODOLOGI PENELITIAN

Dalam metode penelitian ini akan menjelaskan tentang metode yang digunakan dan konsep pembuatan keseluruhan sistem, serta melakukan analisa hasil yang didapat.

3.1 Data

Data yang akan digunakan dalam penelitian berupa gambar daun pada tanaman buah apel. Jenis-jenis daun yang digunakan antara lain *Rome Beauty, Manalagi* , dan jenis daun *Anna.*

Berikut data gambar daun apel yang sudah di ambil gambarnya :

Tabel 3. 1 Gambar daun apel

Gambar	Bentuk/Tekstur	Label	Dataset Training	Dataset Testing
	Cabang tulang daun sedikit dan kurang jelas. Serat daunnya banyak tapi tidak jelas	Rome Beauty	30	6
	Cabang tulang daun sedikit dan kurang jelas. Serat daunya banyak dan jelas	Manalagi	30	6

	Cabang tulang daun banyak dan jelas. Serat daunya banyak dan sangat jelas	Anna	30	6

3.2 Akuisisi data

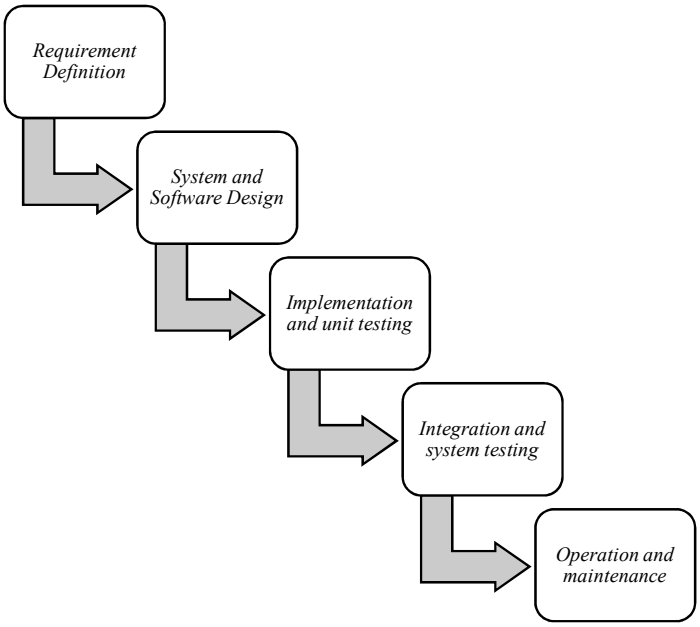
Proses akuisisi menggunakan obyek daun mangga bagian atas untuk kemudian dicitrakan menggunakan kamera *Mirrorless* bermerek Canon EOS M10 dengan jarak antara obyek dan kamera yaitu 30 cm dengan bantuan box yang berlatar belakang warna putih dan lampu (Light Emitting Diode) LED dengan pencahayaan yang sama dan merata. Gambar box sebagai alat untuk pengambilan citra dapat dilihat pada Gambar 3.1.

Gambar 3. 1 Gambar Mini Photo Studio Box

3.3 Metode Pengembangan Perangkat Lunak

Metode Pengembangan sistem yang digunakan dalam melakukan penelitian ini adalah SDLC (*Software Development Life Cycle*) dengan model *waterfall.* Metode *Waterfall* menggambarkan perangkat lunak, dimulai dengan spesifikasi kebutuhan penggna lalu berlanjut melalui tahapan-tahapan perencanaan (planning),

permodelan (modeling), konstruksi (construction), serta yang diakhiri dengan dukungan pada perangkat lunak lengkap yang dihasilkan



Gambar 3. 2 *Software Development Life Cycle*

3.3.1 *Requirement Definition*

Mengumpulkan dan mendifinisikan semua kebutuhan secara lengkap dan melakukan Analisa berdasarkan layanan sistem, kendala, dan tujuan.

3.3.2 *System and software design*

Tahapan perancangan sistem mengalokasikan kebutuhan-kebutuhan seperti mendesain sistem perangkat keras maupun perangkat lunak dengan membentuk arsitektur sistem dan merancang alur sistem secara keseluruhan.

3.3.3 Implementation and unit testing

Tahapan ini merealisasikan perancangan seluruh perangkat lunak sebagai serangkaian program atau unit program.

3.3.4 Integration and system testing

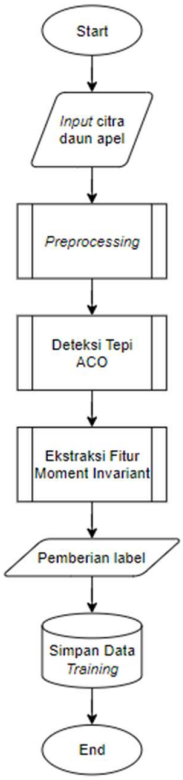
Tahap ini menganggabungkan unit-unit program atau program dan diuji sebagai sebuah sistem lengkap untuk memastikan apakah sesuai dengan kebutuhan atau tidak

3.3.5 Operation and maintenance

Pada tahap ini, melibatkan pembetulan dan memperbaiki jika ada kesalahan-kesalahan yang tidak terduga pada tahapan sebelumnya, meningkatkan implementasi dari unit sistem, dan meningkatkan layanan sistem.

3.4 Proses Training

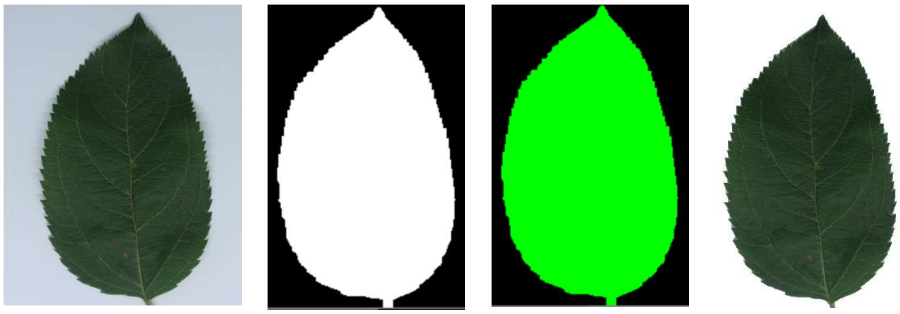
Proses *training* merupakan tahapan untuk memperoleh nilai dari tiap ekstraksi fitur berdasarkan dataset yang digunakan. Nilai yang diperoleh dari proses *training* ini akan digunakan untuk mengidentifikasi *input*an citra pada saat testing. Untuk proses *training* dilakukannya proses pemilihan citra daun apel terlebih dahulu setelah itu citra tersebut dilakukan *preprocessing* dimana citra akan dilakukan proses antaralain *resize*, kemudian *segmentasi thresholding* untuk memisahkan antara *background* dengan *foreground*, dan perubahan citra ke *grayscale*. Setelah itu, citra hasil *preprocessing* akan dilakukan proses deteksi tepi menggunakan ACO setelah itu dilakukan ekstraksi fitur untuk mendapatkan nilai dari fitur-fitur dengan metode *moment invariant*. Kemudian dilakukan pemberian label yang nantinya akan disimpan kedalam *database*.



Gambar 3. 3 Proses tahap *Training*

1. *Preprocessing*

Pada tahap *Preprocessing* ini dimana citra akan dilakukan proses antaralain *resize* yang dimana citra akan diresize menjadi 250 x 250 untuk mempercepat dan memudahkan proses perhitungan, kemudian dilakukan proses *segmentasi thresholding* untuk memisahkan antara *background* dengan *foreground*, dan perubahan citra ke *grayscale*. Pada proses *segmentasi* ini menggunakan *binary image* yang telah diproses menggunakan *threshold otsu*. Setelah dilakukan binerisasi image, selanjutnya dilakukan proses untuk menemukan kontur dari citra berdasarkan objek berwarna putih. Setelah didapatkan kontur berdasarkan *image* warna putih maka langkah selanjutnya yaitu dilakukan *masking* berdasarkan citra inputan, sehingga didapatkan citra yang tersegmentasi. Alur proses segmentasi dapat dilihat pada Gambar 3.4 dibawah ini.



(a) (b) (c) (d)

Gambar 3. 4 Citra input (a); Citra *threshold* (b); Citra *Contour* (c); Citra tersegmentasi (d)

Dari proses segmentasi ini akan didapatkan citra *binary* dari *Treshold*, selanjutnya dari citra binary akan didapatkan kontur citra berdasakan warna putih. Dari hasil kontur yang didapatkan akan di gunakan untuk masking citra terhadap citra asli dimana hasil kontur dapat menghilangkan background citra asli dan mengganti dengan warna putih.

2. Deteksi Tepi

Pada penelitian ini menggunakan metode ACO sebagai metode deteksi adapun langkah dari deteksi tepi *ACO*

a. Inisialisasi Parameter

Tahapan awal dari metode ACO modifikasi adalah inisialisasi parameter masukkan. Parameter terdiri dari jumlah semut (K), jumlah step konstruksi (L), iterasi (N), feromon evaporation rate (ρ), feromon decay (φ), faktor bobot feromon (α), dan faktor pembobot informasi heuristik (β). Matrik informasi heuristik berupa matrik intensitas warna setiap piksel. Matrik feromon awal diinisialisasi 1 atau 0.

b. Perhitungan Gradient

Pada penelitian ini untuk mencari nilai gradient suatu citra menggunakan metode *prewitt.* Metode Prewitt merupakan pengembangan metode robert dengan menggunakan filter HPF yang diberi satu angka nol penyangga. Metode ini mengambil prinsip dari fungsi laplacian yang dikenal sebagai fungsi untuk membangkitkan HPF.

Persamaan gradien pada operator prewitt sama dengan gradien pada operator sobel perbedaannya adalah pada prewitt menggunakan konstanta c = 1.

Gx =

1	0	-1
1	0	-1
1	0	-1

Gy =

1	1	1
0	0	0
-1	-1	-1

Operator *prewitt* merupakan *magnitude* yang dihitung dengan persamaan

$M = \sqrt{Gx^2 + Gy^2}$ .....(3.1)

Seblum dilakukan proses deteksi tepi *Ant Colony Optimization* citra diproses dahulu menggunakan operator prewitt untuk menentukan gradient pada citra. Adapun langkah perhitungan manual dalam ilustrasi pixel berukan 5x5 sebagai berikut:

Tabel 3. 2 Tabel Pixel 5x5

215	**211**	**208**	211	211
158	**151**	**119**	97	101
143	**134**	**134**	125	133
121	127	134	133	134
143	134	134	125	133

Gx = (255 + 0 - 208) + (158 + 0 - 119) + (143 + 0 - 134) = 55

Gy = (255 + 211 + 208)+(0)+(-143-134-134) = 428

Geser kernel satu pixel ke kanan dengan titik koordinat (x,y) sehingga akan didapatkan hasil:

Tabel 3. 3 Tabel Nilai Gx

215	**211**	**208**	211	211
158	**55**	**63**	16	101
143	**35**	**57**	19	133
121	5	12	2	134
143	134	134	125	133

Tabel 3. 4 Tabel Nilai Gy

215	**211**	**208**	211	211
158	**223**	**237**	238	101
143	**46**	**-27**	-84	133
121	0	0	0	134
143	134	134	125	133

Setelah di dapat nilai pixel Gx dan Gy maka nilai tersebut dihitung menggunakan persamaan prewitt sebgai berikut:

M = $\sqrt{Gx^2 + Gy^2}$ .....(3.2)

M = $\sqrt{55^2 + 233^2}$ = 229,6824

Sehingga didapat pixel dengan nilai hasil prewiit sebagai berikut:

Tabel 3. 5 Tabel Nilai Prewitt

215	211	208	211	211
158	229,6824	245,2305	238,5372	101

143	57,80138	63,07139	86,12201	133
121	5	12	2	134
143	134	134	125	133

3. Pemilihan Pixel

Pada deteksi tepi ACO, semut akan berpindah ke piksel tetangga. Piksel ketetanggaan yang digunakan adalah 8 ketetanggaan. Gambar 3.5 berikut ini menunjukkan hubungan ketetanggaan pada tiap piksel.

$I_{i-1,j-1}$	$I_{i-1,j}$	$I_{i-1,j+1}$
$I_{i,j-1}$	I	$I_{i,j+1}$
$I_{i+1,j-1}$	$I_{i+1,j}$	$I_{i+1,j+1}$

Gambar 3. 5 Hubungan Ketetanggaan

Hubungan ketetanggan menunjukkan nilai intensitas yang mewakili tiap piksel dari gambar. Dari nilai intensitas tersebut akan diperoleh informasi heuristik *(η_{ij})* yang ditunjukkan pada persamaan (3.3).

$$\eta_{ij} = \frac{max_{ij}\begin{pmatrix}|I(i-1,j-1)-I(i+1,j+1)|,\\ |I(i-1,j+1)-I(i+1,j-1)|,\\ |I(i,j-1)-I(i,j+1)|,|\\ (i-1,j)-I(i+1,j)|\end{pmatrix}}{\eta_{max}} \quad \text{.....(3.3)}$$

Dengan:

η_{max} = nilai heuristic maksimum (nilai *heuristic* maksimum diambil dari perbandingan hasil *heuristic* sebelumnya dan diambil nilai yang paling besar) jika nilai $max_{ij} > \eta_{max}$ maka $\eta_{max} = max_{ij}$

Berikut contoh perhitungan nilai *heuristic* (η_{ij}):

Tabel 3. 6 Tabel Nilai Citra

215	211	208	211	211
158	229,6824	245,2305	238,5372	101
143	57,80138	63,07139	86,12201	133
121	5	12	2	134
143	134	134	125	133

$$\eta_{ij} = \frac{max_{ij}\begin{pmatrix} |215-63,07139|, \\ |208-143|, \\ |158-245,2305|, \\ |(211-57,80138)| \end{pmatrix}}{\eta_{max}}$$
$$= \frac{(370)}{370}$$
$$= 1$$

Geser kernel satu pixel ke kanan dengan titik koordinat (x,y) sehingga akan didapatkan hasil:

Tabel 3. 7 Tabel *Heuristic*

215	211	208	211	211
158	1	1	1	101
143	1	1	0,62894	133
121	0,169919	0,00432	0	134
143	134	134	125	133

Setelah didapatkan nilai heuristic tahap selanjutnya adalah perhitungan probabilitas. Dimana nilai probalitas merupakan jalur yang akan diikuti dalam perpindahan semut. Perhitungan manual probailitas sebagai berikut:

$$P_{i,j}^{(n)} = \frac{\left(\tau_{i,j}^{n-1}\right)^{a}\left(\left(\eta_{i,j}\right)^{\beta}\right)}{\Sigma_{j\epsilon\Omega_i}\left(\tau_{i,j}^{n-1}\right)^{a}\left(\left(\eta_{i,j}\right)^{\beta}\right)} \; i\, f\, j \in \Omega_i \quad \text{.....(3.4)}$$

Dengan :

$\tau_{i,j}^{n-1} = 0,1$

$\alpha = 0,5$

$\beta = 0,5$

Tabel 3. 8 Tabel perhitungan Nilai Ketentanggaan

$(0,1)^{0,5}$x $((0,1215)^{0,5})$	211	208	211	211
158	1	1	1	101
143	1	1	0,62894	133
121	0,169919	0,00432	0	134
143	134	134	125	133

Tabel 3. 9 Tabel Nilai Ketentanggaan

4,636809	4,593474	4,560702	4,593474	4,593474
3,974921	0,316228	0,316228	0,316228	3,17805
3,781534	0,316228	0,316228	0,250787	3,646917
3,478505	0,412212	0,020785	0	3,660601
3,781534	3,660601	3,660601	3,535534	3,646917

$\sum_{j\epsilon\Omega_i}(\tau_{i,j}^{n-1})^{a}\left((\eta_{i,j})^{\beta}\right)$ = 4,636809 + 4,593474 + 4,560702 + 0,316228 + 0,316228 + 0,316228 + 3,781534 + 3,974921

= 22,49612

$(\tau_{i,j}^{n-1})^{a}\left((\eta_{i,j})^{\beta}\right)$ = 0,316228

$P_{i,j}^{(n)}$ = $\frac{0,316288}{22,49612}$

= 0,014057

Sehingga didapatkan nilai probabilitas sebagai berikut:

Tabel 3. 10 Tabel Nilai Probabilitas

4,636809	4,593474	4,560702	4,593474	4,593474
3,974921	0,014057	0,020718	0,014739	3,17805
3,781534	0,025064	0,162277	0,021893	3,646917
3,478505	0,021677	0,00171	0	3,660601
3,781534	3,660601	3,660601	3,535534	3,646917

4. *Update Process*

Feromon intensitas sebagai informasi heuristik akan menarik semut untuk mengikuti jalur perjalanan semut lainnya. Karena itu, feromon akan diperbarui dua kali, pertama setelah perpindahan yang dilakukan tiap semut dan kedua setelah perpindahan yang dilakukan semua semut. Terdapat dua mekanisme pembaruan feromon (τ), yaitu pembaruan feromon lokal dan pembaruan feromon global. Setiap kali langkah konstruksi, matrik feromon local akan diperbarui karena feromon akan mengalami kerusakan (pheromone decay). Pembaruan feromon lokal bertujuan untuk memverifikasi pencarian yang dilakukan oleh semut berikutnya selama iterasi (Dorigo, dkk., 2006). Persamaan untuk pembaruan feromon lokal ditunjukkan pada persamaan (3.5).

$$\tau_{i,j} = (1-\rho)\tau_{i,j} + \rho\Delta\tau_{i,j} \quad (3.5)$$

Perhitungan manual proses *update local pheromone* dengan $\tau_{i,j} = 0{,}1,\ \rho = 0{,}1$

Tabel 3. 11 Tabel Perhitungan Update Local Pheromon

4,636809	4,593474	4,560702	4,593474	4,593474
3,974921	((1-0,1)*0,014057) + (0,1*1) = 0,112651	0,020718	0,014739	3,17805
3,781534	0,025064	0,162277	0,021893	3,646917
3,478505	0,021677	0,00171	0	3,660601
3,781534	3,660601	3,660601	3,535534	3,646917

Tabel 3. 12 Tabel Hasil Update Local Pheromone

4,636809	4,593474	4,560702	4,593474	4,593474
3,974921	0,112651	0,118646	0,113265	3,17805
3,781534	0,122558	0,246049	0,082598	3,646917
3,478505	0,036501	0,001971	0	3,660601
3,781534	3,660601	3,660601	3,535534	3,646917

Setelah perjalanan semua semut selama langkah kontruksi selesai, akan dilakukan pembaruan feromon global. Persamaan untuk pembaruan feromon global ditunjukkan pada persamaan (3.6).

$$\tau_{i,j} = (1 - \varphi)\tau_{i,j} + \varphi\tau_0 \quad \text{.....(3.6)}$$

Perhitungan manual proses *Update Global Pheromone* sebagai berikut:

Tabel 3. 13 Tabel Perhitungan *Update Local Pheromone*

4,636809	4,593474	4,560702	4,593474	4,593474
3,974921	(1- 0,1) * 0,112651) + (0,1 * 0,1) = 0,111386	0,118646	0,113265	3,17805
3,781534	0,122558	0,246049	0,082598	3,646917
3,478505	0,036501	0,001971	0	3,660601
3,781534	3,660601	3,660601	3,535534	3,646917

Tabel 3. 14 Tabel Perhitungan *Update Global Pheromone*

4,636809	4,593474	4,560702	4,593474	4,593474
3,974921	0,111386	0,116782	0,111938	3,17805
3,781534	0,120302	0,231444	0,084338	3,646917
3,478505	0,042851	0,011774	0,01	3,660601
3,781534	3,660601	3,660601	3,535534	3,646917

5. Penentuan Tepi

Pada langkah ini, sebuah keputusan diambil pada setiap piksel untuk menentukan tepi atau bukan. Cara ini dilakukan dengan mengimplementasikan sebuah threshold (T) pada matrik feromon akhir ($\tau^{(N)}$).

Inisialisasi threshold $T^{(0)}$ dipilih berdasarkan rata-rata nilai feromon matrik. Selanjutnya, feromon matrik diklasifikasinya kedalam dua kategori dengan kriteria nilai kurang dari $T^{(0)}$ atau lebih dari $T^{(0)}$. Inisialisasi nilai $T^{(0)}$ dihitung berdasarkan persamaan (3.7).

$$T_0 = \frac{\sum_{i=1:M1}\sum_{j=1:M2}\tau_{i,j}}{M1.M2} \quad \text{.....(3.7)}$$

$$T_0 = \frac{0{,}840816}{25}$$

$$= 0{,}036996$$

Setelah didapatkan nilai *threshold,* nilai *threshold (T)* akan dibandingkan dengan hasil feromon akhir yang digunakan sebagai nilai fitness dalam menentukan tepi atau bukan. Penentuan tepi dengan nilai fitness yang menentukan sebuah piksel sebagai tepi (E_{ij} = 1) atau bukan (E_{ij} = 1) berdasarkan kriteria seperti ditunjukkan pada persamaan (3.8).

$$E_{ij} = \begin{matrix}1\\0\end{matrix}, \frac{Jika\ \tau_{i,j}{}^{(N)} \geq T_0}{lainya} \quad \text{.....(3.8)}$$

Tabel 3. 15 Tabel Penentuan Kriteria Berdasarkan *Treshold*

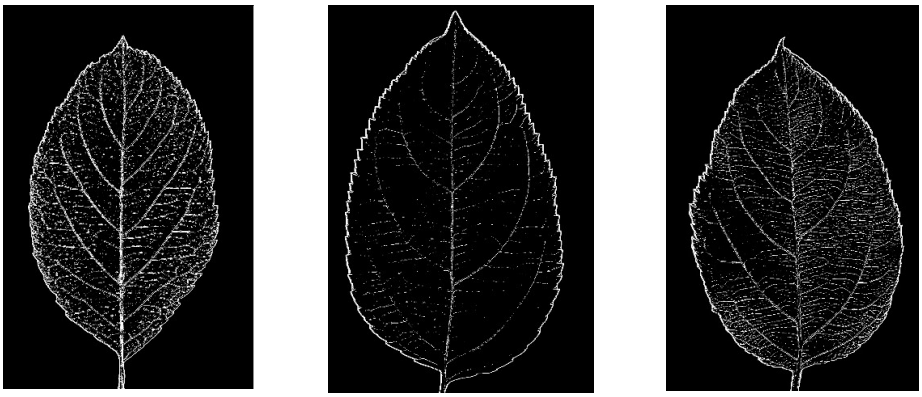
4,636809	4,593474	4,560702	4,593474	4,593474
3,974921	0,111386 >= 0,036996	0,116782 >= 0,036996	0,111938 >= 0,036996	3,17805
3,781534	0,120302 >= 0,036996	0,231444 >= 0,036996	0,084338 >= 0,036996	3,646917
3,478505	0,042851 >= 0,036996	0,011774 <= 0,036996	0,01 <= 0,036996	3,660601
3,781534	3,660601	3,660601	3,535534	3,646917

Tabel 3. 16 Tabel Hasil Penentuan Tepi

4,636809	4,593474	4,560702	4,593474	4,593474
3,974921	1	1	1	3,17805

3,781534	1	1	1	3,646917
3,478505	1	0	0	3,660601
3,781534	3,660601	3,660601	3,535534	3,646917

Hasil deteksi tepi ACO dapat dilihat pada Gambar (3.6).



(a) (b) (c)

Gambar 3. 6 Hasil deteksi tepi, (a) apel anna, (b) apel manalagi, (c) apel rome beauty

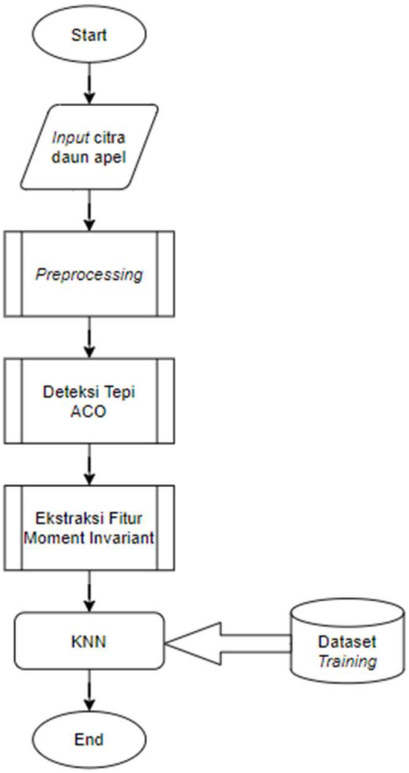
6. Moment Invariant

Selanjutnya dilakukan proses ekstraksi fitur *Moment Invariant,* pada tahap ini penulis menggunakan *library OpenCV* dalam mengerjakan tiap prosesnya, dimana citra dari hasil deteksi tepi ACO diolah untuk didaptkan nilai *Seven Moment Invariant* dengan perhitungan pada persamaan (2.11).

3.5 Proses Testing

Proses testing merupakan proses klasifikasi dari hasil proses *training.* Perbedaan dari proses *training* dan *testing* terletak di metode KNN, pada proses *training* tidak terdapat proses metode KNN. Untuk proses *preprocessing*nya sendiri sama dengan proses *training* yaitu dilakukan input citra daun apel untuk dilakukan testing kemudian dilakukan proses *preprocessing* setelah itu dilakukan deteksi tepi menggunakan ACO kemudian dilakukan ekstraksi fitur menggunakan metode *Moment Invariant*. Setelah ekstraksi fiturnya sudah dilakukan maka akan dilakukan proses klasifikasi dengan menggunakan metode K-NN yang dimana sistem akan mengambil data dalam *database* terlebih dahulu yang kemudian akan dicocokan

dengan hasil dari ekstraksi fitur pada citra *testing*. Apabila hasil ekstraksi fitur pada citra *testing* sama atau mendekati dengan hasil ekstraksi fitur dalam database, maka sistem akan menampilkan hasil output yaitu jenis tanaman apel yang terpilih.



Gambar 3. 7 Proses tahap *Testing*

1. *K-NN*

Dalam penelitian ini KNN digunakan untuk proses mengklasifikasikan yaitu tanaman apel anna, manalagi, dan rome beauty. Pada proses ini, akan dilakukan perhitungan jarak euclidean antara data uji dan data latih menggunakan persamaan (2.4)

Pada proses klasifikasi menggunakan metode KNN dapat dicontohkan seperti pada table dengan menggunakan K = 3 (3.17)

Tabel 3. 17 Contoh Perhitungan KNN

Data Uji (x)	=	0,345
Data Latih 1	=	0,213
$\sqrt{(data\ uji - data\ latih\ 1)^2}$	=	0,132
Data Latih 2	=	0,343
$\sqrt{(data\ uji - data\ latih\ 2)^2}$	=	0,002

Data Latih 3	=	0,191
$\sqrt{(data\ uji - data\ latih\ 3)^2}$	=	0,154
Data Latih 4	=	0,373
$\sqrt{(data\ uji - data\ latih\ 4)^2}$	=	0,028

Setelah menghitung jarak Euclidean, kemudian dilakukan perangkingan seperti pada Tabel (3.18).

Tabel 3. 18 Hasil Perankingan Jarak *Euclidience*

Data Latih Ke-	Nilai *Euclidience*	Ranking
2	0,002	1
4	0,028	2
1	0,132	3

Setelah dilakukan perangkingan, maka dapat ditentukan bahwa data uji tersebut masuk dalam kelas yang sama dengan data latih 2.

3.6 Metode Pengujian

Terdapat 2 pengujian yang dilakukan, yaitu pengujian sistem dan pengujian akurasi. Adapun penjelasannya adalah sebagai berikut :

a. Pengujian sistem

   Pengujian sistem dilakukan saat implementasi sistem dimana pengujian dilakukan untuk menemukan kesalahan program (bug) kemudian memperbaikinya.

b. Pengujian akurasi

   Pengujian akurasi dilakukan dengan cara menghitung jumlah prediksi yang benar dibandingkan dengan jumlah data testing secara keseluruhan. Berikut ini adalah rumus pengujian akurasi :

$$akurasi = \frac{Jumlah\ prediksi\ benar}{jumlah\ dataset}\ x\ 100 \quad \text{.....(3.9)}$$