BAB II. LANDASAN TEORI

Pada bab ini berisi teori-teori yang mendasari dan berkaitan dengan masalah perencanaan dan pembuatan aplikasi yang digunakan untuk memudahkan pemahaman dan pemecahan terhadap masalah yang ada.

2.1 Studi Penelitian Terdahulu

Pada beberapa kasus-kasus penelitian sebelumnya penelitian terdahulu adalah sebagai dasar penulis untuk melakukan penelitian. Dari penelitian terdahulu ini penulis mengangkat beberapa penilitan yang dijadikan rujukan adalah sebagai berikut :

Jurnal hasil penelitian oleh Febri Liantoni pada tahun 2015 yang berjudul "Deteksi Tepi Citra Daun Mangga Menggunakan Algoritma Ant Colony Optimization" dari penelitian tersebut dapat dibuktikan perbedaan hasil deteksi tepi *Ant Colony Optimzation* menghasilkan citra tulang daun yang lebih detail dan memiliki garis tepi yang lebih tebal dibandingkan dengan metode *Robert, Prewit,* dan *Sobel*.

Penelitian yang dilakukan oleh (Liantoni, Suciati, & Fatichah, 2015) yang berjudul "Modifikasi Ant Colony Optimization Berdasarkan Gradient Untuk Deteksi Tepi Citra" mendapatkan hasil deteksi tepi menggunakan ACO modifikasi menunjukkan nilai rata-rata PSNR yang lebih tinggi sebesar 12,724 dibandingkan menggunakan ACO tradisional yang menghasilkan nilai rata-rata PSNR sebesar 12,268.

Dalam penilitan Ni'mah, F. S., Sutojo, T., & Setiadi pada tahun 2018 yang berjudul "Identifikasi Tumbuhan Obat Herbal Berdasarkan Citra Daun Menggunakan Algoritma Gray Level Co-occurence Matrix dan *K-Nearest Neighbor*" dari penelitian tersebut menyatakan untuk melakukan klasifikasinya menggunakan metode KNN memperoleh akurasi sebesar 90%.

2.2 Pengolahan Citra Digital

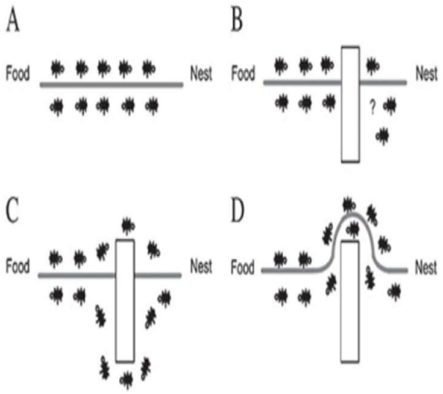
Citra adalah suatu representasi (gambaran), kemiripan, atau imitasi dari suatu objek. Citra terbagi 2 yaitu citra yang bersifat analog dan ada citra yang bersifat

digital. Citra analog adalah citra yang bersifat continue seperti gambar pada monitor televisi, foto sinar X, dan lain-lain. Sedangkan pada citra digital adalah citra yang dapat diolah oleh komputer (Sutojo, 2009). Citra dapat didefinisikan sebagai fungsi f(x,y) berukuran M baris dan N kolom, dengan x dan y adalah koordinat spasial, dan amplitudo f di titik koordinat (x,y) dinamakan intensitas atau tingkat keabuan dari citra pada citra tersebut (Putra, 2010:19).

2.3 *Ant Colony Optimization* (ACO)

Algoritma ACO merupakan metode heuristik yang meniru perilaku semut untuk memecahkan masalah optimasi diskrit(Dorigo, Birattari, & Stützle, 2006). Semut menggunakan senyawa kimia khusus yang disebut *Pheromone* untuk menandai jalur antara sumber makanan dan koloni mereka. Jalur *Pheromone* digunakan oleh semut berikutnya sebagai referensi untuk mencari makanan karena *Pheromone* meningkatkan kemungkinan daun untuk dipilih (Liantoni, 2015).

Terdapat beberapa jenis Algoritma ACO yaitu Ant System (AS), Ant Colony System (ACS), Min-Max Ant System (MMAS), Elitist Ant System (EAS), Rank-Based Ant System (ASRank), Approximate Nondeterministic Tree Search (ANTS) (Etemad & White, 2011).



Gambar 2. 1 Contoh cara kerja semut

Jadi cara kerja semut adalah sebagai berikut :

1. Pada awalnya, semut berkeliling secara acak.
2. Ketika semut-semut menemukan jalur yang berbeda misalnya sampai pada persimpangan, mereka akan mulai menentukan arah daun secara acak

3. Sebagian semut memilih berdaun ke atas dan sebagian lagi akan memilih berdaun ke bawah
4. Ketika menemukan makanan mereka kembali ke koloninya sambil memberikan tanda dengan jejak feromon.
5. Karena jalur yang ditempuh lewat jalur atas lebih pendek, maka semut yang di atas akan tiba lebih dulu dengan asumsi kecepatan semua semut adalah sama
6. Feromon yang ditinggalkan oleh semut di jalur yang lebih pendek aromanya akan lebih kuat dibandingkan feromon di jalur yang lebih panjang
7. Semut-semut lain akan lebih tertariik mengikuti jalur atas karena aroma feromon lebih kuat.

Dari beberapa jenis algoritma di atas, AS dan ACS merupakan jenis ACO yang paling populer digunakan (Gupta, 2013). Algoritma AS merupakan algoritma versi pertama ACO yang diusulkan pada tahun 1992. Kemudian metode ini berkembang menjadi ACS (Dorigo et al., 2006).

Pada penelitian ini penulis memilih jenis algoritma ACS karena hasil penelitian dari Febri Liantoni pada tahun 2015 yang berjudul "Deteksi Tepi Citra Daun Mangga Menggunakan Algoritma Ant Colony Optimizationra Daun Mangga Menggunakan Algoritma Ant Colony Optimization" menghasilkan citra tulang daun yang lebih detail dan memiliki garis tepi yang lebih tebal. algoritma ini telah menerapkan penurunan konsentrasi *Pheromone*, sehingga kemungkinan semut terjebak dalam satu daerah eksplorasi tertentu menjadi lebih kecil (Verma, Singhal, Garg, & Chauhan, 2011). Secara umum, pseudocode ditunjukkan pada Gambar 2.

```
Initialize
SCHEDULE_ACTIVITIES
      ConstructAntSolutions
      DoDaemonActions (optional)
      UpdateFeromons
END_SCHEDULE_ACTIVITIES
```

Gambar 2. 2 *Pseudocode ACO*

- **Initialize**

  Initialize merupakan langkah yang dilakukan di awal proses. Dalam langkah ini dilakukan prosedur inisialisasi, seperti pengaturan parameter dan penempatan nilai *Pheromone* awal (Liantoni, 2015). Adapun parameter yang digunakan yaitu:

  K = Jumlah Semut

  L = Jumlah step konstruksi

  N = Iterasi

  ρ = *Pheromone evaporation rate*

  φ = *Pheromone decay*

  α = factor pembobotan pheromone

  β = factor bobot informasi heuristic

- **ConstructAntSolutions**

  ConstructAntSolutions merupakan kegiatan perdaunan semut. Proses konstruksi berisi sejumlah langkah-langkah konstruksi. Semut akan bergerak dalam suatu gambar sampai jumlah target langkah konstruksi terbentuk. Pada proses konstruksi ke *(n^{th})* jumlah semut *(k^{th})* akan berpindah dari node *(i)* ke node *(j)* yang mengikuti perpindahan probabilitas *($P_{i,j}^{(n)}$)* (Liantoni, 2015). Aturan *proportional pseudorandom* ini ditunjukan pada persamaan (2.1):

$$P_{i,j}^{(n)} = \frac{\left(\tau_{i,j}^{n-1}\right)^{\alpha}\left((\eta_{i,j})^{\beta}\right)}{\sum_{j\epsilon\Omega_i}\left(\tau_{i,j}^{n-1}\right)^{\alpha}\left((\eta_{i,j})^{\beta}\right)} \; i\; f\; j \in \Omega_i \quad \text{.....(2.1)}$$

  Keterangan :

  $P_{i,j}^{(n)}$ = perpindahan probalitas,

  a = factor pembobotan *pheromone,*

  β = factor bobot informasi heuristic,

  Ω_i = node ketetanggaan dari semut yang diberikan pada node ke *(i),*

  τ = *update pheromone,*

  η = informasi heuristic.

- **DoDaemonActions**

  DoDaemonActions merupakan solusi konstruksi yang dilakukan untuk tambahan langkah sebelum pembaruan nilai feromon. Kegiatan ini sebagai tindakan tambahan sebelum memperbarui nilai-nilai feromon. Proses ini tidak bisa dilakukan jika hanya dengan single ant (Liantoni, 2015).

- **UpdateFeromons**

  UpdateFeromons merupakan kegiatan pembaruan feromon setelah proses kontruksi dan daemon actions dilakukan. Terdapat dua kali update, yaitu update global feromon dan update lokal feromon. Pembaruan lokal feromon dilakukan setiap kali langkah konstruksi. Pada tahap ini feromon akan mengalami kerusakan (*pheromone decay*). Hal ini bertujuan untuk menurunkan konsentrasi *pheromone* di tepi yang dilalui (Liantoni, 2015). Rumus *Update local pheromone* ditunjukkan pada persamaan (2.2) :

  $\tau_{i,j} = (1 - \varphi)\tau_{i,j} + \varphi\tau_0$ .....(2.2)

  Keterangan :

  φ = Kerusakan *pheromone,*

  τ = *Update pheromone.*

- **Update Global pheromone**

  Update global feromon dilakukan setelah step konstruksi maksimal dalam satu iterasi dilalui. Pada tahap ini terjadi penguapan feromon (Liantoni, 2015). Update global feromon ditunjukkan pada persamaan (2.3) :

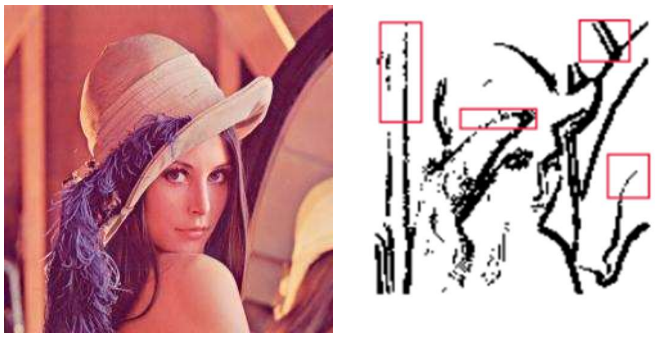
  $\tau_{i,j} = (1 - \rho)\tau_{i,j} + \rho\Delta\tau_{i,j}$ .....(2.3)

  Keterangan :

  ρ = *Pheromone* tingkat penguapan,

  $\Delta\tau_{i,j}$ = Total *Pheromone* untuk pembaruan feromon global.

Gambar 2.4 Dibawah ini adalah contoh hasil deteksi tepi menggunakan ACO :



(a) (b)

Gambar 2. 3 Hasil deteksi tepi, (a) gambar data uji, (b) hasil ACO
(Sumber : Liantoni et al., 2015)

2.4 *K-Nearest Neighbor* (K-NN)

Algoritma K-Nearest Neighbor (K-NN) merupakan metode yang digunakan untuk mengelompokkan objek berdasarkan contoh latih terdekat di ruang fitur. Algoritma algoritma K-Nearest Neighbor adalah pendekatan untuk mencari kasus dengan menghitung kedekatan antara kasus baru dengan kasus lama berdasarkan kecocokan bobot dari sejumlah fitur yang ada yang memiliki kesamaan. Algoritma K-Nearest Neighbor memiliki tujuan untuk mengklasifikasikan objek baru berdasarkan atribut dan training sample.

Algoritma K-NN akan mencari jarak terpendek dari sample uji (testing sample) ke sample latih (training sample) untuk menentukan K-NNnya. Salah satu untuk menghitung jarak terdekat atau jauhnya tetangganya maka digunakan metode *euclidean distance*. Pada *euclidean distance* sendiri berfungsi untuk menguji ukuran yang bisa digunakan sebagai interpretasi kedekatan jarak antara dua obyek. Klasifikasi K-NN mempunyai langkah yaitu :

a. Menentukan parameter k (nomor keturunan terdekat).
b. Menghitung kuadrat jarak eucliden objek terhadap pelatihan data yang diberikan.

$$d = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2 + \cdots + n} \quad \text{.....(2.4)}$$

c. Mengurutkan hasil dari operasi kuadrat jarak eucliden objek secara naik (berurutan dari nilai bawah ke ketinggi)

d. Mengumpulkan kategori (x) (klasifikasi tetangga terdekat berdasarkan nilai k)

e. Dengan menggunakan kategori tetangga terdekat yang terbaik dapat dipredisikan kategori objek.

2.5 *Grayscale Image*

Grayscale adalah suatu citra yang hanya memiliki warna tingkat keabuan. Penggunaan citra grayscale dikarenakan membutuhkan sedikit informasi yang diberikan pada tiap piksel dibandingkan dengan citra berwarna. Warna abu-abu pada citra grayscale adalah warna R (Red), G (Green), B (Blue) yang memiliki intensitas yang sama. Sehingga dalam grayscale image hanya membutuhkan nilai intensitas tunggal dibandingkan dengan citra berwarna membutuhkan tiga intensitas untuk tiap pikselnya. Intensitas dari citra *grayscale* disimpan dalam 8 bit integer yang memberikan 256 kemungkinan yang mana dimulai dari level 0 sampai dengan 255 (0 untuk hitam dan 255 untuk putih dan nilai diantaranya adalah derajat keabuan).

Terdapat 3 Metode untuk merubah RGB menjadi citra *Grayscale image*:

1. *Lightniess* $= \frac{(\max[R,G,B]+\min[R,G,B])}{2}$ ..... (2.5)
2. *Average* $= \frac{(R+G+B)}{2}$ ..... (2.6)
3. *Luminosity* $= (0.21 * R) + (0.72 * G) + (0.07 * B)$ ..... (2.7)

Original Image *Lightness* *Average* *Luminosity*

Gambar 2. 4 Hasil konversi RGB – Citra *grayscale*

2.6 *Moment Invariant*

Seven Moment Invariant atau biasanya hanya disebut *Moment Invariant* saja, merupakan metode yang digunakan untuk proses ekstraksi fitur dari sebuah obyek citra. Metode ini bisa mengetahui perubahan rotasi sumbu obyek, skala dari obyek dan perubahan posisi obyek. Dengan mendapatkan sejumlah informasi nilai momen, baik momen tingkat ke nol, momen sentral dan momen tingkat lanjutanya, maka obyek tersebut dapat diidentifikasi sekalipun telah mengalami pergeseran (translasi), perputaran (rotasi) maupun perubahan skala (scale) (Liantoni & Hermanto, 2017).

Momen invariant telah diperkenalkan oleh Hu pada tahun 1961 (Hu, 1961). Hu memperkenalkan momen invariant untuk citra digital dengan ukuran M × N piksel (Haryono, Hapsari, Angesti, & Felixiana, 2016), dihitung dengan menggunakan persamaan (2.8).

$$m_{pq} = \sum_{X=1}^{M} \sum_{y=}^{N} x^p y^p \, f(x,y) \quad \text{.....(2.8)}$$

dengan f(x,y) merupakan nilai piksel pada koordinat (x,y).

Invarian translasi dapat dihitung dengan menggunakan central moment yang didefinisikan dengan persamaan (2.9).

$$\mu_{pq} = \sum_{X=1}^{M} \sum_{y=}^{N} (x - \bar{x})^p (y - \bar{y})^q \, f(x,y) \quad \text{.....(2.9)}$$

dengan $\bar{x} = \frac{m1}{m0}, dan\ \bar{y} = \frac{m01}{m00}$

Central moment yang dinormalisasi didefiniskan dengan persamaan (2.10).

$$\boldsymbol{\eta}_{pq} = \frac{\mu_{pq}}{(\mu_{pq})^{\lambda}} \quad \text{.....(2.10)}$$

dengan $\lambda = \frac{(i+j)}{2} + 1$

Berdasarkan momen ternormalisasi di atas, Hu memperkenalkan tujuh invariant yang diberikan dalam persamaan (2.11).

$$\begin{aligned}
\boldsymbol{Hu_1} &= \boldsymbol{\eta_{20}} + \boldsymbol{\eta_{02}} \\
\boldsymbol{Hu_2} &= (\boldsymbol{\eta_{20}} - \boldsymbol{\eta_{02}})^2 + \boldsymbol{4\eta_{11}}^2 \\
\boldsymbol{Hu_3} &= (\boldsymbol{\eta_{30}} - \boldsymbol{3\eta_{12}})^2 + (\boldsymbol{3\eta_{21}} - \boldsymbol{\eta_{03}})^2 \\
\boldsymbol{Hu_4} &= (\boldsymbol{\eta_{30}} + \boldsymbol{\eta_{12}})^2 + (\boldsymbol{\eta_{21}} + \boldsymbol{\eta_{03}})^2
\end{aligned} \quad (2.11)$$

$$Hu_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})\,[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

$$Hu_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})]$$

$$Hu_7 = (\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})^2[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (\eta_{30} - 3\eta_{12})(\eta_{21} - \eta_{03})\,[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

2.7 *Python*

Python adalah salah satu bahasa pemograman tingkat tinggi yang bersifat interpreter, interactive, object oriented, dan dapat beroperasi hampir di semua platform: Mac, Linux, dan Windows. *Python* termasuk bahasa pemograman yang mudah dipelajari karena sintugas yang jelas, dapat dikombinasikan dengan penggunaan modul-modul siap pakai, dan struktur data tingkat tinggi yang efisien (Kadir, 2019).

2.8 *Open CV*

OpenCV adalah suatu library gratis yang dikembangkan oleh developer-developer Intel Corporation. Library ini terdiri dari fungsi-fungsi computer vision dan API (Aplication Programming Interface) untuk image processing high level maupun low level dan sebagai optimasi aplikasi realtime. *OpenCV* sangat disarankan untuk programmer yang akan berkutat pada bidang computer vision, karena library ini mampu menciptakan aplikasi yang handal, kuat dibidang digital vision, dan mempunyai kemampuan yang mirip dengan cara pengolahan visual pada manusia.