**KLASIFIKASI HAMA MELOIDOGYNE PADA TANAMAN DENGAN IMAGE PROCESSING MENGGUNAKAN OPENCV**

**TUGAS AKHIR**

*Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik*



**Baginda Bonar Siregar**

**14110210019**

**PROGRAM STUDI TEKNIK KOMPUTER**

**FAKULTAS TEKNIK DAN INFORMATIKA**

**UNIVERSITAS MULTIMEDIA NUSANTARA**

**TANGERANG**

**2020**

# **HALAMAN PENGESAHAN TUGAS AKHIR**

**Tugas akhir dengan judul:**

**MENDETEKSI UMUR DAN FASE PERTUMBUHAN HAMA PADA TANAMAN DENGAN IMAGE PROCESSING MENGGUNAKAN OPENCV**

Oleh

Baginda Bonar Siregar

14110210019

Telah dinyatakan diterima dan dapat dilanjutkan pada

Sidang Proposal Tugas Akhir Universitas Multimedia Nusantara

Tangerang, 19 Agustus 2019

Menyetujui,

|  |  |
| --- | --- |
| Dosen Pembimbing | Ketua Program Studi |
| Nabila Husna Shabrina, S.T. M.T. | Samuel Hutagalung, M.T.I. |

# **PERNYATAAN TIDAK MELAKUKAN PLAGIAT**

Dengan ini saya,

Nama : Baginda Bonar Siregar

NIM : 14110210019

Fakultas : Teknik dan Informatika

Program Studi : Teknik Komputer

Menyatakan bahwa skripsi yang berjudul “KLASIFIKASI HAMA MELOIDOGYNE PADA TANAMAN DENGAN IMAGE PROCESSING MENGGUNAKAN OPENCV” ini adalah karya ilmiah saya sendiri, bukan plagiat dari karya ilmiah yang ditulis oleh orang lain atau lembaga lain, dan semua karya ilmiah orang lain atau lembaga lain yang dirujuk dalam skripsi ini telah disebutkan sumber kutipannya serta dicantumkan di Daftar Pustaka.

Jika di kemudian hari terbukti ditemukan kecurangan / penyimpangan, baik dalam pelaksanaan skripsi maupun dalam penulisan laporan skripsi, saya bersedia menerima konsekuensi dinyatakan TIDAK LULUS untuk mata kuliah Skripsi yang telah saya tempuh.

Tangerang, 2 Januari 2020

Baginda Bonar Siregar

# **KATA PENGANTAR**

Puji syukur kepada Tuhan Yang Maha Esa atas berkat dan anugerah-Nya, sehingga tugas akhir ini dapat diselesaikan dengan baik. Tugas akhir dengan judul “MENDETEKSI UMUR DAN FASE PERTUMBUHAN HAMA PADA TANAMAN DENGAN OPENCV” ini diajukan untuk memenuhi salah satu persyaratan dalam memperoleh gelar Sarjana Teknik, Program Studi Teknik Komputer, Fakultas Teknik dan Informatika, Universitas Multimedia Nusantara, Tangerang.

Skripsi ini dapat terselesaikan karena adanya bimbingan, bantuan, doa serta dukungan dari berbagai pihak. Oleh sebab itu, penulis mengucapkan terima kasih kepada:

1. Dr. Ninok Leksono, Rektor Universitas Multimedia Nusantara,
2. Hira Meidia, Ph.D., Wakil Rektor Bidang Akademik Universitas Multimedia Nusantara,
3. Ir. Andrey Andoko, M.Sc., Wakil Rektor Bidang Administrasi Umun dan Keuangan Universitas Multimedia Nusantara,
4. Ika Yanuarti, S.E., MSF, Wakil Rektor Bidang Kemahasiswaan Universitas Multimedia Nusantara,
5. Prof. Dr. Muliawati G. Siswanto, M.Eng.S.c, Wakil Rektor Bidang Hubungan dan Kerjasama Universitas Multimedia Nusantara,
6. Friska Natalia, Ph.D., Dekan Fakultas Teknik dan Informatika Universitas Multimedia Nusantara,
7. Samuel Hutagalung, M.T.I., Selaku Ketua Program Studi Teknik Komputer Universitas Multimedia Nusantara,
8. Nabila Husna Shabrina, S.T. M.T., Selaku Dosen Pembimbing yang telah membimbing pembuatan tugas akhir dan yang telah mengajar penulis tata cara menulis karya ilmiah dengan benar.
9. Penulis juga mengucapkan terima kasih kepada keluarga dan segenap sahabat atas dukungan, doa, bantuan, dan perhatian yang diberikan.

Semoga tugas akhir ini dapa bermanfaat, baik sebagai sumber informasi maupun sumber inspirasi, bagi para pembaca.

Tangerang, Januari 2020

Penulis

# **ABSTRAK**

Hama pada tanaman merupakan momok menakutkan bagi para petani dan orang –orang yang bekerja dalam bidang perkebunan baik dalam skala kecil maupun skala kecil. Kerugian yang diakibatkan oleh hama ini kepada setiap tahunnya sangatlah besar, maka diperlukan cara cepat untuk bisa mendeteksi hama ini dengan cara melakukan klasifikasi karakteristik dengan program yang portable dan bisa diaplikasikan secara cepat. Penelitian ini menggunakan *library image processing* OpenCV yang digunakan untuk mengklasifikasi hama berjenis *meloidogyne* beserta variannya dengan cara menghitung jumlah piksel dalam gambar dan hasil tersebut di konversi menjadi panjang sebenarnya dari hama tersebut.

Kata Kunci: Python, OpenCV, meloidogyne, image processing, segmentation, skeletonization

# **ABSTRACT**

Pests on plants are a frightening specter for farmers and people who work in plantations both on a small scale and large scale. The losses caused by these pests are huge every year, so we need a quick way to detect these pests by classifying characteristics with a portable program that can be applied quickly. This study uses the OpenCV image processing library which is used to classify meloidogyne pests and their variants by counting the number of pixels in the image and the results are converted to the actual length of the pest.

Keywords: Python, OpenCV, meloidogyne, image processing, segmentation, skeletonization

**DAFTAR ISI**

[**HALAMAN PENGESAHAN TUGAS AKHIR** ii](#_Toc27692453)

[**PERNYATAAN TIDAK MELAKUKAN PLAGIAT** iii](#_Toc27692454)

[**KATA PENGANTAR** iv](#_Toc27692455)

[**ABSTRAK** vi](#_Toc27692456)

[**ABSTRACT** vii](#_Toc27692457)

[**BAB 1** 1](#_Toc27692458)

[**PENDAHULUAN** 1](#_Toc27692459)

[**1.1** **Latar Belakang Masalah** 1](#_Toc27692460)

[**1.2 Rumusan Masalah** 2](#_Toc27692461)

[**1.3 Batasan Masalah** 2](#_Toc27692462)

[**1.4 Tujuan Penelitian** 3](#_Toc27692463)

[**1.5 Manfaat Penelitian** 3](#_Toc27692464)

[**BAB 2** 4](#_Toc27692465)

[**TINJAUAN PUSTAKA** 4](#_Toc27692466)

[**2.1 OpenCV** 4](#_Toc27692467)

[**2.2 Python** 5](#_Toc27692468)

[**2.3 Anaconda** 5](#_Toc27692469)

[**2.4 Digital Image Processing** 5](#_Toc27692470)

[**2.4.1 Grayscale** 6](#_Toc27692471)

[**2.4.2 Binarization** 6](#_Toc27692472)

[**2.4.3 Image Segmentation** 6](#_Toc27692473)

[**2.4.4 Skeleton Algorithm** 11](#_Toc27692474)

[**2.5 Penelitian Terdahulu** 12](#_Toc27692475)

[**2.5.1 An Algorithm to Extract Physical Characteristics of Nematodes from Microscopic Images of Plant Roots** 12](#_Toc27692476)

[**BAB 3** 13](#_Toc27692477)

[**METODOLOGI PENELITIAN** 13](#_Toc27692478)

[**3.1 Diagram Blok Sistem** 13](#_Toc27692479)

[**3.2 Diagram Flowchart** 13](#_Toc27692480)

[**3.3 Instrumen Penelitian** 15](#_Toc27692481)

[**BAB 4** 16](#_Toc27692482)

[**IMPLEMENTASI DAN ANALISIS** 16](#_Toc27692483)

[**4.1 Implementasi** 16](#_Toc27692484)

[**4.1.1** **Grayscale** 16](#_Toc27692485)

[**4.1.2** **Gaussian Blur** 18](#_Toc27692486)

[**4.1.3** **Canny Edge Detection** 19](#_Toc27692487)

[**4.1.4** **Contour & Bounding Box** 21](#_Toc27692488)

[**4.1.5** **Opening** 21](#_Toc27692489)

[**4.1.6** **Closing** 22](#_Toc27692490)

[**4.1.7** **Skeletonization** 23](#_Toc27692491)

[**4.2 Analisis** 24](#_Toc27692492)

[**BAB 5** 28](#_Toc27692493)

[**KESIMPULAN DAN SARAN** 28](#_Toc27692494)

[**5.1 Kesimpulan** 28](#_Toc27692495)

[**5.2 Saran** 29](#_Toc27692496)

[**DAFTAR PUSTAKA** 30](#_Toc27692497)

# **BAB 1**

# **PENDAHULUAN**

## **Latar Belakang**

Hama adalah organisme yang dianggap merugikan dan tak diinginkan dalam kehidupan manusia sehari-hari. Walaupun istilah tersebut dapat digunanakan untuk semua organisme, dalam praktiknya istilah ini paling sering dipakai hanya kepada hewan. Dalam pertanian, hama adalah organisme pengganggu tanaman yang bertanggung jawab dalam kerusakan tanaman secara fisik. Dan kedalamnya, praktis adalah semua hewan yang menyebabkan kerugian dalam pertanian [1] [2].

Salah satu hama yang paling banyak menyerang tanaman adalah spesimen *pythoparasitic nematodes* jenis *meloidogyne/radhopulus* yang menyebabkan kerugian global sebesar USD $100 Miliar, hama ini menyerang dengan cara menempel pada akar tanaman dan mengkonsumsi zat dan nutrien dalam tanaman serta menghalangi akar untuk menyerap makanan dari dalam tanah [3].

Penelitian yang dilakukan oleh Alexis Toribio, Luis Vargas, Guillermo Kemper, dan Alfonso Palomo di Peru [4], menemukan bahwa hama jenis *pythoparasitic nematodes* ini merupakan salah satu hama yang paling banyak menimbulkan kerusakan dan kerugian, terutama pada tanaman buah-buahan. Ada beberapa metode analisis yang dapat digunakan untuk mendeteksi hama jenis ini, diantaranya yang paling akurat adalah *realtime PCR,* dimana hama tersebut diteliti dan dikelompokkan berdasarkan *DNA* [4].

Untuk mempercepat analisis jenis nematoda, mereka memilih untuk meneliti berdasarkan karakteristik fisik dari hama tersebut menggunakan image processing, dari penelitian tersebut dapat diekstrak panjang hama tersebut dan dapat diketahui umur dan fase dimana hama sedang berada. Metode dengan image processing yang dilakukan dapat memprediksi keakuratan umur dan fase dan hama tersebut sampai 85% [4].

Penelitian ini dibuat bertujuan untuk melanjutkan penelitian sebelumnya yang dilakukan oleh Alexis Toribio dan koleganya dengan menambahkan beberapa metode dan algoritma yang berbeda, serta menggunakan library OpenCV untuk mengolah data gambar dengan metode segmentasi agar dapat mengukur panjang hama tersebut, dan diharapkan mendapat hasil yang lebih baik dari penelitian sebelumnya.

## **1.2 Rumusan Masalah**

Berdasarkan latar belakang masalah diatas, penting sekali untuk mengetahui fase dan umur hama *pythoparasitic nematodes* berjenis *meloidogyne/radhopulus*  tersebut agar dapat dilakukan solusi yang tepat dan efisien untuk mengatasi hama tersebut.

## **1.3 Batasan Masalah**

Batasan masalah dari penelitian ini adalah:

1. Hanya akan mendeteksi panjang dari hama.
2. Menggunakan algoritma *binarization* dan *skeletonization* pada *OpenCV.*
3. Menggunakan *image segmentation* agar mempermudah untuk pendeteksian dan memisahkan dari objek asing yang dapat mengganggu peneltian.
4. Hanya mendeteksi hama jenis *meloidogyne* beserta varian spesimennya*.*

## **1.4 Tujuan Penelitian**

Tujuan dari penelitian ini adalah menggunakan algoritma *computer vision* pada OpenCV untuk mengetahui karakteristik fisik dari hama tanaman jenis *pythoparasitic nematodes* agar dapat mengetahui umur dan fase dari hama tersebut, dari pengetahuan tersebut bisa diketahui solusi yang tepat untuk mengatasinya.

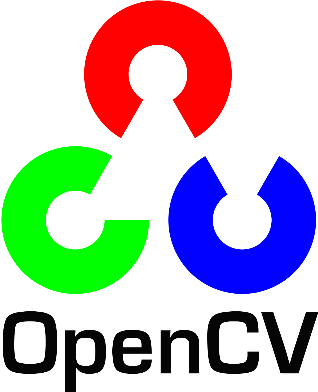
## **1.5 Manfaat Penelitian**

Manfaat dari penelitian ini adalah mengekstrak informasi karakteristik fisik dari hama berupa panjangnya, dari data tersebut ilmuwan dapat dengan mudah mengidentifikasi jenis, serta umur dan fase dimana hama ini sedang berada. Sehingga dapat dilakukan tindakan preventif yang tepat dan efisien.

# **BAB 2**

# **TINJAUAN PUSTAKA**

## **2.1 OpenCV**



Gambar 2.1.

OpenCV (Open Source Computer Vision Library) adalah *library* untuk *computer vision* dan *machine learning* berkode sumber terbuka. OpenCV dibuat untuk menyediakan infrastruktur umum untuk pengaplikasian *computer vision* dan mempercepat pembangunan projek yang menggunakan algoritma *machine learning* baik di bidang komersial maupun penggunaan pribadi.

Terdapat lebih dari 2500 algoritma yang telah disempurnakan dalam *library* OpenCV, diantaranya adalah algoritma untuk mendeteksi wajah, mengidentifikasi objek, mengklasifikasi pergerakan manusia dari sebuah video, mengikuti pergerakan kamera, mengikuti pergerakan objek yang sedang bergerak, mengekstrak model 3 dimensi dari sebuah objek, menyatukan gambar-gambar yang terpisah menjadi satu gambar yang beresolusi tinggi, menemukan sebuah gambar yang sama dari sekumpulan gambar yang sangat banyak, mengikuti pergerakan mata manusia, menghapus efek *red eye* pada gambar, dan lain sebagainya. OpenCV memiliki lebih dari 47 ribu anggota dalam komunitasnya dan lebih dari 18 juta jumlah download dari seluruh dunia. *Library* OpenCV digunakan di banyak perusahaan, proyek penelitian, dan juga dalam pemerintahan [5].

## **2.2 Python**



Gambar .2.1

Bahasa pemrograman yang dibuat oleh Guido Von Rossum pada 1991, merupakan Bahasa pemrograman level abstraksi tinggi, *interpreted,* dan dapat digunakan untuk segala keperluan projek yang berbeda-beda. Python mendukung tipe pemrograman berorientasi objek dan juga berorientasi terstruktur [6].

## **2.3 Anaconda**

Merupakan program gratis berkode sumber terbuka yang biasa digunakan untuk penelitan dan pengembangan *scientific computing*, seperti *data science,* aplikasi *machine learning,* pengolahan data yang besar, analisis dan visualisasi data, dan sebagainya.

## **2.4 Digital Image Processing**

Dalam disiplin ilmu komputer, *Digital image processing* adalah penggunaan algoritma komputer untuk memproses gambar mentah menjadi gambar yang diinginkan, sebagai sub bagian dari *Digital Signal Processing*, *Digital image processing* memiliki keunggulan dibandingkan *Analog image processing*, yaitu dapat menggunakan banyak algoritma yang memang dibuat dan ditujukan untuk pengolahan digital dan juga lebih tahan terhadap noise dan distorsi sinyal dibandingkan pengolahan analog [7].

Beberapa algoritma dalam *Digital Image Processing* yang sering dipakai:

### **2.4.1 Grayscale**

Dalam *Digital image processing*, umumnya gambar yang akan diolah dikonversi dulu ke mode warna grayscale karena bersifat monokrom, yaitu hanya menggunakan satu warna 8 bit, dan juga mempermudah komputer dalam mengolahnya.

### **2.4.2 Binarization**

Adalah mode warna yang hanya menggunakan 2 nilai warna, yaitu hitam dan putih. Binarization dibuat dengan menggunakan *threshold*, yaitu angka patokan untuk menentukan apakah sebuah gambar grayscale akan diubah ke warna hitam atau putih. Jika nilai warna dalam grayscale lebih besar dari *threshold* maka warna akan diubah menjadi warna putih dan jika nilai warna grayscale lebih kecil dari *threshold* akan diubah menjadi warna hitam.

### **2.4.3 Image Segmentation**

Dalam *computer vision*, *image segmentation* adalah proses untuk membagi-bagi sebuah gambar digital menjadi beberapa segmen, dalam kaitan ini, segmen tersebut adalah sebuah set dari banyak sekali piksel, atau bisa disebut sebagai *superpixel*). Tujuan utama dari segmentasi adalah untuk menyederhanakan dan atau mengganti representasi sebuah gambar dari sesuatu kompleks menjadi sesuatu yang lebih bermakna dan lebih mudah untuk dianalisa [8].

*Image segmentation* biasa digunakan untuk menentukan batas-batas dari suatu objek berupa garis, lekukan, dan tepi dari sebuah gambar. Lebih tepatnya, segmentasi adalah proses untuk memberi label dari setiap batas-batas objek tersebut agar komputer lebih mudah membedakan karakteristik dari objek yang akan diproses/dianalisa [9].



Gambar 2.5.3

Dari gambar diatas terlihat sekumpulan pemain *American Football* sedang bermain di atas lapangan yang memiliki rumput berwarna hijau dengan beberapa pohon dengan daun berwarna hijau dan juga pohon dengan daun berwarna kecoklatan, serta langit yang berwarna biru.

Dengan menggunakan visualisasi mata manusia, dapat dengan mudah sekali membedakan objek apa saja yang terdapat di gambar tersebut dengan cepat.



Gambar 2.5.3

Gambar diatas menunjukkan algoritma *superpixel* yang mencontohkan apa yang komputer dan program *computer vision* seperti OpenCV lihat, yaitu sebuah gambar yang dikelompokkan berdasarkan warna yang sama dan juga tekstur yang sama, bukan mengelompokkan berdasarkan objek yang berbeda-beda.

*Image segmentation* dibagi menjadi 3 bagian:

1. *Semantic Segmentation*

Di dalam *Semantic Segmentation*, tujuan utamanya adalah membagi-bagi setiap objek pada gambar yang ingin dianalisa dengan cara memberikan label (seperti pohon, rumah, orang, hewan, rumput, dan sebagainya) [10].



Gambar 2.5.3-1

Jika algoritma *semantic segmentation* diaplikasikan ke gambar raw 2.4.1 maka OpenCV akan memberikan semacam label untuk membedakan masing-masing objek dengan memberi warna yang berbeda. Manusia diberikan label warna merah #FB0200, rumput diberikan label warna mint #9AFA98, Latar belakang pepohonan diberikan label warna hijau #6A8E22, dan langit diberikan warna biru #4480B2.

Dari pengklasifikasian dengan label warna ini, dapat ditentukan objek mana yang mewakilkan sesuatu dengan melihat warnanya, tetapi masih belum diketahui apabila ada dua atau lebih label warna tersebut mewakilkan satu objek atau tidak, sebagai contoh, komputer akan tahu apabila warna merah itu mewakilkan manusia, tetapi tidak dapat mengetahui ada berapa jumlah manusia dalam label berwarna merah tersebut.

1. *Instance Segmentation*

Didalam algoritma *instance segmentation*, komputer dan aplikasi *computer vision* seperti OpenCV tidak akan memberi label warna terhadap semua objek yang terdapat dalam gambar, melainkan hanya akan menyorot objek yang diinginkan saja [11].

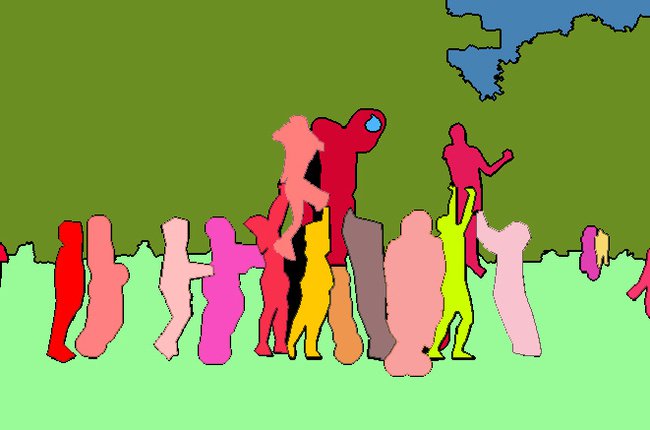


Gambar 2.5.3-2

Dari gambar diatas, terlihat algoritma *instance segmentation* hanya menyorot objek yang di persepsikan sebagai manusia dan memberikan warna yang berbeda-beda satu sama lain sehingga dapat dengan mudah membedakan objek manusia satu dengan objek manusia lainnya.

1. *Panoptic Segmentation*

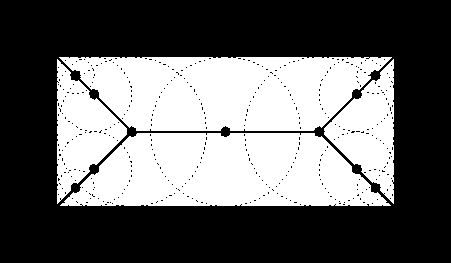
Merupakan gabungan dari *semantic segmentation* dan juga *instance segmentation.* Setiap objek diberikan label yang berbeda-beda sama seperti *semantic segmentation* tetapi untuk objek yang ingin di proses, dalam hal ini objek manusia yang terdapat dalam gambar 2.4.1, maka diberikan juga warna yang berbeda-beda agar dapat dengan mudah membedakan objek manusia satu dengan yang lainnya [11].



Gambar 2.5.3-3

Dari gambar diatas terlihat bahwa objek yang tidak relevan untuk diproses seperti rumput diberi label dengan warna mint, latar belakang pepohonan diberikan label warna hijau, dan langit diberikan warna biru. Sedangkan objek yang hendak di proses seperti objek manusia yang terdapat dalam gambar diberikan warna yang berbeda satu dengan yang lainnya.

### **2.4.4 Skeleton Algorithm**



Adalah cara untuk mengurangi ukuran dari sebuah bentuk atau pola dari sebuah objek pada gambar dengan jenis *binary* menjadi hanya direpresentasikan oleh piksel dengan jumlah tertentu saja, dalam hal ini adalah bentuknya diwakili oleh satu piksel masing-masing tetapi tetap mempertahan bentuk dan pola dari objek tersebut.

## **2.5 Penelitian Terdahulu**

Penelitian tentang cara mengukur panjang hama *pythoparasitic nematodes* yang di Indonesia lebih dikenal dengan nama *meloidogyne/radhopulus* telah dilakukan sebelumnya dengan metode lain.

### **2.5.1 An Algorithm to Extract Physical Characteristics of Nematodes from Microscopic Images of Plant Roots**

Peneletian yang dilakukan oleh Toribio dan koleganya dari Peru menemukan bahwa hama jenis ini menyerang dengan cara menempel pada akar tanaman buah-buahan dan menyerap *nutrient* dari tanaman dan sekaligus menghalangi tanaman untuk menyerap *nutrient* dari dalam tanah sehingga menyebabkan tanaman tidak dapat tumbuh sempurna dan menyebabkan kerugian yang besar. Mereka menggunakan Matlab dan Computer Vision Toolbox untuk mengukur panjang dari hama tersebut dan dapat mendeteksi panjang dari hama tersebut dengan keakuratan sampai 85% [4].

# **BAB 3**

# **METODOLOGI PENELITIAN**

## **3.1 Diagram Blok Sistem**





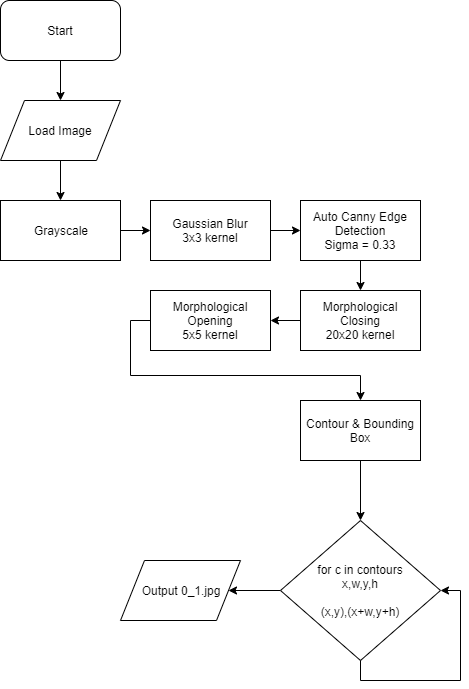
Gambar .1

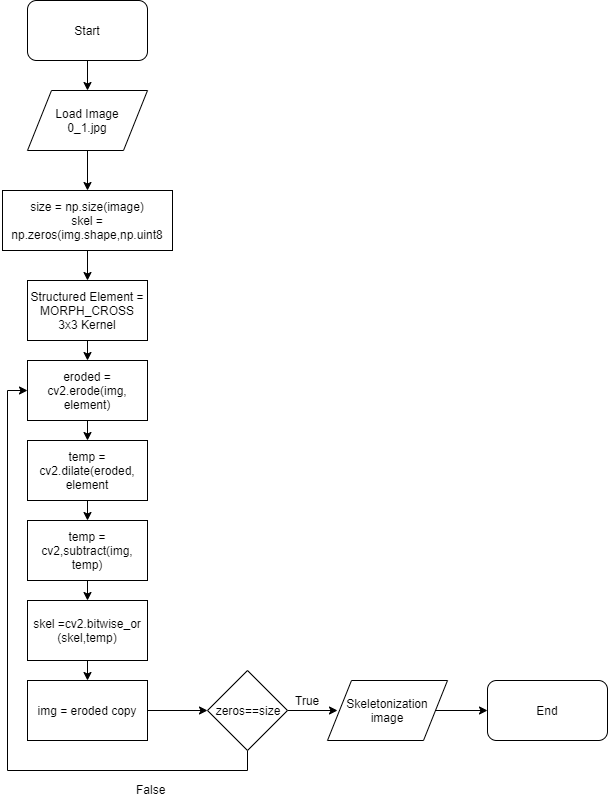
Langkah yang pertama kali dilakukan adalah pengambilan gambar *raw* dari hasil penelitian di laboratorium menggunakan mikroskop. Gambar mikroskop kemudian di foto menggunakan kamera smartphone. Gambar keluaran dari smartphone akan menghasilkan gambar berformat .png atau .jpeg.

Gambar kemudian akan diolah oleh OpenCV dengan pertama-tama mengubahnya menjadi gambar berformat *Grayscale*, untuk mempermudah pemrosesan oleh komputer. Kemudian gambar diproses kembali menjadi gambar *binary*, yaitu gambar hitam dan putih yang akan mempermudah program *computer vision* dalam mengatur *threshold* yang cocok untuk membedakan objek yang satu dengan yang lain.

Setelah mengatur threshold dan memisahkan objek yang satu dengan yang lain dengan Watershed Algorithm, maka *rotated bounding box* dan *contour* akan di tetapkan di objek, yaitu hama yang akan diteliti. Langkah terakhir adalah memberikan label kepada gambar tersebut. Label tersebut akan berisi informasi panjang piksel yang didapat dari *rotated bounding box* dan juga *contour*.

## **3.2 Diagram Flowchart**





Langkah pertama program akan menerima input gambar *raw* yang akan diteliti, kemudian gambar diubah menjadi gambar *grayscale* dan diberi *threshold* yang akan disesuaikan dengan kebutuhan untuk membuat gambar dapat membedakan intensitas perbedaan warna yang terdeteksi pada objek hama. Setelah melewati *threshold*, gambar kemudian akan berubah menjadi *Binary* yaitu hanya berwarna hitam dan putih dengan *outline* hama yang dapat dibedakan dengan objek lain yang bukan fokus penelitian. Kemudian gambar binary ini akan melewati tahap pemrosesan dengan algoritma Watershed sebanyak dua kali, untuk menghilangkan kemungkinan objek hama yang saling menempel dengan yang lain dan berbentuk tidak simetris/*irregular* agar dapat dibedakan satu sama lain.

Setelah pemrosesan dengan algoritma Watershed selesai, gambar kemudian akan dimasukkan kedalam *Rotated Bounding Box* dan akan diberi *contour* untuk mengikuti lekuk dan bentuk dari hama yang terdapat pada gambar. *Rotated bounding box* dan juga *contour* memiliki fungsi untuk mengeluarkan informasi pixel, berupa panjangnya, informasi ini akan sangat berguna untuk meneliti panjang sesungguhnya dari hama tersebut.

## **3.3 Instrumen Penelitian**

1. Sebuah laptop dengan prosessor Intel Core i5, RAM 12GB, dan Sistem Operasi Windows 10 1903
2. Python 3.7
3. Visual Studio Code
4. Anaconda 2019.07
5. OpenCV 4.1.1

# **BAB 4**

# **IMPLEMENTASI DAN ANALISIS**

## **4.1 Implementasi**

 Hal yang pertama dilakukan adalah gambar RAW dari sumber di load kedalam program kemudian dilakukan implementasi algoritma segmentation sebagai berikut. Gambar yang digunakan adalah gambar berikut.

### **Grayscale**



Image yang telah di *load* kedalam program akan di konversi menjadi gambar *grayscale*. Gambar *grayscale* adalah gambar dimana nilai piksel dari tiap-tiap nilai RGB atau BGR yaitu gambar berwarna diubah menjadi gambar abu-abu, dengan nilai value berkisar diantara 1, yaitu mewakili warna hitam, 255, yaitu warna putih, dan nilai tengah diantara value tersebut, yaitu 128, yang mewakili warna abu-abu.

Untuk mengkonversi sebuah gambar berwarna menjadi gambar *grayscale*, program akan mengambil warna dari masing-masing array RGB (R,G,B) pada piksel, dan mengaplikasikan algoritma Luma BT601 seperti dibawah ini.



Algoritma tersebut menyatakan bahwa tiap-tiap warna RGB mempunyai *luminance* yang berbeda-beda, dengan warna hijau mempunyai *luminance* yang lebih besar dibandingkan warna merah dan warna biru. Algoritma ini mengambil persentase *luminance* dari warna merah adalah 29.9%, warna hijau dengan persentase tertinggi 58.7%, dan warna biru dengan persentase 11.4%.

Setelah diaplikasikan algoritma ini dengan menggunakan *function* bawaan dari OpenCV, yaitu BGR2GRAY, hasil dari gambar awal menjadi seperti berikut ini.



### **Gaussian Blur**



Kemudian gambar grayscale tersebut akan ditambahkan efek Gaussian blur untuk objek apabila terdapat noise yang membuat OpenCV menganggap gambar objek adalah gambar terpisah. Dari kode diatas, gambar grayscale digunakan sebagai source dan digunakan ukuran kernel Gaussian 3x3.



### **Canny Edge Detection**



Gambar yang telah diberi efek Gaussian blur kemudian akan diubah menjadi gambar hitam putih untuk dapat memisahkan antara background, forground, dan object of interest, yaitu objek yang akan diteliti.

Hasil yang didapatkan dari gambar yang telah di lakukan *Gaussian blur/filter* kemudian ditambahkan algoritma thresholding/canny edge detection adalah seperti berikut ini.



Hasil yang didapatkan adalah gambar hama dengan background berwarna hitam dan objek hama yang didalamnya mengandung banyak sekali noise yang diakibatkan dari perbedaan warna hitam, putih, dan grayscale yang terdapat pada gambar tersebut. Terdapat pula noise diluar objek hama yaitu dua buah titik putih yang diakibatkan oleh masih terdeteksinya noise di background, hal yang akan dilakukan setelah langkah ini adalah untuk mengekstrak gambar objek menjadi hanya satu objek besar yang terdeteksi tanpa ada noise didalam objek hama ini.

### **Contour & Bounding Box**



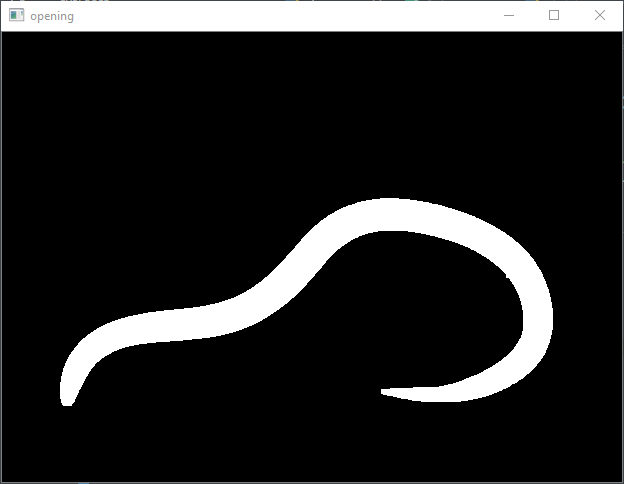
Digunakan untuk memberi contour, yaitu garis yang mengelilingi objek of interest dan juga sebuah bounding box yang mengelilingi seluruh objek of interest apabila contour tidak terlihat secara jelas dalam segmentasi.

### **Opening**



Digunakan untuk mengisi lubang-lubang atau noise yang terdapat dalam object of interest agar objek dapat sepenuhnya tersambung dan menjadi satu kesatuan.

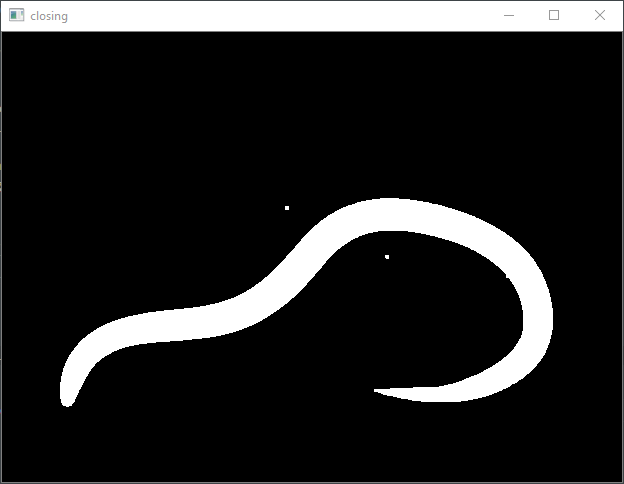
Dari hasil gambar yang didapatkan setelah dilakukan proses thresholding/canny edge detection, gambar hama tersebut mengandung noise dan mempunyai lubang-lubang yang, untuk itu dilakukanlah proses pengaplikasian opening, hasilnya adalah berikut ini.



### **Closing**



Digunakan untuk menutup lubang-lubang atau noise yang terdapat diluar object of interest supaya contour dan bounding box tidak mendeteksi objek lain diluar object of interest.



### **Skeletonization**



Gambar hasil dari closing kemudian akan disimpan dengan nama 0\_1.jpg dan dibuka di modul program lain untuk dilakukan proses skeletonization yang digunakan untuk mengekstrak bentuk dasar dari object of interest tersebut.



## **4.2 Analisis**

Hasil gambar yang telah dilakukan proses *skeletonization* kemudian memasuki tahap akhir untuk dilakukan perhitungan jumlah piksel putih yang terdapat pada gambar tersebut. Berikut adalah source code yang digunakan untuk menghitung jumlah piksel putih pada sebuah gambar.



Pertama kali yang dilakukan adalah, program meload gambar *skeleton*, lalu dibuat variable baru yang menyimpan *function* np.sum() dengan parameter yang digunakan adalah apabila gambar yang diload mempunyai piksel warna putih, yaitu valuenya sama dengan 255. Function np.sum() bekerja dengan cara menghitung semua elemen di array sesuai dengan apa yang ingin dihitung, dalam hal ini adalah jumlah piksel yang sama dengan 255. Kemudian setelah itu program mengeluarkan output berapa banyak jumlah piksel putih.

Dari jumlah piksel ini dapat dihitung berapa nilai yang diwakili oleh satu piksel pada gambar untuk mengetahui panjang hama, dalam hal ini satuannya adalah mikrometer atau µm. Besar ukuran yang diwakili oleh satu piksel dalam mikrometer atau micron dapat dihitung dengan menggunakan relasi formula sebagai berikut.

Dalam hal ini, diasumsikan mikroskop digital yang digunakan adalah Nikon TI2 series dengan spesifikasi sensor FF 36.0x23.9mm beresolusi 16.25 megapixel. Resolusi 16.25 megapixel dapat menghasilkan gambar dengan ukuran kira-kira 4045x4000 piksel, dengan menggunakan formula diatas ditemukan bahwa satu piksel pada gambar penelitian yang telah dilakukan proses *skeletonization* adalah 8.8 mikrometer atau µm.

Gambar penelitian yang digunakan adalah kumpulan gambar specimen dari berbagai macam jenis hama *meloidogyne* yang dikoleksi oleh Departemen Nematology Universitas Nebraska-Lincon. Hasil analisis dari pengaplikasian algoritma klasifikasi ini telah dikumpulkan pada tabel dibawah ini.



Analisis yang dapat disimpulkan dari tabel diatas adalah, spesimen dengan dimensi gambar yang memiliki resolusi kecil dan jumlah piksel yang sedikit menunjukkan keakuratan data diatas 50%. Perlu diketahui, bahwa masing-masing spesies dari hama meloidogyne ini sebenarnya memiliki ukuran standar yang berbeda-beda, tergantung tipe tanaman yang diinfeksi oleh hama ini, tipe juvenile atau J2 bisa saja memiliki nilai yang sedikit berbeda dengan tipe juvenile dari hama meloidogyne spesies lain, tetapi pada dasarnya Juvenile adalah fase termuda dari hama meloidogyne ini, fase ini dapat di identifikasi karena panjang hama adalah yang terpendek dibandingkan female, yang memiliki panjang diatas Juvenile/J2 dan male, yang memiliki panjang yang paling besar.

Seperti yang terjadi pada spesimen NM Par N dan NM Par 0, yang masing-masing memiliki resolusi 376x338 dan 401x310, menghasilkan keakuratan 97% dan 100% karena jumlah piksel yang dideteksi oleh program pendeteksi jumlah piksel memiliki nilai paling kecil dari spesimen lain. Hal ini diakibatkan oleh resolusi yang dimiliki gambar tersebut, semakin besar resolusi yang dimiliki maka semakin rapat dan semakin banyak jumlah piksel yang bisa dimuat, dan semakin kecil pula keakuratan yang bisa dicapai bila jumlah piksel yang dideteksi tersebut karena jumlah piksel akan dikalikan dengan nilai dari asumsi penggunaan mikroskop digital yaitu, satu piksel memiliki panjang pixel pitch sebesar 8.89 µm.

# **BAB 5**

# **KESIMPULAN DAN SARAN**

## **5.1 Kesimpulan**

Dari penelitian yang dilakukan pada subbab implementasi dan analisis dapat diperoleh kesimpulan bahwa penelitian ini dapat digunakan untuk menghitung panjang hama yang didapatkan hanya dari sebuah gambar, yang apabila berdasarkan rumusan masalah yang telah ditulis di sub bab pendahuluan di BAB 1 adalah sebagai berikut.

1. Hama meloidogyne memiliki berbagai macam jenis spesimen yang berbeda-beda tergantung dari jenis tanaman apa yang diinfeksi oleh hama tersebut, misalnya sebagai contoh terdapat satu spesimen meloidogyne dengan nama meloidogyne graminicola atau biasa dikenal dengan nama *rice root knot nematode* yaitu hama yang biasa menyerang tanaman padi. Perbedaan tentang spesimen meloidogyne satu dengan yang lainnya pada tiap-tiap klasifikasi di fase Juvenile/J2, Female, dan Male berpengaruh dengan beda panjang dari masing-masing spesimen, dan ini dapat berpengaruh juga pada hasil penelitian dan pengolahan gambar dengan OpenCV seperti analisis diatas.
2. Besar dari resolusi yang berbeda-beda dari berbagai gambar hama dapat berpengaruh pada hasil perhitungan jumlah piksel putih, dan ini mempengaruhi keakuratan data saat panjang hama yang diketahui dibandingkan dengan panjang hama dari hasil penelitian.
3. Keberagaman fase hama dari data gambar yang diperoleh dari sumber sangatlah tidak beragam, dari 22 gambar meloidogyne dan berbagai jenis spesimennya yang berbeda-beda, fase yang paling banyak adalah 21 gambar hama jenis Juvenile/J2 dan hanya 1 gambar hama di fase male.
4. Gambar yang diolah pada fase *skeletonization* dengan menggunakan algoritma dari fungsi cv2.MORPH\_CROSS dengan kernel matrix 3x3 membuat gambar yang telah menjadi skeleton memiliki garis bercabang pada ujung-ujungnya, dan ini dapat menambah jumlah piksel putih yang terdeteksi, walaupun tidak akan berpengaruh banyak terhadap hasil perhitungan panjangnya, karena antara fase Juvenile/J2, Female, dan Male terdapat perbedaan panjang 100-150µm.

## **5.2 Saran**

Berdasarkan dari penelitian yang dilakukan pada bab sebelumnya dan juga pada sub bab kesimpulan diatas, penulis dapat mengusulkan beberapa saran sebagai berikut.

1. Mengetahui mikroskop digital apa yang digunakan pada untuk mengambil gambar hama beserta spesifikasi lengkapnya.
2. Mengambil gambar dari satu sumber saja, misalnya dari laboratorium, agar mendapatkan gambar dengan resolusi dan posisi pengambilan gambar hama meloidogyne yang konsisten.

# **DAFTAR PUSTAKA**

|  |  |
| --- | --- |
| [1] | Britannica, "Pest vermin," [Online]. Available: https://www.britannica.com/science/pest-vermin. [Accessed 14 August 2019]. |
| [2] | M.-W. Dictionary, "Pest," [Online]. Available: https://www.merriam-webster.com/dictionary/pest. [Accessed 15 August 2019]. |
| [3] | "Science Daily," [Online]. Available: https://www.sciencedaily.com/releases/2019/03/190327142119.htm. [Accessed 13 August 2019]. |
| [4] | A. Toribio, L. Vargas, G. Kemper and A. Pamolo, "An Algorithm to Extract Physical Characteristics of Nematodes from Microscopic Images of Plant Roots," Greater Concepción, 2018. |
| [5] | OpenCV, "About," [Online]. Available: https://opencv.org/about/. [Accessed 13 August 2019]. |
| [6] | Python, "About," [Online]. Available: https://www.python.org/about/. [Accessed 13 August 2019]. |
| [7] | P. Chakravorty, "What Is a Signal? [Lecture Notes]," *What Is a Signal?,* vol. 35, no. 5, pp. 175-177, 2018. |
| [8] | L. G. Shapiro and G. C. Stockman, Computer Vision, New Jersey: Prentice-Hall, 2001. |
| [9] | Bargout, Lauren and L. W. Lee, Perceptual Information Processing System, Paravue Inc, 2003. |
| [10] | MATLAB, "Semantic Segmentation Basics," [Online]. Available: https://www.mathworks.com/help/vision/ug/semantic-segmentation-basics.html. [Accessed 14 August 2019]. |
| [11] | S. Mallick, "Image Segmentation," [Online]. Available: https://www.learnopencv.com/image-segmentation/. [Accessed 13 August 2019]. |
| [12] | wisc.edu, "What is Matlab," [Online]. Available: https://cimss.ssec.wisc.edu/wxwise/class/aos340/spr00/whatismatlab.htm. [Accessed 14 August 2019]. |
| [13] | OpenCV, "Watershed Segmentation," [Online]. Available: https://docs.opencv.org/master/d3/db4/tutorial\_py\_watershed.html. [Accessed 14 August 2019]. |