

**RANCANG BANGUN DETEKSI PENYAKIT LAYU
FUSARIUM PADA TANAMAN BAWANG MERAH
DENGAN PENGOLAHAN CITRA**

LAPORAN AKHIR



oleh

**FERNANDA TERESIA VENTURIN
NIM E32181959**

**PROGRAM STUDI TEKNIK KOMPUTER
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI JEMBER
2021**

**RANCANG BANGUN DETEKSI PENYAKIT LAYU
FUSARIUM PADA TANAMAN BAWANG MERAH
DENGAN PENGOLAHAN CITRA**

LAPORAN AKHIR



sebagai salah satu syarat untuk memperoleh gelar Ahli Madya Teknik (A.Md. T)
di Program Studi Teknik Komputer
Jurusan Teknologi Informasi

oleh

**FERNANDA TERESIA VENTURIN
NIM E32181959**

**PROGRAM STUDI TEKNIK KOMPUTER
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI JEMBER
2021**

**KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI JEMBER
JURUSAN TEKNOLOGI INFORMASI**

**RANCANG BANGUN DETEKSI PENYAKIT LAYU *FUSARIUM*
PADA TANAMAN BAWANG MERAH
DENGAN PENGOLAHAN CITRA**

Fernanda Teresia Venturin (E32181959)
Telat Diuji pada Tanggal 03 Agustus 2021
Telah Dinyatakan Memenuhi Syarat

Ketua Penguji,



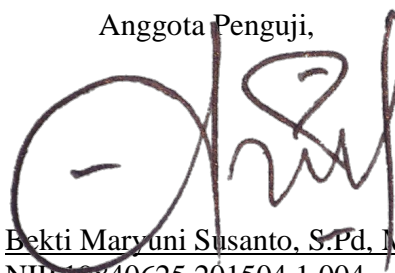
Denny Wijanarko, ST, MT
NIP 19780908 200501 1 001

Sekretaris Penguji/
Dosen Pembimbing,



Agus Hariyanto, ST, M.Kom
NIP 19780817 200312 1 005

Anggota Penguji,



Bekti Maryuni Susanto, S.Pd, M.Kom
NIP 19840625 201504 1 004

Menyetujui,
Ketua Jurusan Teknologi Informasi



Hendra Yuni Riskawan, S.Kom, M.Cs
NIP 19830203 200604 1 003



**PERNYATAAN
PERSETUJUAN PUBLIKASI KARYA
ILMIAH UNTUK KEPENTINGAN
AKADEMIS**

Yang bertandatangan di bawah ini, saya:

Nama : Fernanda Teresia Venturin
NIM : E32181959
Program Studi : Teknik Komputer
Jurusan : Teknologi Informasi

Demi pengembangan Ilmu Pengetahuan, saya menyetujui untuk memberikan kepada UPT.Perpustakaan Politeknik Negeri Jember, Hak Bebas Royalti Non-Eksklusif (Non-Exclusive Royalty Free Right) atas Karya Ilmiah berupa laporan Tugas Akhir saya yang berjudul:

**RANCANG BANGUN DETEKSI PENYAKIT LAYU *FUSARIUM* PADA
TANAMAN BAWANG MERAH DENGAN PENGOLAHAN CITRA**

Dengan Hak Bebas Royalti Non-Eksklusif ini UPT.Perpustakaan Politeknik Negeri Jember berhak menyimpan, mengalih media atau format, mengelola dalam bentuk Pangkalan Data (Database), mendistribusikan karya dan menampilkan atau mempublikasikannya di internet atau media lain untuk kepentingan akademis tanpa perlu meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis atau pencipta

Saya bersedia untuk menanggung secara pribadi tanpa melibatkan pihak Politeknik Negeri Jember, segala bentuk tuntutan hukum yang timbul atas Pelanggaran Hak Cipta dalam Karya Ilmiah ini.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Jember
Pada Tanggal : 15 Juni 2021
Yang Menyatakan



Nama : Fernanda Teresia Venturin
NIM : E32181959

MOTTO

“Setiap orang pasti memiliki tujuan hidup. Namun, yang membedakan satu orang dengan yang lainnya adalah sebesar apa kesungguhan mereka untuk menggapainya”

PERSEMBAHAN

Puji syukur kehadirat Allah SWT atas limpahan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan laporan tugas akhir yang berjudul Rancang Bangun Deteksi Penyakit Layu *Fusarium* Pada Tanaman Bawang Merah Dengan Pengolahan Citra.

Saya persembahkan laporan tugas akhir ini kepada :

1. Kedua Orang tua Bapak Purwanto, Ibu Tutik Tri Wulandari dan Adik Aura yang selalu mendoakan, menemani, memberikan semangat, dukungan materil maupun motivasi yang tiada hentinya.
2. Bapak Agus Hariyanto, ST, M. Kom selaku dosen pembimbing yang telah sabar membantu dan membimbing sehingga laporan tugas akhir ini dapat diselesaikan.
3. Bapak Denny Wijanarko, ST, MT selaku dosen penguji yang telah memberi masukan serta membimbing penyelesaian laporan tugas akhir.
4. Dosen, teknisi beserta jajarannya jurusan Teknologi Informasi khususnya Program Studi Teknik Komputer yang telah memberi banyak ilmu dan pengetahuan serta nasehat selama perkuliahan di Politeknik Negeri Jember.
5. Sahabat baik yang selalu ada saat suka maupun duka, selalu memberi dukungan Dewi Sulistyningrum, Kiki, Adit, Kristin, Firda, dan Nadya
6. Teman-teman seperjuangan TTK Angkatan 2018 yang telah memberi warna selama kuliah terutama golongan D.
7. Almamater tercinta Politeknik Negeri Jember.

RINGKASAN

RANCANG BANGUN DETEKSI PENYAKIT LAYU *FUSARIUM* PADA TANAMAN BAWANG MERAH DENGAN PENGOLAHAN CITRA ,
Fernanda Teresia Venturin, NIM E32181959, Tahun 2021, Teknologi Informasi
Politeknik Negeri Jember, Agus Hariyanto, ST, M.Kom. (Pembimbing)

Salah satu penyakit yang sering menyerang komoditas bawang merah adalah penyakit layu fusarium. Penyakit ini dapat menyerang sejak dari benih hingga pada fase pertumbuhan tanam. Belum ada implementasi pengolahan citra untuk mendeteksi penyakit layu fusarium yang dapat diimplementasikan pada sistem konveyor sebagai teknologi pendukung untuk petani komoditas bawang merah.

Sistem dirancang menggunakan raspberry pi yang terintegrasi dengan webcam. Raspberry pi ditempatkan pada sebuah box dengan display LCD dan memiliki lubang. Alat deteksi ini dikendalikan dengan sebuah tombol yang berfungsi untuk melakukan capture gambar ketika ada tanaman bawang yang dimasukkan pada box melalui lubang yang tersedia.

PRAKATA

Puji syukur penulis panjatkan kehadiran Allah SWT atas rahmat serta karunia Nya sehingga penulis dapat menyelesaikan laporan tugas akhir ini yang berjudul “Rancang Bangun Deteksi Penyakit Layu *Fusarium* Pada Tanaman Bawang Merah Dengan Pengolahan Citra ” diajukan sebagai salah satu syarat untuk menyelesaikan pendidikan di Politeknik Negeri Jember, Jurusan Teknologi Informasi, Program Studi Teknik Kompter.

Terselesaikannya laporan ini tidak terlepas dari bantuan beberapa pihak, oleh karena itu penulis mengucapkan banyak terimakasih kepada:

1. Direktur Politeknik Negeri Jember
2. Ketua Jurusan Teknologi Informasi
3. Ketua Program Studi Teknik Komputer
4. Seluruh staf pengajar di Program Studi Teknik Komputer.
5. Agus Hariyanto, ST, M.Kom.selaku Dosen Pembimbing
6. Dosen Penguji Tugas Akhir
7. Teman-teman dan semua pihak yang selalu memberi semangat serta masukan dalam menyelesaikan laporan ini.

Penulis berharap agar pembaca berkenan menyampaikan saran dan kritiknya dan semoga laporan ini dapat membawa manfaat kepada pembaca.

Jember,15 Juni 2021

Fernanda Teresia Venturin

E32181959

DAFTAR ISI

	halaman
HALAMAN JUDUL.....	ii
HALAMAN PENGESAHAN	iii
SURAT PERNYATAAN.....	iv
PERNYATAAN PERSETUJUAN PUBLIKASI	v
HALAMAN MOTTO.....	vi
HALAMAN PERSEMBAHAN	vii
RINGKASAN	viii
PRAKATA.....	ix
DAFTAR ISI.....	ix
DAFTAR GAMBAR.....	xii
DAFTAR TABEL.....	xiv
DAFTAR LAMPIRAN	xv
BAB 1. PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	1
1.3 Tujuan	1
1.4 Batasan Masalah	2
1.5 Manfaat	2
BAB 2. TINJAUAN PUSTAKA.....	3
2.1 Layu <i>Fusarium</i>	3
2.2 Definisi Citra Digital	4
2.3 Mikrokontroler	8
2.4 Python-Open CV	10
2.5 State Of The Art.....	14
BAB 3. METODE KEGIATAN.....	15
3.1 Waktu dan Tempat Pelaksanaan	15
3.2 Alat dan Bahan	15
3.3 Metode Kegiatan	16

3.4 Skenario Pengujian	19
3.5 Gambaran Sistem	20
3.6. Jadwal Kegiatan.....	23
BAB 4. IMPLEMENTASI DAN PENGUJIAN.....	24
4.1 Studi Literatur.....	24
4.2 Perancangan	24
4.3 Implementasi	27
4.4 Pengujian dan Analisa	36
BAB 5. KESIMPULAN DAN SARAN	50
5.1 Kesimpulan	50
5.2 Saran.....	50
DAFTAR PUSTAKA	51
LAMPIRAN.....	52

DAFTAR GAMBAR

Gambar 2.1 Layu Fusarium	4
Gambar 2.2 Representasi Citra Digital	5
Gambar 2.3 Representasi Citra RGB	6
Gambar 2.4 Nilai Intensitas Piksel.....	7
Gambar 2.5 Citra Hasil Konversi RGB.....	8
Gambar 2.6 Diagram Blok dan Struktur Mikrokontroler	8
Gambar 3.1 Gambar Alat tampak Depan	18
Gambar 3.2 Skema Deteksi Layu Fusarium.....	19
Gambar 3.3 Desain Box Casing.....	20
Gambar 3.4 Komponen Dudukan Bawang.....	21
Gambar 3.5 Rancangan Dudukan Bawang	21
Gambar 3.6 Flowchart.....	22
Gambar 4.1 Realisasi Perangkat Keras	24
Gambar 4.2 Perangkat Raspberry pi	25
Gambar 4.3 Rancangan Dudukan Bawang.....	25
Gambar 4.4 Implementasi Dasar Kamera	25
Gambar 4.5 Hasil Pembacaan.....	26
Gambar 4.6 Thresholding.....	26
Gambar 4.7 Pengujian SNR.....	27
Gambar 4.8 Pembacaan Program.....	29
Gambar 4.9 Hasil Pengujian	29
Gambar 4.10 Pembacaan Program	30
Gambar 4.12 Hasil Implementasi	31
Gambar 4.13 Pengambilan Gambar	32
Gambar 4.14 Hasil Implementasi	32
Gambar 4.15 Deteksi Fusarium	33
Gambar 4.16 Deteksi Fusarium	34
Gambar 4.17 Deteksi Daun	34
Gambar 4.18 Deteksi Bawang	35

Gambar 4.19	Deteksi Bawang Sehat atau Sakit	36
Gambar 4.20	Pengujian Akurasi Daun Layu.....	37
Gambar 4.21	Hasil Menampilkan Gambar.....	38
Gambar 4.22	Kondisi Daun Bawang	38
Gambar 4.23	Konversi Warna Daun.....	39
Gambar 4.24	Konversi Warna Daun.....	39
Gambar 4.25	Fungsi Kontur	40
Gambar 4.26	Deteksi Kontur.....	40
Gambar 4.27	Hasil Akhir Deteksi Kontur.....	41
Gambar 4.28	Pengujian Akurasi Bawang	42
Gambar 4.29	Hasil Masking.....	43
Gambar 4.30	Pengujian Akurasi Bawang	43
Gambar 4.31	Pengujian Keseluruhan Sistem	45
Gambar 4.32	Data Kondisi Daun.....	45
Gambar 4.33	Kondisi Bawang Sehat	46
Gambar 4.34	Implementasi nilai.....	46
Gambar 4.35	Pengujian Waktu Deteksi	48

DAFTAR TABEL

Tabel 2.1 State Of The Art	14
Tabel 4.1 Pengujian.....	27
Tabel 4.2 Angka Statistic	28
Tabel 4.3 Pengambilan Gambar.....	33
Tabel 4.4 Data Penelitian	41
Tabel 4.5 Data Penelitian	44
Tabel 4.6 Hasil Data.....	46
Tabel 4.7 Hasil Data Keseluruhan	48

DAFTAR LAMPIRAN

Lampiran 1. Program Python.....	52
Lampiran 2. Dokumentasi Pembuatan Alat	55

BAB 1. PENDAHULUAN

1.1. Latar Belakang

Salah satu penyakit yang sering menyerang komoditas bawang merah adalah penyakit layu fusarium. Penyakit ini dapat menyerang sejak dari benih hingga pada fase pertumbuhan tanam melalui media tanah. Apabila tanaman terserang penyakit layu fusarium dari benih, maka gejala awal akan terlihat pada saat tanaman berumur 5–10 hari setelah tanam. Jika penularan penyakit berasal dari tanah, maka gejala akan tampak pada saat tanaman berumur 3 minggu setelah tanam.

Belum ada implementasi pengolahan citra untuk mendeteksi penyakit layu fusarium yang dapat diimplementasikan pada sistem konveyor sebagai teknologi pendukung untuk petani komoditas bawang merah.

1.2. Rumusan Masalah

1. Bagaimana melakukan deteksi penyakit layu fusarium pada tanaman bawang merah?
2. Bagaimana mengimplementasikan program image processing untuk deteksi penyakit layu fusarium pada bawang merah?
3. Bagaimana merancang alat yang dapat mendeteksi penyakit layu fusarium pada tanaman bawang merah secara real time?

1.3. Tujuan

1. Merancang sistem deteksi penyakit layu fusarium pada tanaman bawang merah.
2. Merancang program image processing untuk mendeteksi penyakit layu fusarium pada tanaman bawang merah.
3. Merancang alat berbasis raspberry pi dan webcam untuk mendeteksi penyakit layu fusarium pada tanaman bawang merah.

1.4. Batasan Masalah

1. Sistem ini menggunakan deteksi pada daun dan bawang yang sample diambil di desa Banaran Kulon,kec.Bagor,kab.Nganjuk.
2. Penelitian tidak sampai diimplementasikan di konveyor namun hingga dapat mendeteksi pada sebuah box.
3. Penelitian menggunakan perangkat raspberry pi dan webcam.

1.5. Manfaat

1. Meningkatkan efisiensi produktivitas pertanian bawang merah.
2. Meningkatkan fungsi dari konveyor pada proses panen pertanian bawang merah.

BAB 2. TINJAUAN PUSTAKA

2.1 Layu *Fusarium*

Sesuai namanya, layu fusarium ini menunjukkan gejala layu pada tanaman yang sedang terserang. Layu fusarium sendiri merupakan penyakit yang disebabkan oleh infeksi jamur patogen *Fusarium oxysporum*. Layu fusarium bisa menyerang tanaman bawang merah kapan saja, terutama pada musim hujan seperti sekarang ini. Pada musim hujan jamur *Fusarium oxysporum* mudah berkembang biak dan mudah menyebar dari satu tanaman ketanaman lainnya. Tingkat kelembaban udara yang tinggi sangat berpengaruh terhadap perkembangbiakan jamur ini, terlebih lagi jika terjadi genangan air hujan dilahan dan pH tanah yang rendah. Sementara itu, inisiasi infeksi dari penyakit ini terjadi pada leher batang bagian bawah tanaman yang bersinggungan dengan tanah. Bagian tersebut membusuk dan berwarna cokelat. Infeksi menjalar ke akar sehingga mengalami busuk basah. Apabila kelembapan tanah cukup tinggi, bagian leher batang yang semula busuk kering tersebut berubah warna menjadi putih keabu-abuan karena terbentuk masa spora. Tidak hanya itu, serangan layu fusarium juga dapat menjalar pada bagian ranting tanaman dan berakhir pada layunya daun tanaman yang kemudian dapat menyebabkan kematian pada tanaman. Serangan layu fusarium sering sekali dijumpai pada tanaman baik usia muda maupun sudah dewasa. Gejala yang ditunjukkan yaitu tanaman akan tampak layu pada pukul 10.00-14.30 (selama siang hari) dan akan kembali tampak segar pada pagi serta sore hari selama proses fotosintesis berkurang. Sekilas, gejala ini mirip dengan layu bakteri namun bedanya adalah pada lamanya fase infeksi. Pada layu bakteri, tanaman akan langsung mati kering dalam 2-3 hari sedangkan layu fusarium akan tampak layu dan semakin parah hingga mati membutuhkan waktu sekitar 7-10 hari. Hingga saat ini memang belum ditemukan fungisida kimia yang benar-benar efektif untuk mengatasi serangan layu fusarium. Namun demikian jamur patogen ini tetap bisa dikendalikan populasinya pada lahan pertanian.

Gambar Layu Fusarium



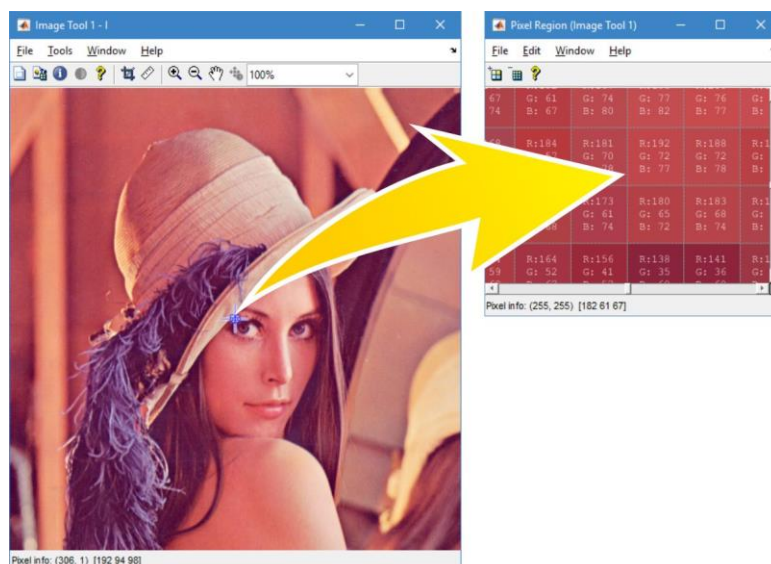
Gambar 2.1 Layu Fusarium

Sumber:<http://sumsel.litbang.pertanian.go.id/web/berita-penyakit-layu-fusarium-fusarium-oxysporum-pada-bawang-merah.html>

2.2 Definisi Citra Digital

Pengolahan Citra Digital (*Digital Image Processing*) merupakan bidang ilmu yang mempelajari tentang bagaimana suatu citra itu dibentuk, diolah, dan dianalisis sehingga menghasilkan informasi yang dapat dipahami oleh manusia. Sebelum mempelajari lebih lanjut mengenai pengolahan citra digital, kita perlu mengetahui definisi dari citra itu terlebih dahulu. Berdasarkan bentuk sinyal penyusunnya, citra dapat digolongkan menjadi dua jenis yaitu citra analog dan citra digital. Citra analog adalah citra yang dibentuk dari sinyal analog yang bersifat kontinyu, sedangkan citra digital adalah citra yang dibentuk dari sinyal digital yang bersifat diskrit. Citra analog dihasilkan dari alat akuisisi citra analog, contohnya adalah mata manusia dan kamera analog. Gambaran yang tertangkap oleh mata manusia dan foto atau film yang tertangkap oleh kamera analog merupakan contoh dari citra analog. Citra tersebut memiliki kualitas dengan tingkat kerincian (resolusi) yang sangat baik tetapi memiliki kelemahan di antaranya adalah tidak dapat disimpan, diolah, dan diduplikasi di dalam komputer. Citra digital merupakan representasi dari fungsi intensitas cahaya dalam bentuk diskrit pada bidang dua dimensi. Citra tersusun oleh sekumpulan piksel (*picture element*) yang memiliki koordinat (x,y) dan amplitudo $f(x,y)$.

Koordinat (x,y) menunjukkan letak/posisi piksel dalam suatu citra, sedangkan amplitudo $f(x,y)$ menunjukkan nilai intensitas warna citra. Representasi citra digital beserta piksel penyusunnya ditunjukkan pada Gambar 1 berikut ini.



Gambar 2.2 Representasi Citra Digital

Sumber: https://id.wikipedia.org/wiki/Pengolahan_citra_digital

Pada umumnya, berdasarkan kombinasi warna pada piksel, citra dibagi menjadi tiga jenis yaitu citra RGB, citra grayscale, dan citra biner. Citra pada Gambar 1 termasuk dalam jenis citra *RGB truecolor 24-bit*. Citra tersebut tersusun oleh tiga kanal warna yaitu kanal merah, kanal hijau, dan kanal biru.

Masing-masing kanal warna memiliki nilai intensitas piksel dengan kedalaman bit sebesar 8-bit yang artinya memiliki variasi warna sebanyak 2^8 derajat warna (0 s.d 255). Pada kanal merah, warna merah sempurna direpresentasikan dengan nilai 255 dan hitam sempurna dengan nilai 0. Pada kanal hijau, warna hijau sempurna direpresentasikan dengan nilai 255 dan hitam sempurna dengan nilai 0. Begitu juga pada kanal biru, warna biru sempurna direpresentasikan dengan nilai 255 dan hitam sempurna dengan nilai 0.

Perintah MATLAB untuk menampilkan citra digital dan masing-masing kanal penyusunnya adalah sebagai berikut:

```

1      clc; clear; close all; warning off all;
2
3      I = imread('lena_color_256.tif');
4      Red = I(:,:,1);
5      Green = I(:,:,2);
6      Blue = I(:,:,3);
7      I_Red = cat(3,Red,Green*0,Blue*0);
8      I_Green = cat(3,Red*0,Green,Blue*0);
9      I_Blue = cat(3,Red*0,Green*0,Blue);
10
11     figure, imshow(I);
12     figure, imshow(I_Red);
13     figure, imshow(I_Green);
14     figure, imshow(I_Blue);

```

Representasi citra RGB dan masing-masing kanal warna penyusunnya ditunjukkan pada Gambar 2.3



Gambar 2.3 Representasi Citra RGB

Sumber:<https://pemrogramanmatlab.files.wordpress.com/2017/07/citra-rgb-dan-kanal-warna-rgb.jpg>

Setiap piksel pada citra RGB, memiliki intensitas warna yang merupakan kombinasi dari tiga nilai intensitas pada kanal R, G, dan B. Sebagai contoh, suatu piksel yang memiliki nilai intensitas warna sebesar 255 pada kanal merah, 255 pada kanal hijau, dan 0 pada kanal biru akan menghasilkan warna kuning. Pada contoh lain, suatu piksel yang memiliki nilai intensitas warna sebesar 255 pada kanal merah, 102 pada kanal hijau, dan 0 pada kanal biru akan menghasilkan warna orange. Banyaknya kombinasi warna piksel yang mungkin pada citra RGB truecolor 24-bit adalah sebanyak $256 \times 256 \times 256 = 16.777.216$. Representasi nilai intensitas piksel dengan kombinasi warna R, G, dan B ditunjukkan pada Gambar 2.4

Yellow R = 255 G = 255 B = 0	Orange R = 255 G = 102 B = 0	Green R = 0 G = 255 B = 0
Cyan R = 0 G = 255 B = 255	Violet R = 204 G = 102 B = 204	White R = 255 G = 255 B = 255
Black R = 0 G = 0 B = 0	Turquoise R = 102 G = 255 B = 204	Brown R = 153 G = 102 B = 51

Gambar 2.4 Nilai Intensitas Piksel

Sumber: <https://pemrogramanmatlab.files.wordpress.com/2017/07/citra-rgb-dan-kanal-warna-rgb.jpg>

Gambar 2.3 Representasi piksel dengan kombinasi warna R, G, dan B

Jenis citra yang kedua adalah citra grayscale. Citra grayscale merupakan citra yang nilai intensitas pikselnya didasarkan pada derajat keabuan. Pada citra grayscale 8-bit, derajat warna hitam sampai dengan putih dibagi ke dalam 256 derajat keabuan di mana warna hitam sempurna direpresentasikan dengan nilai 0 dan putih sempurna dengan nilai 255. Citra RGB dapat dikonversi menjadi citra grayscale sehingga dihasilkan hanya satu kanal warna. Persamaan yang umumnya digunakan untuk mengkonversi citra RGB truecolor 24-bit menjadi citra grayscale 8-bit adalah $Grayscale = 0.2989 * R + 0.5870 * G + 0.114$

Citra hasil konversi RGB menjadi grayscale ditunjukkan pada Gambar 4.



Gambar 2.5 Citra Hasil Konversi RGB

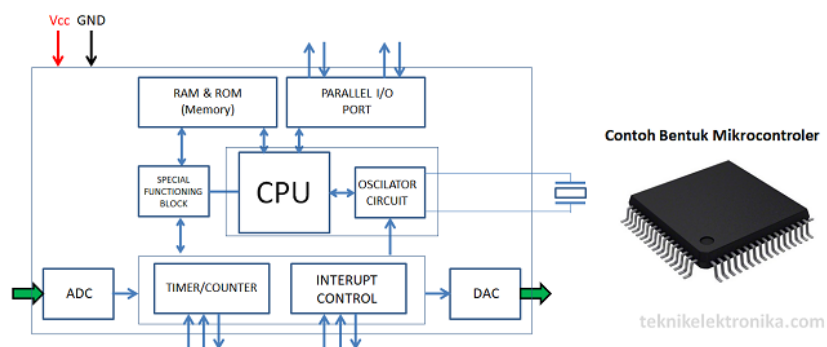
Sumber:<https://pemrogramanmatlab.files.wordpress.com/2017/07/citra-rgb-dan-kanal-warna-rgb.jpg>

2.4 Mikrokontroler

Pengertian Mikrokontroler (MicroController) dan Strukturnya – Mikrokontroler adalah sebuah komputer kecil yang dikemas dalam bentuk chip IC (Integrated Circuit) dan dirancang untuk melakukan tugas atau operasi tertentu. Pada dasarnya, sebuah IC Mikrokontroler terdiri dari satu atau lebih Inti Prosesor (CPU), Memori (RAM dan ROM) serta perangkat INPUT dan OUTPUT yang dapat diprogram.

Diagram Blok dan Struktur Mikrokontroler

Berikut ini adalah Diagram Blok dan Struktur Mikrokontroler beserta penjelasan singkat tentang bagian-bagian utamanya.



Gambar 2.6 Diagram Blok dan Struktur Mikrokontroler

Sumber:<https://teknikelektronika.com/wp-content/uploads/2020/03/Pengertian-Mikrokontroler-Microcontroller.png?x91019>

1. CPU

CPU adalah otak mikrokontroler. CPU bertanggung jawab untuk mengambil instruksi (fetch), menerjemahkannya (decode), lalu akhirnya dieksekusi (execute). CPU menghubungkan setiap bagian dari mikrokontroler ke dalam satu sistem. Fungsi utama CPU adalah mengambil dan mendekode instruksi. Instruksi yang diambil dari memori program harus diterjemahkan atau melakukan decode oleh CPU tersebut.

2. Memori (Penyimpanan)

Fungsi memori dalam mikrokontroler sama dengan mikroprosesor. Memori ini digunakan untuk menyimpan data dan program. Sebuah mikrokontroler biasanya memiliki sejumlah RAM dan ROM (EEPROM, EPROM dan lain-lainnya) atau memori flash untuk menyimpan kode sumber program (source code program).

3. Port INPUT / OUTPUT paralel

Port Input / Output paralel digunakan untuk mendorong atau menghubungkan berbagai perangkat seperti LCD, LED, printer, memori dan perangkat INPUT/OUTPUT lainnya ke mikrokontroler.

4. Port Serial (Serial Port)

Port serial menyediakan berbagai antarmuka serial antara mikrokontroler dan periferal lain seperti port paralel.

5. Pengatur Waktu dan Penghitung (Timer dan Counter)

Timer dan Counter adalah salah satu fungsi yang sangat berguna dari Mikrokontroler. Mikrokontroler mungkin memiliki lebih dari satu timer dan counter. Pengatur waktu (Timer) dan Penghitung (Counter) menyediakan semua fungsi pengatur waktu dan penghitungan di dalam mikrokontroler. Operasi utama yang dilakukan di bagian ini adalah fungsi jam, modulasi, pembangkitan pulsa, pengukuran frekuensi, osilasi, dan lain sebagainya. Bagian ini juga dapat digunakan untuk menghitung pulsa eksternal.

6. Analog to Digital Converter atau Pengonversi Analog ke Digital (ADC)

Konverter ADC digunakan untuk mengubah sinyal analog ke bentuk digital. Sinyal input dalam konverter ini harus dalam bentuk analog (misalnya

Output dari Sensor) sedangkan Outputnya dalam bentuk digital. Output digital dapat digunakan untuk berbagai aplikasi digital seperti layar digital pada Perangkat pengukuran.

7. Digital to Analog Converter atau Pengonversi Digital ke Analog (DAC)

DAC melakukan operasi pembalikan konversi ADC. DAC mengubah sinyal digital menjadi format analog. Ini biasanya digunakan untuk mengendalikan perangkat analog seperti motor DC dan lain sebagainya.

8. Kontrol Interupsi (Interrupt Control)

Kontrol interupsi atau Interrupt Control digunakan untuk menyediakan interupsi (penundaan) untuk program kerja. Interrupt dapat berupa eksternal (diaktifkan dengan menggunakan pin interrupt) atau internal (dengan menggunakan instruksi interupsi selama pemrograman).

9. Blok Fungsi Khusus (Special Functioning Block)

Beberapa Mikrokontroler yang hanya dapat digunakan untuk beberapa aplikasi khusus (misalnya sistem Robotik), pengontrol ini memiliki beberapa port tambahan untuk melakukan operasi khusus tersebut yang umumnya dinamakan dengan Blok Fungsi Khusus.

1.1 Python-Open CV

2.4.1 Python

Python adalah bahasa pemrograman tujuan umum yang ditafsirkan, tingkat tinggi. Dibuat oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991, filosofi desain Python menekankan keterbacaan kode dengan penggunaan spasi putih yang signifikan. Konstruksi bahasanya dan pendekatan berorientasi objek bertujuan untuk membantu programmer menulis kode yang jelas dan logis untuk proyek skala kecil dan besar.

Python diketik secara dinamis dan pengumpulan sampah. Ini mendukung beberapa paradigma pemrograman, termasuk pemrograman terstruktur (terutama, prosedural), berorientasi objek, dan fungsional. Python sering dideskripsikan sebagai bahasa "termasuk baterai" karena perpustakaan standarnya yang komprehensif.

Python dikandung pada akhir 1980-an sebagai penerus bahasa ABC. Python 2.0, dirilis pada tahun 2000, memperkenalkan fitur-fitur seperti pemahaman daftar dan sistem pengumpulan sampah dengan penghitungan referensi.

Python 3.0, dirilis pada tahun 2008, adalah revisi utama dari bahasa yang tidak sepenuhnya kompatibel dengan versi sebelumnya, dan banyak kode Python 2 yang tidak berjalan tanpa modifikasi pada Python 3.

Penerjemah Python tersedia untuk banyak sistem operasi. Komunitas programmer global mengembangkan dan memelihara CPython, implementasi referensi^[30] yang gratis dan bersumber terbuka. Sebuah organisasi nirlaba, Python Software Foundation, mengelola dan mengarahkan sumber daya untuk pengembangan Python dan CPython.

2.4.2 **OpenCV** (*Open Source Computer Vision Library*)

OpenCV (Open Source Computer Vision Library) adalah sebuah pustaka perangkat lunak yang ditujukan untuk pengolahan citra dinamis secara real-time, yang dibuat oleh Intel, dan sekarang didukung oleh Willow Garage dan Itseez. Program ini bebas dan berada dalam naungan sumber terbuka dari lisensi BSD. Pustaka ini merupakan pustaka lintas platform. Program ini didedikasikan sebagian besar untuk pengolahan citra secara real-time. Jika pustaka ini menemukan pustaka Integrated Performance Primitives dari intel dalam sistem komputer, maka program ini akan menggunakan rutin ini untuk mempercepat proses kerja program ini secara otomatis. OpenCV pertama kali diluncurkan secara resmi pada tahun 1999 oleh Inter Research sebagai lanjutan dari bagian proyek bertajuk aplikasi intensif berbasis CPU, real-time ray tracing dan tembok penampil 3D. Para kontributor utama dalam proyek ini termasuk mereka yang berkecimpung dalam bidang optimasi di Intel Russia, dan juga Tim Pustaka Performansi Intel. Pada awalnya, tujuan utama dari proyek OpenCV ini dideskripsikan sebagai berikut,

Penelitian penginderaan citra lanjutan tidak hanya melalui kode program terbuka, tetapi juga kode yang telah teroptimasi untuk infrastruktur penginderaan citra.

Menyebarkan ilmu penginderaan citra dengan menyediakan infrastruktur bersama di mana para pengembang dapat menggunakannya secara bersama-sama, sehingga kode akan tampak lebih mudah dibaca dan ditransfer.

Membuat aplikasi komersial berbasis penginderaan citra, di mana kode yang telah dioptimasi tersedia secara bebas dengan lisensi yang tersedia secara bebas yang tidak mensyaratkan program itu harus terbuka atau gratis.

2.4.3 Contoh Program OpenCV

1. Membaca Gambar

Untuk membaca gambar dalam OpenCV menggunakan fungsi `cv2.imread()` dimana parameter pertama adalah nama file gambar lengkap dengan ekstensinya.

Sebagai contoh :

Fungsi `imread()` OpenCVPython

```
import cv2
img = cv2.imread('foo.jpg')
# Opsional
cv2.waitKey(0)
cv2.destroyAllWindows()
1
2
3
4
5
import cv2
img = cv2.imread('foo.jpg')
# Opsional
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Fungsi `cv2.waitKey(0)` adalah untuk mempertahankan window agar tetap menampilkan gambar. Sedangkan fungsi `cv2.destroyAllWindows()` adalah untuk menutup window lain yang sedang terbuka.

2. Menampilkan Gambar

Menampilkan gambar dalam OpenCV menggunakan fungsi `cv2.imshow()` dengan parameter pertama adalah nama window untuk menampilkan gambar dan parameter kedua adalah gambar itu sendiri. Contoh :

```
Fungsi imshow() OpenCVPython
import cv2
img = cv2.imread('foo.jpg')
cv2.imshow('Menampilkan Gambar', img)
1
2
3
import cv2
img = cv2.imread('foo.jpg')
cv2.imshow('Menampilkan Gambar', img)
```

3. Menulis / Menyimpan Gambar

Untuk menulis / menyimpan gambar dalam OpenCV menggunakan fungsi `cv2.imwrite()` dimana parameter pertama adalah nama file baru yang akan kita simpan dan parameter kedua adalah sumber gambar itu sendiri. Contoh :

```
Fungsi imwrite() OpenCV
import cv2
img = cv2.imread('foo.jpg')
cv2.imwrite('bar.jpg', img)
1
2
3
import cv2
img = cv2.imread('foo.jpg')
cv2.imwrite('bar.jpg', img)
```

2.5 State Of The Art

Berdasarkan dari ketiga penelitian tugas akhir ini memiliki perbedaan dan persamaan yang terdapat pada tabel berikut ini.

Tabel 2.1 State Of The Art

Penulis	Sreenivasa, M. Y., et al	Lim, Hadrian Paulo M., and Maria Regina Justina E. Estuar	Fernanda Teresia
Tema	Deteksi Fusarium	Deteksi Fusarium	Deteksi Fusarium
Judul	Molecular detection of fumonisin producing Fusarium species of freshly harvested maize kernels using polymerase chain reaction (PCR)."	Microscopic fusarium detection and verification with convolutional neural networks	Rancang bangun deteksi penyakit layu fusarium pada tanaman bawang merah dengan pengolahan citra
studi kasus	Spesies Fusarium	Microscopic Fusarium	Tanaman bawang merah
Metode	PCR	Convolutional neural network	Image Processing
Aplikasi	PCR Lab	Computer	Raspberry Pi
Tahun	2006	2018	2021

Berdasarkan isi dari ketiga karya tulis tersebut yaitu sama-sama mengambil tema tentang deteksi layu fusarium. Namun yang membedakan antara karya tulis ilmiah yang satu dengan yang lainnya terletak pada metode atau alat yang digunakan

BAB 3. METODE KEGIATAN

3.1 Waktu dan Tempat Pelaksanaan

3.1.1 Tempat Pelaksanaan

Pelaksanaan tugas akhir dengan judul “Rancang Bangun Deteksi Layu Fusarium Pada Tanaman Bawang Merah dengan Pengolahan citra” dilaksanakan di rumah penulis di desa Banaran Kulon kecamatan Bagor Kabupaten Nganjuk.

3.2.2 Waktu Pelaksanaan

Pelaksanaan tugas akhir ini dilaksanakan dalam kurun waktu kurang lebih 7 bulan, dimulai bulan Januari 2021 sampai dengan bulan Juli 2021

3.2 Alat dan Bahan

Adapun bahan dan Alat yang digunakan untuk mendeteksi Layu Fusarium pada tanaman bawang merah ini adalah sebagai berikut:

Alat-alat yang dibutuhkan:

NO	JENIS	SPESIFIKASI
1	Laptop	Windows 10 Pro 64 Bit
2	Solder	40 Watt
3	Gunting	-
4	Obeng	-
5	Tang Potong	-
6	Multimeter	-
7	Kabel LAN	1 meter
8	bor listrik	-
9	double tape	3M

Adapun kebutuhan perangkat lunak yang dipergunakan dalam penelitian ini:

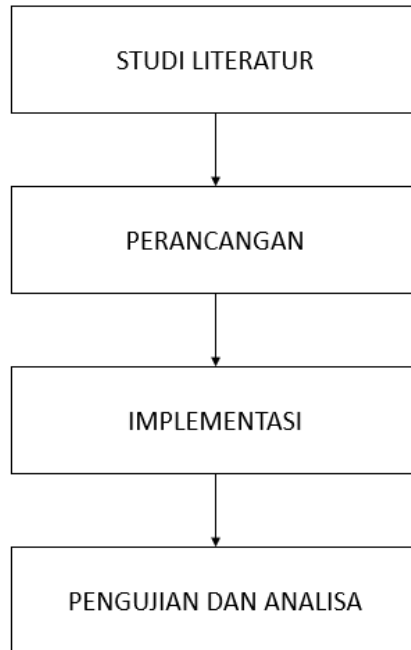
NO	JENIS	SPESIFIKASI
1	VNC Viewer	Versi 6.0.1
2	Geany IDE	Aplikasi pemrograman Python
3	Python	Program interpreter hardware dan citra digital versi 2.7
4	Library OpenCV	Library open source computer vision
5	Library I2C LCD	Library pengatur display LCD berbasis komunikasi I2C
6	Sketch Up	3d designer

Untuk merealisasikan penelitian ini digunakan bahan-bahan antara lain:

NO	JENIS	JUMLAH DAN KETERANGAN
1	Box	1 Set Acrylic 3mm 20cm x 20 cm x 20cm
2	Web Cam	1 Buah , merk Logitech C270 - 720p
3	Raspberry Pi	1 Buah - Generasi 3
4	LCD	1 buah - Blue Light 16 x 2
5	Kabel jumper	30cm x 40 set
6	LED	Putih 1/4 Watt 2 buah
7	Resistor	330 Ohm 1/4 Watt 2 buah
8	Adaptor	5V 2A USB Socket
9	mur baut	3mm 8 set
10	lem akrilik	-
11	i2c driver	driver LCD berbasis komunikasi i2c - 1 buah
12	push button	-
13	Engsel	-
14	dudukan bawang	3d print - 1 set
15	cable ties	20cm - hitam 2 set

3.3 Metode kegiatan

Metode dalam pelaksanaan tugas akhir ini terdiri dari beberapa tahap antara lain studi literatur, perancangan, implementasi, pengujian dan Analisa. Tahapan metode penelitian ini digambarkan melalui diagram berikut:

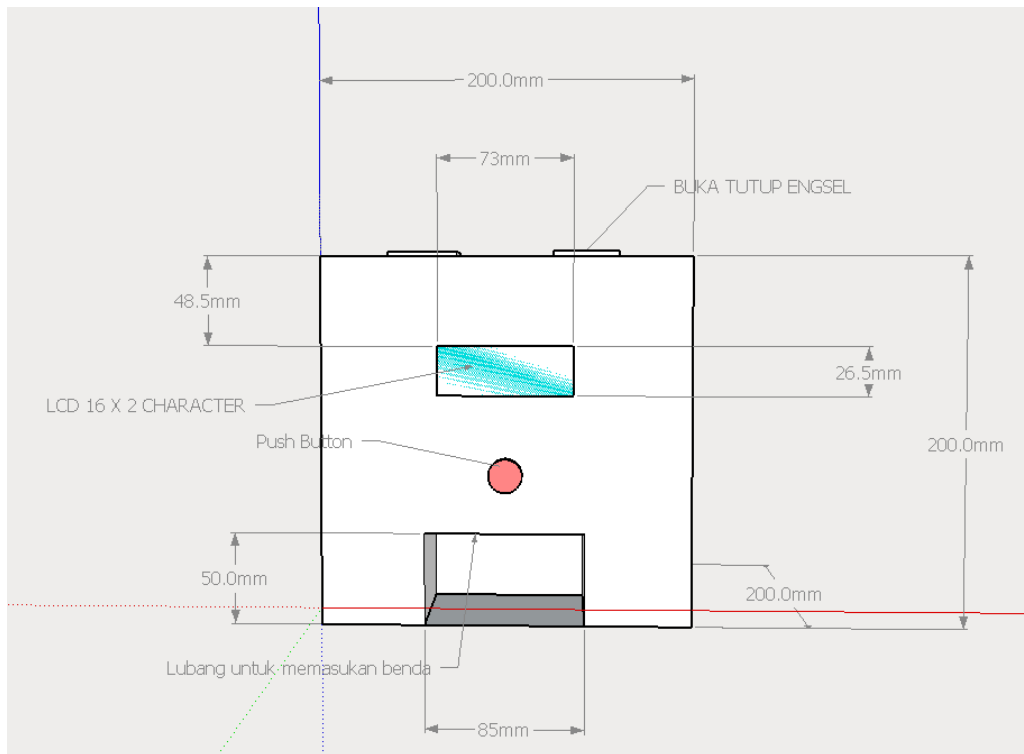


3.3.1 Studi Literatur

Studi literatur dilakukan penulis untuk mengetahui aspek aspek teknologi dan penelitian terkait yang dapat memenuhi tujuan dari penelitian. Pada penelitian ini dikaji berbagai referensi dari buku jurnal, paper, laporan penelitian artikel dan berbagai situs di internet. Melalui kajian studi literatur ini diharapkan diperoleh informasi yang mendukung penyelesaian rumusan masalah dan dapat merealisasikan luaran penelitian sesuai dengan tujuan penelitian.

3.3.2 Perancangan

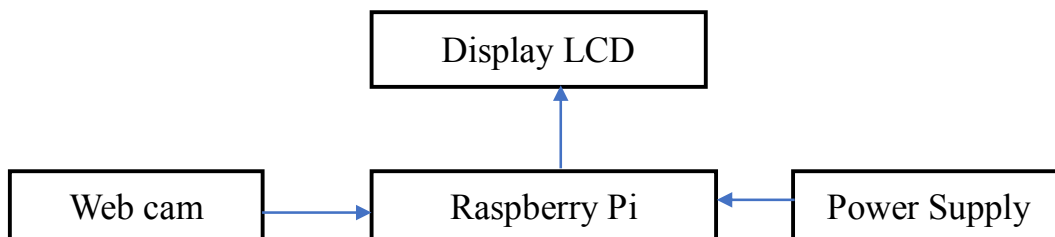
Perancangan perangkat dilakukan untuk mempermudah proses penelitian sehingga dapat memberikan gambaran hasil akhir sesuai rencana dan tujuan penelitian. Untuk memenuhi kebutuhan penelitian ini, dilakukan perancangan perangkat keras secara 3 dimensi menggunakan aplikasi sketch up sehingga diperoleh rancangan tampak depan sebagai berikut:



Gambar 3.1 Gambar Alat Tampak Depan

Sistem dirancang menggunakan raspberry pi yang terintegrasi dengan webcam. Raspberry pi ditempatkan pada sebuah box dengan display LCD dan memiliki lubang. Alat deteksi ini dikendalikan dengan sebuah tombol yang berfungsi untuk melakukan *capture* gambar. Ketika ada tanaman bawang yang dimasukkan pada box melalui lubang yang tersedia.

Adapun gambaran umum cara kerja perangkat ini digambarkan melalui diagram blok pada gambar 3.2. Pada diagram ini digunakan webcam sebagai input kepada raspberry pi untuk membaca informasi citra digital pada tanaman bawang merah yang terdiri dari daun dan umbi. Sistem dirancang untuk menampilkan instruksi juga hasil melalui LCD yang dirancang menggunakan LCD 16 x 2 character. Power supply digunakan untuk menyalakan sistem secara keseluruhan.



Gambar 3.2 Skema Deteksi Layu Fusarium

3.3.3. Implementasi

Pada Implementasi ini ,diuji hasil tampilan display untuk menampilkan Bawang Sakit atau Bawang Sehat.

3.3.4. Pengujian dan Analisa

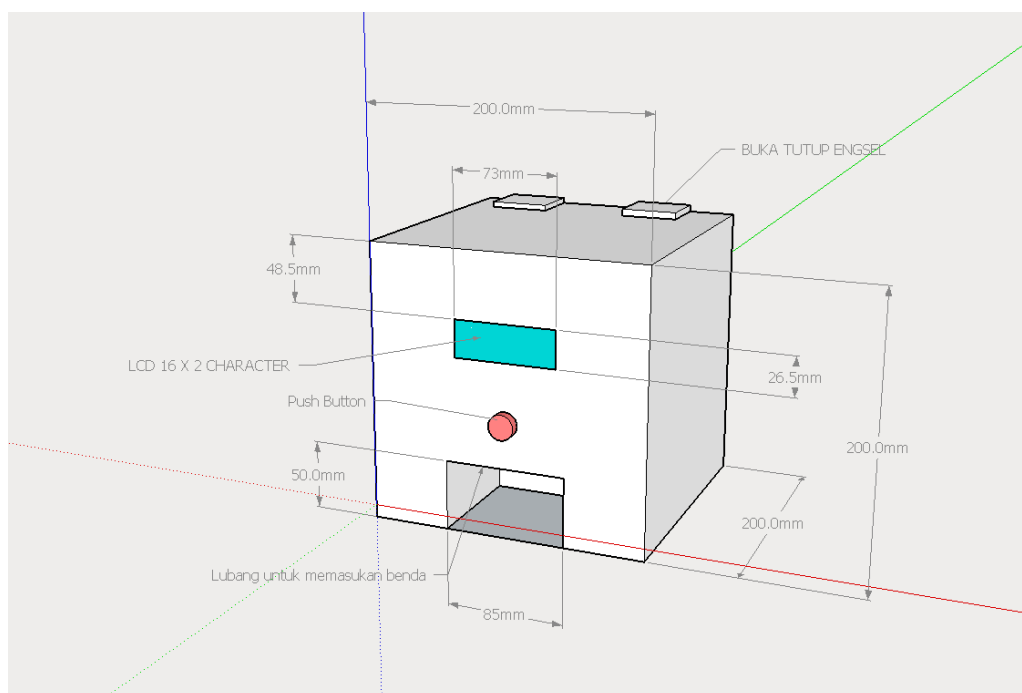
Tahap pengujian akan dilakukan dengan cara.Tujuannya untuk menentukan kondisi bawang mengalami layu fusarium dengan mendeteksi apakah daun layu dan bawang layu.

3.4 Skenario Pengujian

- Mengambil SNR dari webcam
- Tujuan pengujian ini untuk mengetahui kualitas noise dari webcam yang digunakan sehingga menjadi dasar kualitas dari proses selanjutnya.
- Uji Fungsi pengambilan gambar dengan button
- Tujuan pengujian ini untuk mengetahui kualitas fungsi button pada sistem apakah dapat berfungsi dengan baik atau tidak
- Uji Fungsi Display
- Tujuan pengujian ini untuk mengetahui kualitas fungsi display pada sistem apakah dapat berfungsi dengan baik atau tidak
- Akurasi Deteksi daun layu
- Tujuan pengujian ini untuk mengetahui akurasi pendeteksian daun layu
- Akurasi Deteksi umbi busuk

- Tujuan pengujian ini untuk mengetahui akurasi pendeteksian umbi busuk
- Akurasi Keseluruhan
- Tujuan pengujian ini untuk mengetahui akurasi pendeteksian layu fusarium
- Pengujian waktu deteksi
- Tujuan pengujian ini untuk mengetahui berapa lama waktu yang dibutuhkan untuk melakukan setiap proses pendeteksian

3.5 Gambaran Sistem



Gambar 3.3 Desain BOX-Casing

Pada penelitian ini dirancang sebuah box untuk memenuhi kebutuhan sistem agar dapat menerima input bawang merah dan daun melalui sebuah lubang dengan dimensi 50mm x 85 mm.

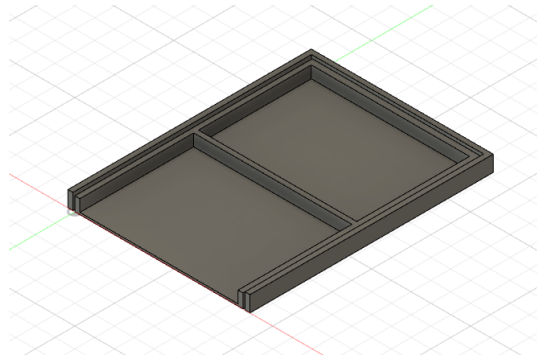
Lubang push button disediakan yang berfungsi sebagai input perintah kemudian lubang LCD 16 x 2 dirancang sesuai dimensi yaitu 73mm x 26.5mm . box ini dirancang dengan penutup yang dapat dibuka melalui 2 buah engsel. Secara

keseluruhan box ini berdimensi 200mm x 200mm x 200mm yang dirancang dengan material berbahan akrilik hitam dengan tebal 3mm.

Desain Dudukan Input

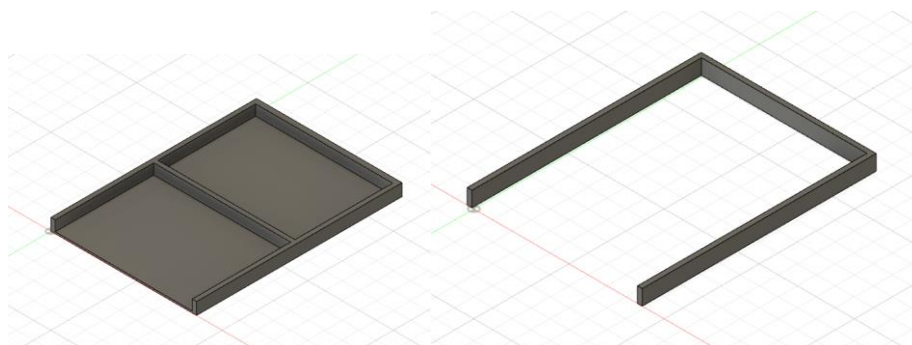
Desain dudukan input adalah dudukan yang disediakan untuk memasukan bawang merah dan daun agar dapat diposisikan sesuai area pembacaan kamera. Perancangan ini dilakukan dengan perangkat lunak fusion 360 agar dapat dikonversi menjadi file dengan ekstensi STL yang pada implementasinya akan direalisasikan menggunakan proses pencetakan 3d print dengan bahan PLA+ berwarna putih.

Dimensi yang digunakan pada desain ini sebesar 123mm x 92 mm.



Gambar 3.4 Desain Dudukan Input

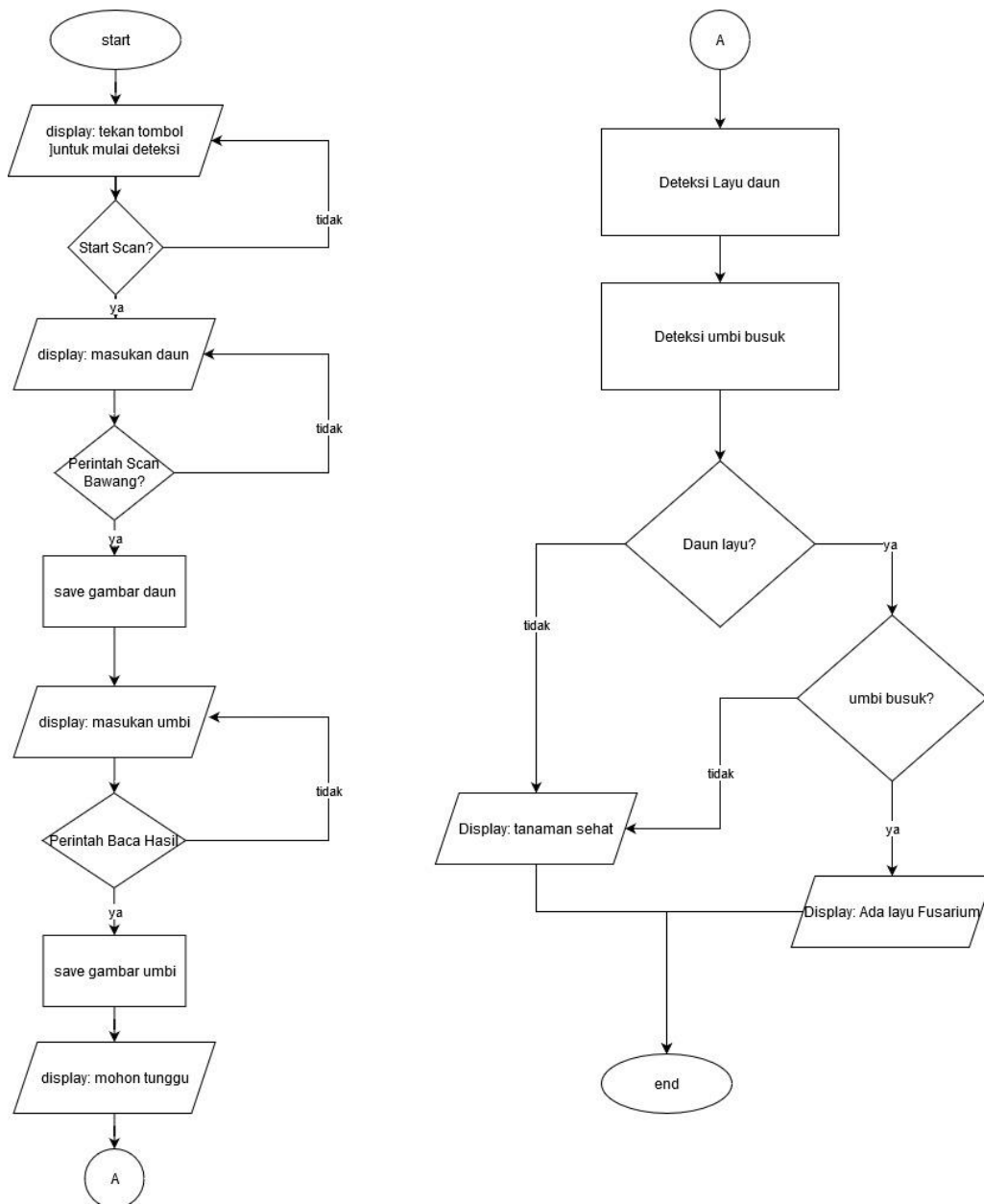
Perancangan dudukan ini terdiri dari 2 buah komponen yaitu dudukan bawang yang dapat berfungsi keluar masuk box dan bagian penahan yang dipatenkan di bagian dalam box.



Gambar 3.5 Dudukan Bawang

Kiri rancangan dudukan bawang, kanan rancangan penahan dudukan

3.5.1. Flowchart



Gambar 3.6 Flowchart

3.6. Jadwal Kegiatan

Jadwal kegiatan yang akan dilakukan sesuai dengan langkah-langkah metode kegiatan yang telah ditentukan seperti pada Tabel 3.1 berikut.

Tabel 3.1 Jadwal Kegiatan

Jenis Kegiatan	Bulan Ke -						
	1	2	3	4	5	6	7
Studi Literatur	■	■					
Perancangan		■	■				
Implementasi			■	■	■		
Pengujian dan Analisa						■	■

BAB 4. HASIL DAN PEMBAHASAN

4.1. Studi Literatur

Studi literatur dilakukan untuk mengetahui aspek aspek teknologi dan penelitian terkait yang dapat memenuhi tujuan dari penelitian. Pada penelitian ini dikaji berbagai referensi dari buku jurnal, paper, laporan penelitian artikel dan berbagai situs di internet. Melalui kajian studi literatur ini diharapkan diperoleh informasi yang mendukung penyelesaian rumusan masalah dan dapat merealisasikan luaran penelitian sesuai dengan tujuan penelitian.

4.2 Perancangan

4.2.1. Skematik Alat

Pada penelitian ini, box acrylic dibangun dengan acrylic berwarna hitam pekat dengan ketebalan 3mm. berdasarkan rancangan yang dipaparkan pada bab 3, secara keseluruhan sistem ini telah selesai dibangun dengan hasil sebagai berikut:



Gambar 4.1 Box Acrylic

Setelah diinstallan LCD dan perangkat raspberry pi yang terpasang pada sisi samping box, diperoleh hasil rancangan sebagai berikut



Gambar 4.2 LCD dan Raspberry pi

Rancangan dudukan bawang dicetak menggunakan jasa percetakan 3d print yang tersedia secara online dan diperoleh hasil sebagai berikut:



Gambar 4.3 Percetakan 3d

4.2.2. Implementasi dasar kamera

Pada penelitian ini dilakukan pengujian pembacaan kamera yang terhubung dengan sebuah webcam dan library python opencv. Melalui program pembacaan kamera sebagai berikut

```

lcd_test.py x Aplikasi.py x baca_kamera.py x
3 import cv2 # library opencv
4
5 import I2C_LCD_driver
6 import time
7 import RPi.GPIO as GPIO
8
9
10 camera = cv2.VideoCapture(0) # membuat objek kamera, indeks 0 artinya kamera pertama
11
12 while True:
13     (grabbed, frame) = camera.read() # ambil gambar dari kamera, grabbed = status 0/1, hasil gambar dimasukan ke frame
14     frame = imutils.resize(frame, width=320) # rubah ukuran frame dengan lebar 320 pixel
15     gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
16
17
18     thresh = cv2.threshold(frame, 90, 255, cv2.THRESH_BINARY)[1]
19
20     cv2.imshow("Gambar asli", frame) # tampilkan gambar dengan windows bernama gambar asli
21     cv2.imshow("abu-abu", gray)
22     cv2.imshow("threshold", thresh)
23     key = cv2.waitKey(1) & 0xFF #waitkey artinya tunggu ada input dari keyboard
24
25     # if the 'q' key is pressed, break from the lop
26     if key == ord("q"):
27         break

```

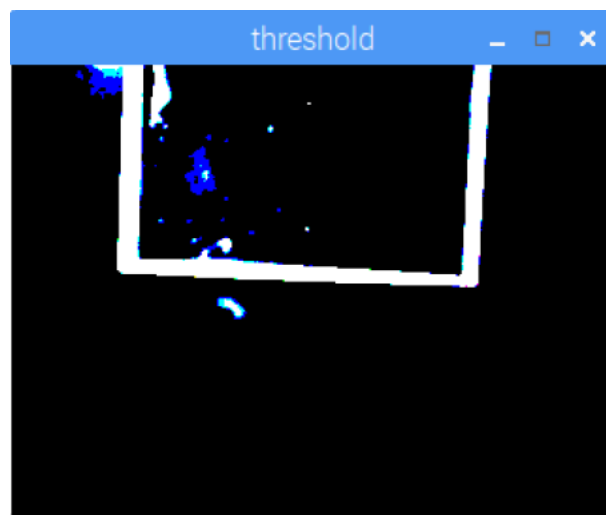
Gambar 4.4 Program Pembacaan Kamera

Diperoleh hasil pembacaan sebagai berikut:



Gambar 4.5 Hasil Pembacaan

Pada pembacaan ini dilakukan thresholding untuk menampilkan warna dasar RGB dari kamera dan dihasilkan gambar sebagai berikut



Gambar 4.6 Pembacaan thresholding

4.3 Implementasi

Pada Implementasi ini ,diuji hasil tampilan display untuk menampilkan Bawang Sakit atau Bawang Sehat.

4.3.1.Pengujian SNR

SNR atau signal to noise ratio mengukur kualitas pembacaan kamera dari webcam yang digunakan yaitu Logitech C720. Pengujian ini menggunakan program sebagai berikut

```
def signaltonoise(a, axis=0, ddof=0):
    a = np.asarray(a)
    m = a.mean(axis)
    sd = a.std(axis=axis, ddof=ddof)
    return np.where(sd == 0, 0, m/sd)
```

Hasil dari pengujian ini terdapat pada gambar 4.7:

```
[ 15.00994781  30.69599376  17.0091347 ]
[ 14.96507394  31.32148372  16.59491794 ]
[ 14.30314555  28.519241    16.02421995 ]
[ 14.58788455  30.38234585  18.4520103 ]
[ 16.03251234  30.90631391  19.40400126 ]
[ 15.89201697  31.71127678  21.31611725 ]
[ 15.67571635  29.13499299  20.88402653 ]
[ 14.98605553  29.34460323  19.99054801 ]
[ 16.37790455  30.67859955  19.24262366 ]
```

Gambar 4.7 Pengujian SNR

Berdasarkan tabel 4.1 pada pengujian ini diperoleh data sebagai berikut:

Tabel 4.1 Tabel Pengujian

B	G	R
15	30.69	17
14.96	31.32	16.59
14.3	28.51	16.02
14.58	30.38	18.45
16.03	30.9	19.4
15.89	31.7	21.31
15.67	29.13	20.88
14.9	29.34	19.99
16.37	30.67	19.24

Berdasarkan data diatas diperoleh angka statistic. Berikut angka statistic pada tabel 3.2 dibawah ini:

Tabel 4.2 Tabel Angka Statistic

statistic	B	G	R
max	16.37	31.7	21.31
min	14.3	28.51	16.02
average	15.3	30.29333	18.76444

Dengan data pada tabel diatas diperoleh nilai rata-rata terendah ada pada spektrum warna biru dengan nilai SNR 15.3dB, dan tertinggi ada pada spektrum warna hijau dengan nilai 30.29dB. Adapun nilai terendah terjadi pada warna biru yaitu 14.3dB dan nilai tertinggi pada warna hijau yaitu 31.7dB. berdasarkan nilai SNR diatas menunjukkan nilai SNR diatas 0dB yang menunjukkan bahwa kemampuan kamera Logitech C270 ini menghasilkan sinyal yang baik dibandingkan noise yang diperoleh.

Pengujian tombol dan pengambilan gambar

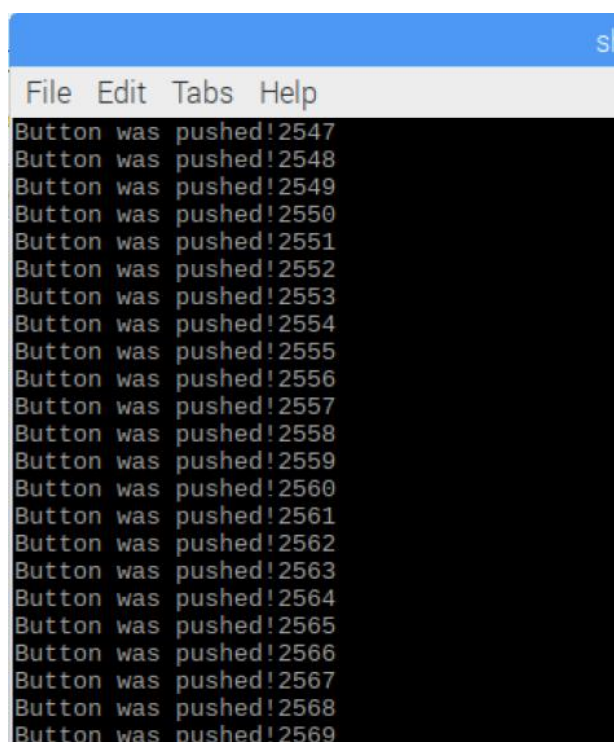
Pada pengujian ini dilakukan pembacaan tombol dimana tombol ini dihubungkan pada pin GPIO 16 dan pada bagian lain dihubungkan ke pin Ground. Pada penelitian ini digunakan konsep Pull UP dimana ketika tombol dalam keadaan tidak ditekan maka nilai pembacaan akan bernilai 1 sedangkan ketika tombol ditekan pembacaan akan bernilai 0.

Program pembacaan tombol ini diimplementasikan dengan program pada gambar 4.8 berikut:

```
1 import RPi.GPIO as GPIO # Import Raspberry Pi GPIO library
2 GPIO.setwarnings(False) # Ignore warning for now
3 GPIO.setmode(GPIO.BCM) # Use physical pin numbering
4
5
6 GPIO.setup(16, GPIO.IN, pull_up_down=GPIO.PUD_UP)
7 cnt = 0
8 while True: # Run forever
9     if GPIO.input(16) == GPIO.LOW:
10        print("Button was pushed!" + str(cnt))
11        cnt = cnt + 1
12
```

Gambar 4.8 Program Pembacaan Tombol

Pada pengujian ini, dilakukan pengujian menekan tombol selama 2 detik dan diperoleh hasil pada gambar 4.9 sebagai berikut:



```
sh
File Edit Tabs Help
Button was pushed!2547
Button was pushed!2548
Button was pushed!2549
Button was pushed!2550
Button was pushed!2551
Button was pushed!2552
Button was pushed!2553
Button was pushed!2554
Button was pushed!2555
Button was pushed!2556
Button was pushed!2557
Button was pushed!2558
Button was pushed!2559
Button was pushed!2560
Button was pushed!2561
Button was pushed!2562
Button was pushed!2563
Button was pushed!2564
Button was pushed!2565
Button was pushed!2566
Button was pushed!2567
Button was pushed!2568
Button was pushed!2569
```

Gambar 4.9 Hasil Pengujian

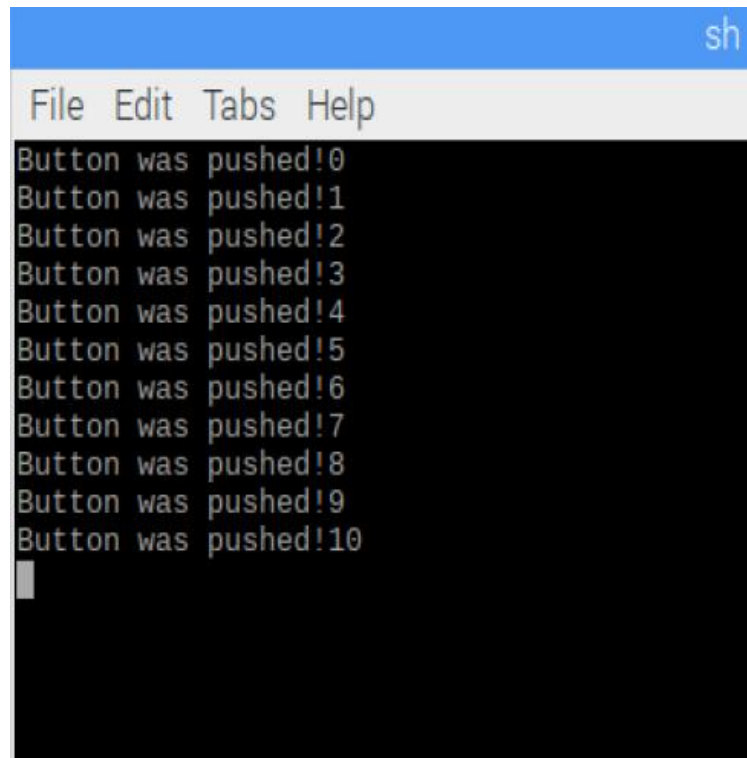
Berdasarkan nilai angka yang ditampilkan pada terminal, dalam waktu 2 detik pembacaan ini menampilkan bahwa tombol ditekan hingga 2569 kali. Nilai tersebut terbaca ketika GPIO 16 bernilai 0, oleh karena itu sistem pembacaan

tombol dimodifikasi dengan menggunakan sistem status yang akan membaca tombol Kembali setelah GPIO bernilai 1. Untuk menghindari terlalu cepat pembacaan tombol dan noise yang dihasilkan, diimplementasikan time sleep sehingga program pembacaan tombol diimplementasikan pada gambar 4.10 sebagai berikut:

```
1 import RPi.GPIO as GPIO # Import Raspberry Pi GPIO library
2 GPIO.setwarnings(False) # Ignore warning for now
3 GPIO.setmode(GPIO.BCM) # Use physical pin numbering
4 import time
5
6 GPIO.setup(16, GPIO.IN, pull_up_down=GPIO.PUD_UP)
7 cnt = 0
8 status = 0
9 while True: # Run forever
10     if GPIO.input(16) == GPIO.LOW and status==0:
11         print("Button was pushed!" + str(cnt))
12         cnt = cnt + 1
13         status = 1
14         time.sleep(0.1)
15     elif GPIO.input(16) == GPIO.HIGH and status==1:
16         status = 0
17         time.sleep(0.1)
18
19
```

Gambar 4.10 Pembacaan Tombol

Hasil dari implementasi ini diperoleh nilai pembacaan tombol yang stabil dengan hasil pada gambar 4.11 sebagai berikut:



```

sh
File Edit Tabs Help
Button was pushed!0
Button was pushed!1
Button was pushed!2
Button was pushed!3
Button was pushed!4
Button was pushed!5
Button was pushed!6
Button was pushed!7
Button was pushed!8
Button was pushed!9
Button was pushed!10

```

Gambar 4.12 Hasil Pembacaan

Implementasi pengambilan gambar

Dengan penggunaan tombol ini, diimplementasikan pengambilan gambar ketika tombol ditekan. Untuk melakukan aplikasi ini digabungkan program pembacaan tombol dan library opencv yang dapat membaca kamera.

Program untuk menyimpan gambar dari bacaan opencv menggunakan syntax

Syntax: `cv2.imwrite(filename, image)`

Parameters:

filename: A string representing the file name. The filename must include image format like **.jpg**, **.png**, etc.

image: It is the image that is to be saved.

Return Value: It returns true if image is saved successfully.

Fungsi pengambilan gambar diimplementasikan dengan program pada gambar 4.13 sebagai berikut:

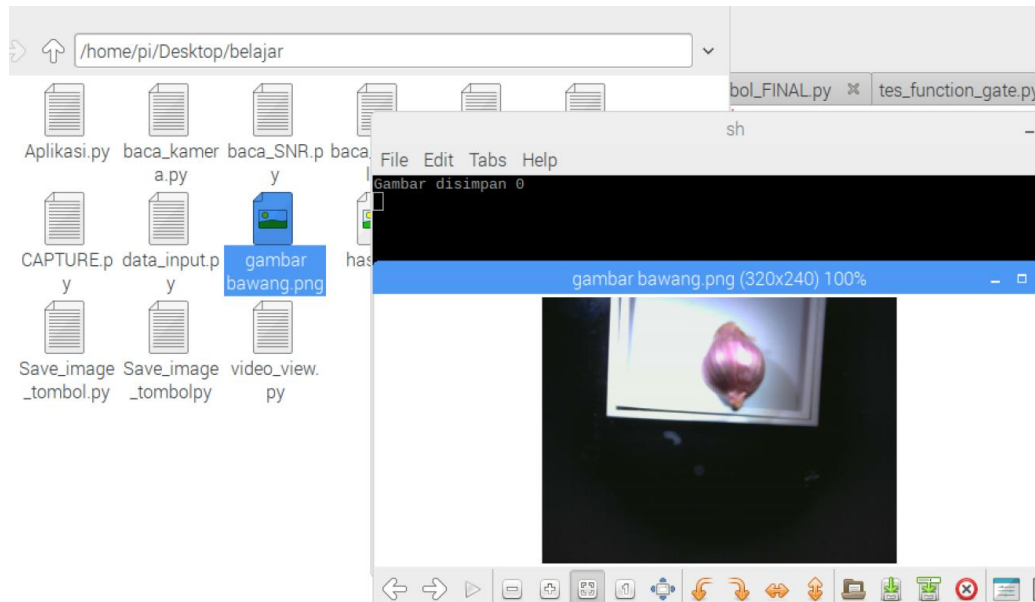
```

20 while True:
21     (grabbed, frame) = camera.read() # ambil gambar dari kamera, grabbed = status 0/1, hasil gambar
22     frame = imutils.resize(frame, width=320) # rubah ukuran frame dengan lebar 320 pixel
23     gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
24
25
26     thresh = cv2.threshold(frame, 90, 255, cv2.THRESH_BINARY)[1]
27
28     cv2.imshow("Gambar asli", frame) # tampilkan gambar dengan windows bernama gambar asli
29     cv2.imshow("abu-abu", gray)
30     cv2.imshow("threshold", thresh)
31     key = cv2.waitKey(1) & 0xFF #waitkey artinya tunggu ada input dari keyboard
32
33     if GPIO.input(16) == GPIO.LOW and status==0:
34         print("Gambar disimpan " + str(cnt))
35         cnt = cnt + 1
36         status = 1
37         cv2.imwrite("gambar bawang.png", frame)
38         time.sleep(0.1)
39     elif GPIO.input(16) == GPIO.HIGH and status==1:
40         status = 0
41         time.sleep(0.1)
42
43     # if the `q` key is pressed, break from the loop
44     if key == ord("q"):

```

Gambar 4.13 Pengambilan Gambar

Dengan implementasi ini, diperoleh hasil pada gambar 4.14 sebagai berikut:



Gambar 4.14 Pengambilan Gambar

Secara keseluruhan pengujian ini dilakukan sebanyak 10 kali dan mampu menyimpan gambar dengan baik 100%. Berikut data pengujian pada tabel 3.3 dibawah ini:

Tabel 4.3 Tabel Pengambilan Gambar

Pengambilan gambar	status
1	berhasil
2	berhasil
3	berhasil
4	berhasil
5	berhasil
6	berhasil
7	berhasil
8	berhasil
9	berhasil
10	berhasil

4.3.2 Pengujian Display

Pada implementasi ini, diuji hasil tampilan display sesuai dengan flowchart. Dalam rancangan penelitian ini, dibuat beberapa menu yaitu:

1. Menu awal: menampilkan “Deteksi Fusarium”

Fungsi menu ini untuk menampilkan kondisi bahwa sistem sudah mulai berjalan namun dalam proses mengaktifkan kamera pada gambar 4.15 berikut ini:



Gambar 4.15 Deteksi Fusarium

2. Menu ready: menampilkan “Silahkan tekan tombol”

Menu ini disediakan untuk memberikan pesan kepada pengguna agar dapat memulai proses deteksi. Pada saat menu ini ditampilkan, kamera sudah siap membaca seperti pada gambar 4.16 dibawah ini:



Gambar 4.16 Menu Tombol

3. Menu deteksi daun: menampilkan “Masukan daun” dan “tekan tombol”

Pada menu ini, kamera siap membaca daun dengan berbagai fungsi pembacaan khusus mendeteksi layu pada daun. Setelah tombol ditekan, sistem akan menyimpan gambar dan melanjutkan ke menu selanjutnya. Seperti pada gambar 4.17 berikut ini:



Gambar 4.17 Menu Deteksi Daun

4. Menu deteksi bawang: menampilkan “Masukan bawang” dan “tekan tombol”

Pada menu ini, sistem siap membaca gambar bawang atau umbi, setelah tombol ditekan sistem akan melakukan pembacaan gambar keseluruhan baik

umbu dan daun untuk menentukan apakah bawang mengalami layu fusarium atau tidak pada gambar 4.18 berikut ini:



Gambar 4.18 Menu Deteksi Bawang

5. Menu hasil : menampilkan “Bawang : sakit” atau “Bawang :sehat” dan “tekan tombol”

Setelah terdeteksi, hasil akan ditampilkan pada LCD dan user dipersilahkan untuk menekan tombol Kembali agar dapat mengulangi proses deteksi pada bawang dan daun lainnya seperti pada gambar 4.19 dibawah ini:





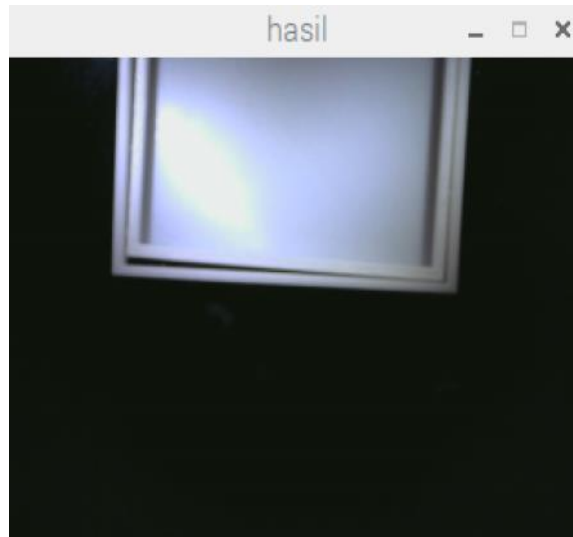
Gambar 4.19 Menu Hasil Bawang Sakit atau Sehat

Secara keseluruhan, fungsi-fungsi display dapat berjalan dengan baik dan setiap tampilan dijalankan dengan program yang dipicu oleh tombol yang ditekan.

4.4 Pengujian dan Analisa

pada sistem ini, daun layu dideteksi dengan mengukur ada atau tidaknya porsi warna hijau pada daun. Untuk mengetahui nilai warna hijau pada gambar yang ditangkap oleh kamera, digunakan masking dengan opencv yaitu opencv hanya menampilkan warna yang berada pada rentang warna tertentu.

Dalam kondisi kosong, atau tidak ada objek berwarna hijau, kamera akan menangkap gambar sebagai berikut pada gambar 4.20:



Gambar 4.20 Pengujian Akurasi Daun Layu

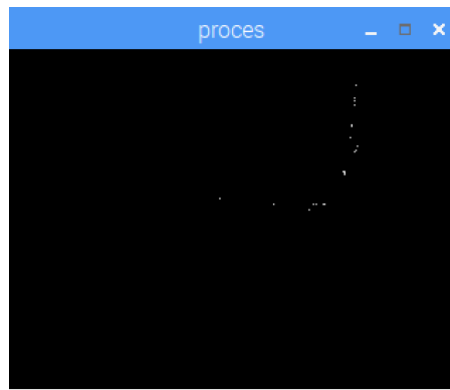
Pada sistem ini untuk melakukan masking warna hijau, dipilih rentang yang bernilai sebagai berikut:

`low_green=[0, 1, 0]`

`high_green=[0, 255, 0]`

dimana rentang terendah berdasarkan konfigurasi warna open cv yaitu Blue, Green dan Red atau disingkat BGR, warna biru atau Blue dan warna merah atau Red diberi rentang terendah bernilai 0 dan rentang tertinggi bernilai 0 juga. Dengan pengaturan rentang blue dan red tersebut, maka warna tidak selain hijau tidak akan ditampilkan, sedangkan untuk warna hijau diberi rentang 1 – 255.

Pada kondisi kosong, hasil masking menampilkan gambar pada gambar 4.21 berikut:



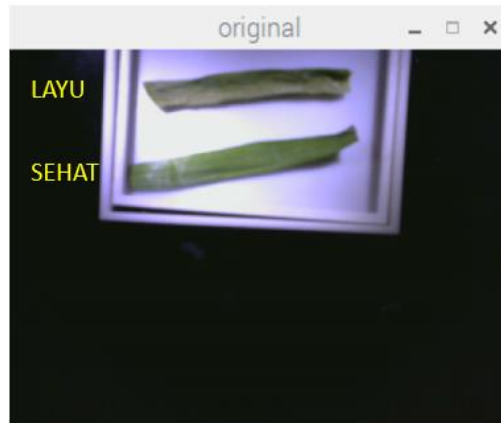
Gambar 4.21 Hasil Masking

Pada kondisi ada objek berwarna hijau, dalam penelitian ini adalah daun bawang, maka dapat dideteksi daun bawang baik yang sehat ataupun yang sakit. Berikut kondisi daun bawang pada gambar 4.22 dibawah ini:



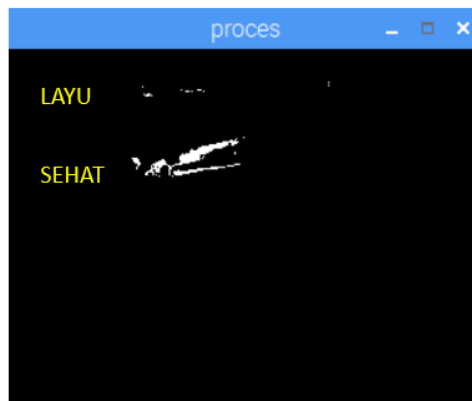
Gambar 4.22 Kondisi Daun Bawang

Dibawah ini adalah gambar ketika kedua jenis daun dimasukan kedalam sistem/ Berikut gambarannya pada gambar 4.23 dibawah ini:



Gambar 4.23 Layu Daun

Setelah mengalami masking warna hijau sistem ini akan mengkonversi warna hijau pada gambar menjadi warna putih dan sistem akan menampilkan gambar sebagai berikut pada gambar 4.24 dibawah ini:



Gambar 4.24 Implementasi Masking

Dapat terlihat bahwa daun yang layu memiliki area warna hijau yang tidak luas, sedangkan daun yang sehat menampilkan area warna hijau yang luas. Dengan mengimplementasikan masking warna ini, dapat dideteksi apakah sebuah daun memiliki porsi warna hijau yang banyak atau tidak dengan cara mendeteksi area kontur. Kontur adalah fitur yang terdapat pada opencv untuk mendeteksi area warna putih pada sebuah gambar hitam putih dan dapat mengukur luas area nya.

Fungsi kontur dipaparkan dengan program pada gambar 4.25 berikut:

```

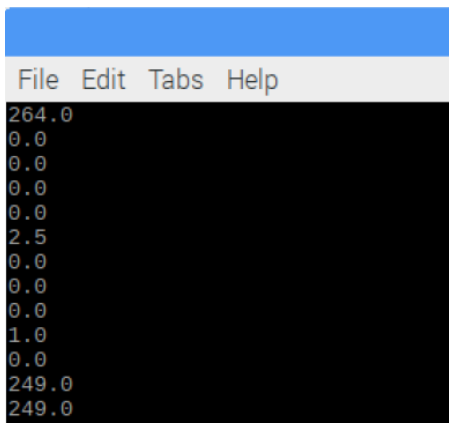
cnts, _ = cv2.findContours(Th_output.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

# loop over the contours
for c in cnts:
    print cv2.contourArea(c)
    if cv2.contourArea(c) < 60:
        continue

```

Gambar 4.25 Fungsi Kontur

Dengan fungsi kontur, diperoleh luas area yang besar berada pada kisaran di atas 150 pixel². Berikut hasil data pada gambar 4.26 dibawah ini:



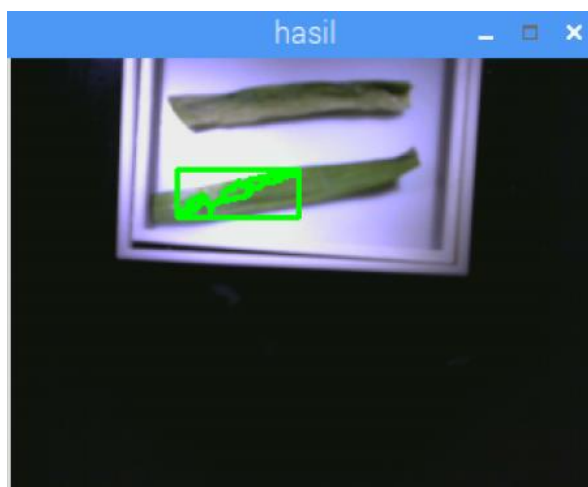
```

File Edit Tabs Help
264.0
0.0
0.0
0.0
0.0
2.5
0.0
0.0
0.0
1.0
0.0
249.0
249.0

```

Gambar 4.26 Hasil Kontur

Hasil akhir dari proses deteksi kontur ini dapat menentukan area warna hijau yang hanya muncul pada daun yang sehat, dimana pada fungsi kontur juga area hijau yang dominan akan dapat dideteksi lokasinya. Berikut hasil akhir deteksi kontur pada gambar 4.27 dibawah ini:



Gambar 4.27 Hasil Akhir Deteksi Kontur

Percobaan pengujian akurasi dilakukan dengan mengambil 10 kali pendeteksian kontur pada daun sehat dan daun layu. Status berhasil pada daun sehat menyatakan adanya kontur area warna hijau dengan luas diatas 150 pixel pada daun, sedangkan status daun layu dinyatakan berhasil apabila tidak terdeteksi kontur area berwarna hijau diatas 150 pixel. Berikut hasil data penelitian pada tabel 3.4 dibawah ini:

Tabel 4.4 Tabel Status Daun

daun sehat	status	daun sakit	status
1	berhasil	1	berhasil
2	berhasil	2	gagal
3	berhasil	3	berhasil
4	berhasil	4	berhasil
5	gagal	5	gagal
6	berhasil	6	berhasil
7	berhasil	7	berhasil
8	berhasil	8	gagal
9	gagal	9	berhasil
10	berhasil	10	berhasil

Dari hasil 10 kali percobaan diperoleh akurasi penentuan daun sehat sebesar 80% dan daun sakit sebesar 70%. Nilai daun sakit memiliki nilai akurasi yang lebih kecil diakibatkan karena daun yang berpenyakit memiliki berbagai macam variasi

tingkat kelayuan sehingga semakin layu daun akan semakin mudah terdeteksi karena tidak memiliki area warna hijau yang luas. Secara umum, rata-rata pendeteksian daun diperoleh akurasi senilai 75%.

4.4.1 Pengujian akurasi Bawang

pada sistem ini, bawang layu dideteksi dengan mengukur ada atau tidaknya porsi warna kuning pada bawang. Untuk mengetahui nilai warna kuning pada gambar yang ditangkap oleh kamera, digunakan masking dengan opencv dengan rentang

```
low_red=[0, 250, 1]
```

```
high_red=[0, 255, 255]
```

pada opencv ini, warna kuning adalah warna yang dihasilkan oleh gabungan warna hijau dan merah, sehingga rentang yang diatur pada penentuan layu fusarium bawang merah digunakan rentang bawah dan atas untuk warna biru bernilai 0, sedangkan warna hijau 250 hingga 255 dan warna merah 1 hingga 255. Warna hijau dipilih pada rentang yang sempit yaitu 250 hingga 255 agar hanya menampilkan hijau terang yang bila digabungkan dengan merah akan menampilkan warna kuning. Berikut tampilan gambar pada gambar 4.28 dibawah ini:



Gambar 4.28 Pengujian Akurasi Bawang

Dengan implementasi masking, layu fusarium pada bawang dikatakan apabila terdeteksi warna kuning, sedangkan bawang sehat tidak akan menampilkan warna

kuning. Hasil masking ini menampilkan gambar pada gambar 4.29 sebagai berikut:



Gambar 4.29 Hasil Masking

Dapat terlihat bahwa kontur area bawang yang layu menampilkan kontur yang lebih besar dibandingkan bawang yang sehat. Dengan penentuan luas area diatas 25 pixel adalah bawang layu dapat dideteksi box area bawang yang layu seperti gambar 4.30 berikut ini:



Gambar 4.30 Kontur Area Bawang

Percobaan pengujian akurasi dilakukan dengan mengambil 10 kali pendeteksian kontur pada bawang sehat dan bawang layu. Status berhasil pada bawang sehat menyatakan tidak terdeteksi kontur warna kuning dengan luas area diatas 25 pixel, sedangkan status bawang layu dinyatakan berhasil apabila terdeteksi kontur area berwarna kuning diatas 25 pixel. Berikut hasil data penelitian pada tabel 3.5 dibawah ini:

Tabel 4.5 Tabel Status Pada Bawang

bawang sehat	status	bawang sakit	status
1	berhasil	1	berhasil
2	berhasil	2	berhasil
3	berhasil	3	berhasil
4	berhasil	4	berhasil
5	berhasil	5	berhasil
6	berhasil	6	berhasil
7	gagal	7	berhasil
8	berhasil	8	gagal
9	berhasil	9	berhasil
10	gagal	10	gagal

Dari hasil 10 kali percobaan diperoleh akurasi penentuan bawang sehat sebesar 80% dan bawang sakit sebesar 80%. Nilai bawang sakit dan bawang sehat menunjukkan akurasi yang sama yaitu 80% dimana perbedaan kondisi layu cukup signifikan berbeda untuk bawang yang sehat.

Pengujian Keseluruhan Sistem

Sistem ini dirancang untuk menentukan kondisi bawang mengalami layu fusarium dengan mendeteksi apakah daun layu dan bawang layu. berdasarkan hasil pengujian diperoleh hasil pendeteksian sebagai berikut:

1. Mengalami Layu Fusarium



Gambar 4.31 Pengujian Keseluruhan Sistem

Pada kondisi layu fusarium seperti pada gambar 4.31 diatas. Daun tidak terdeteksi area warna hijau sedangkan bawang terdeteksi area warna kuning di beberapa titik. Dengan kondisi ini, dikatakan bahwa bawang mengalami layu fusarium.

Pada program diimplementasikan nilai Boolean untuk kondisi daun tidak terdeteksi warna hijau bernilai *true* dan bawang terdeteksi warna kuning bernilai *true*. Berikut data kondisi daun pada gambar 4.32 dibawah ini:

```
File Edit Tabs Help
daun sakit:True
bawang sakit:True
kondisi bawang: Sakit
```

Gambar 4.32 Kondisi Daun

2. Kondisi bawang sehat

Pada kondisi sehat, daun terdeteksi area warna hijau sedangkan bawang tidak terdeteksi area warna kuning dengan luas area diatas 25 pixel. Dengan kondisi ini, dikatakan bahwa bawang sehat seperti pada gambar 4.33 dibawah ini:



Gambar 4.33 Kondisi Bawang Sehat

Pada program diimplementasikan nilai Boolean untuk kondisi daun terdeteksi warna hijau bernilai *false* dan bawang tidak terdeteksi warna kuning bernilai *false* seperti gambar 4.34 dibawah ini:

```

File Edit Tabs Help
daun sakit:False
bawang sakit:False
kondisi bawang: Sehat

```

Gambar 4.34 Implementasi Program

Dilakukan percobaan sebanyak 20 kali untuk mengetahui kinerja sistem dan diperoleh hasil. Berikut hasil data pada tabel 3.6 dibawah ini:

Tabel 4.6 Tabel Kinerja Sistem

Layu Fusarium	status	Tanaman Sehat	status
1	berhasil	1	berhasil
2	berhasil	2	berhasil
3	berhasil	3	berhasil
4	berhasil	4	berhasil
5	gagal	5	berhasil
6	berhasil	6	gagal
7	berhasil	7	berhasil

8	berhasil	8	berhasil
9	gagal	9	berhasil
10	berhasil	10	berhasil
11	berhasil	11	gagal
12	gagal	12	gagal
13	berhasil	13	berhasil
14	gagal	14	berhasil
15	gagal	15	berhasil
16	berhasil	16	gagal
17	berhasil	17	berhasil
18	berhasil	18	gagal
19	berhasil	19	berhasil
20	berhasil	20	gagal

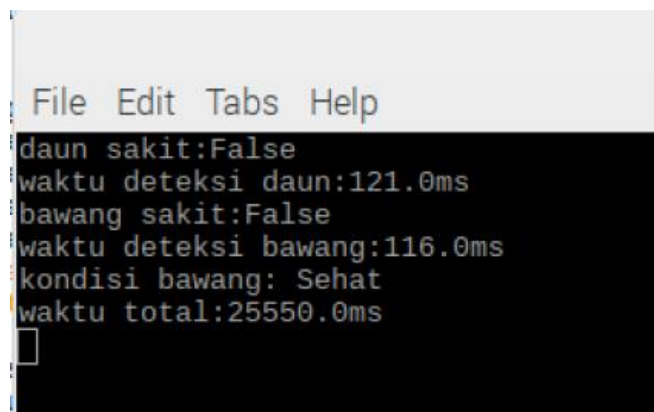
Dari hasil percobaan ini, diketahui proses deteksi bawang merah yang sehat sebesar 75% sedangkan deteksi bawang yang mengalami layu fusarium 70%. Hasil pendeteksian bawang yang mengalami layu fusarium lebih rendah dikarenakan sistem yang diimplementasikan menggunakan logika Boolean “AND” dimana syarat diketahui bawang mengalami fusarium adalah kondisi daun terdeteksi layu dan bawang terdeteksi layu. Adapun kondisi yang menurunkan pendeteksian layu fusarium dikarenakan objek warna hijau masih terdeteksi pada daun yang mengalami layu. secara umum rata-rata akurasi pendeteksian layu fusarium senilai 72,5%.

4.4.2 Pengujian waktu deteksi

Untuk menguji kinerja secara waktu, diimplementasikan fungsi millisecond dengan library time pada opencv yang dibuat dengan fungsi sebagai berikut:

Pengujian ini mengukur waktu pendeteksian daun, waktu pendeteksian bawang dan total waktu yang dibutuhkan dimulai tombol ditekan hingga

menampilkan hasil. Pada terminal ditampilkan hasil pembacaan waktu pada gambar 4.35 sebagai berikut:



```
File Edit Tabs Help
daun sakit:False
waktu deteksi daun:121.0ms
bawang sakit:False
waktu deteksi bawang:116.0ms
kondisi bawang: Sehat
waktu total:25550.0ms
```

Gambar 4.35 Pengujian Waktu Deteksi

Pada pengujian ini, diketahui waktu deteksi daun 121 milisecond dan waktu deteksi bawang 116 milisecond. Adapun waktu keseluruhan yang dibutuhkan dimulai dari menekan tombol, memasukan sampel daun, mengeluarkan sampel daun, memasukan sampel bawang dan mengetahui hasil akhir dibutuhkan waktu sebesar 25550 milsecond, atau secara total 25 detik. Berikut hasil data pada tabel 3.7 dibawah ini:

Tabel 4.7 Tabel Deteksi

Deteksi Daun (ms)	Deteksi bawang (ms)	waktu keseluruhan (ms)
121	117	28161
118	135	36746
123	119	33902
114	120	29533
134	113	29290
121	113	41413
114	116	35985
112	135	31193
118	129	30331
114	130	26914

Dari 10 kali percobaan, diketahui rata-rata proses pendeteksian daun sebesar 118.9 milisecond, pendeteksian bawang 122.7 milisecond dan waktu percobaan rata-rata 32 detik.

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Telah dirancang dan diimplementasikan sistem deteksi penyakit layu fusarium pada bawang merah dengan pengolahan citra. Sistem ini dirancang dengan library opencv sebagai library pengolahan citra berbasis bahasa pemrograman python dan ditanamkan pada mikrokomputer raspberry pi dengan hasil:

- Pendeteksian layu fusarium pada daun dilakukan dengan mendeteksi kontur area warna hijau pada daun dimana luas area warna hijau terdeteksi dibawah 150 pixel².
- Pendeteksian layu fusarium pada bawang dilakukan dengan mendeteksi kontur area warna kuning dimana luas area warna kuning terdeteksi diatas 25 pixel² .
- Akurasi deteksi layu fusarium pada daun diperoleh senilai 75%, akurasi deteksi layu fusarium pada bawang diperoleh senilai 80% dan deteksi keseluruhan senilai 72.5%.
- Waktu rata-rata proses pendeteksian daun sebesar 118.9 milisecond, pendeteksian bawang 122.7 milisecond dan waktu percobaan rata-rata 32 detik

5.2 Saran

Adapun saran yang diusulkan penulis untuk penelitian kedepannya antara lain:

- Mengimplementasikan algoritma cerdas dalam pendeteksian layu fusarium berbasis artificial intelligent
- Meningkatkan spesifikasi perangkat keras untuk dapat diotomasi di skala pertanian besar
- Mengintegrasikan sistem menjadi sebuah perangkat internet of things.

DAFTAR PUSTAKA

Abdul Kadir, Adhi Susanto (2013), “Teori dan Aplikasi Pengolahan Citra”, 50-60.

Azzamy.2017.pengendalian penyakit layu F. oxysporum pada tanaman bawang merah. di akses pada 7 april 2017.

Fadhilah, S., Wiyono dan Surahman. 2014. Pengembangan Teknik Deteksi Fusarium Patogen Pada Benih Umbi Bawang Merah (*Allium ascalonicum*) di Laboratorium. *J. Horti*. 24(2): 171-178, 2014.

R.C. Gonzales and R.E. Wood (2002), “Digital Image Processing, Second Edition”, Prentice Hall, 117-121..

Sigit riyanto, step by step pengolahan citra digital. surabaya, 2005.

LAMPIRAN

1. Program Python (Aplikasi Deteksi Bawang Layu Fusarium)

```
import imutils # image utility ->resize
import time # waktu untuk sleep
import cv2 # library opencv
import numpy as np
import I2C_LCD_driver
import time
import RPi.GPIO as GPIO

GPIO.setwarnings(False) # Ignore warning for now
GPIO.setmode(GPIO.BCM) # Use physical pin numbering

mylcd = I2C_LCD_driver.lcd() #inisiasi objek mylcd

mylcd.lcd_display_string("Deteksi", 1, 0) #tampilan pada bagian
pertama
mylcd.lcd_display_string("Fusarium", 2, 0) #tampilan pada bagian
kedua

camera = cv2.VideoCapture(0) # membuat objek kamera, indeks 0
artinya kamera pertama
GPIO.setup(16, GPIO.IN, pull_up_down=GPIO.PUD_UP)
cnt = 0
status = 0
menu = 0
ready = 0
bawang_sakit = False
daun_sakit = False

th = 200
th_daun = 130
th_bawang = 80
def millis_time():
    return round(time.time() * 1000)

def deteksi_fusarium(img,th,min_area,menu):
    detect=False
    low_red=[0, 250, 1]
    high_red=[0, 255, 255]

    low_green=[0, 1, 0]
    high_green=[0, 255, 0]

    if(menu==1):
        lower = np.array(low_green, dtype = "uint8")
        upper = np.array(high_green, dtype = "uint8")
    elif(menu==2):
        lower = np.array(low_red, dtype = "uint8")
        upper = np.array(high_red, dtype = "uint8")
    thresh = cv2.threshold(img,th,255,cv2.THRESH_BINARY)[1]
```

```

    thresh = cv2.erode(thresh, None, iterations=5)
    mask = cv2.inRange(thresh, lower, upper)
    output = cv2.bitwise_and(thresh, thresh, mask = mask)
    gray_output = cv2.cvtColor(output, cv2.COLOR_BGR2GRAY)
    Th_output =
cv2.threshold(gray_output,55,255,cv2.THRESH_BINARY) [1]
    cnts, _ = cv2.findContours(Th_output.copy(),
cv2.RETR_EXTERNAL,
    cv2.CHAIN_APPROX_SIMPLE)

    # loop over the contours
    for c in cnts:
        #print cv2.contourArea(c)
        if cv2.contourArea(c) < min_area:
            continue
        detect = True
        #print cv2.contourArea(c)
        (x, y, w, h) = cv2.boundingRect(c)
        cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
        cv2.drawContours(img, [c], -1, (0, 255, 0), 2)

    if(menu==1):
        detect = not detect
        return img,detect
    timestart = 0
    time_daun = 0
    time_bawang = 0
    timeend = 0
    while True:
        (grabbed, frame) = camera.read() # ambil gambar dari
kamera, grabbed = status 0/1, hasil gambar dimasukan ke frame
        frame = imutils.resize(frame, width=320) # rubah ukuran
frame dengan lebar 320 pixel
        thresh = cv2.threshold(frame, th, 255, cv2.THRESH_BINARY) [1]

        cv2.imshow("Gambar asli", frame) # tampilkan gambar dengan
windows bernama gambar asli
        cv2.imshow("threshold",thresh)
        key = cv2.waitKey(1) & 0xFF #waitkey artinya tunggu ada input
dari keyboard

    if(menu==0 and ready==0):
        mylcd.lcd_clear()
        mylcd.lcd_display_string("Silahkan", 1, 0) #tampilan pada
bagian pertama
        mylcd.lcd_display_string("Tekan Tombol", 2, 0) #tampilan
pada bagian kedua
        ready=1
        timestart = millis_time()

    if GPIO.input(16) == GPIO.LOW and status==0:
        status = 1
        if(menu==0):
            mylcd.lcd_clear()
            mylcd.lcd_display_string("1.masukan daun", 1, 0)
            mylcd.lcd_display_string("2.TEKAN TOMBOL", 2, 0)

```

```

        th = th_daun
        menu = menu + 1
    elif(menu==1):
        time_daun = millis_time()
        cv2.imwrite("gambar daun.png", frame)
        cv2.imwrite("gambar daun th.png", thresh)
        daun, daun_sakit = deteksi_fusarium(frame, th, 60, 1)
        cv2.imwrite(str(daun_sakit) + "deteksi daun.png", daun)
        print("daun sakit:" + str(daun_sakit))
        print("waktu deteksi daun:" + str(millis_time() -
time_daun) + "ms")
        mylcd.lcd_clear()
        mylcd.lcd_display_string("1.masukan bawang", 1, 0)
        mylcd.lcd_display_string("2.TEKAN TOMBOL", 2, 0)
        th = th_bawang
        menu = menu + 1

    elif(menu==2):
        time_bawang = millis_time()
        cv2.imwrite("gambar bawang.png", frame)
        cv2.imwrite("gambar bawang th.png", thresh)
        bawang, bawang_sakit = deteksi_fusarium(frame, th, 20, 2)
        cv2.imwrite(str(bawang_sakit) + "deteksi
bawang.png", bawang)
        print("bawang sakit:" + str(bawang_sakit))
        print("waktu deteksi bawang:" + str(millis_time() -
time_bawang) + "ms")
        fusarium = "Sehat"
        if(bawang_sakit and daun_sakit):
            fusarium = "Sakit"
        print("kondisi bawang: " + fusarium)
        mylcd.lcd_clear()
        mylcd.lcd_display_string("Bawang:" + fusarium, 1, 0)
        mylcd.lcd_display_string("tekan tombol", 2, 0)
        print("waktu total:" + str(millis_time() - timestart) +
"ms")
        menu = menu + 1
    elif(menu==3):
        menu = menu + 1
    elif(menu==4):
        menu=0
        ready = 0

    time.sleep(0.1)
    elif GPIO.input(16) == GPIO.HIGH and status==1:
        status = 0
        time.sleep(0.1)

    # if the `q` key is pressed, break from the loop
    if key == ord("q"):
        break
    time.sleep(0.01)

# cleanup the camera and close any open windows
camera.release()
cv2.destroyAllWindows()

```

2. Dokumentasi Pembuatan Alat



```
sh
CAPTURE.py - /home...
Add New Device
sh
sh
File Edit Tabs Help
CAPT
INAL_PROG_REV2.py
deks 0 artinya kamera
Gambar asli
threshold
25 # cleanup the camera and close any open windows
26 camera.release()
27 cv2.destroyAllWindows()
28
```

