IMPLEMENTASI PENGOLAH CITRA PADA PROTOTIPE ALAT PENYORTIR LADA

PROYEK AKHIR

Laporan akhir ini dibuat dan diajukan untuk memenuhi salah satu syarat kelulusan Sarjana Terapan Politeknik Manufaktur Negeri Bangka Belitung

Disusun Oleh:

Nuzul Bragas Sabilillah NIM: 1051819

Rafda Ardianto NIM: 1052001

POLITEKNIK MANUFAKTUR NEGERI BANGKA BELITUNG

TAHUN 2024

LEMBAR PENGESAHAN

JUDUL PROYEK AKHIR

IMPLEMENTASI PENGOLAH CITRA PADA

PROTOTIPE ALAT PENYORTIR LADA

Oleh:

Nuzul Bragas Sabilillah /1051819

Rafda Ardianto/1052001

Laporsn akhir ini telah disetujui dan disahkan sebagai salah satu syarat kelulusan Program Sarjana Terapan Politeknik Manufaktur Negeri Bangka Belitung

Menyetujui,

Pembimbing 1

(Indra Dwisaputra, M.T.)

Pembimbing 2

(Ocsirendi, M.T.)

Penguji 1

(Aan Febriansyah, M.T.)

Penguji 2

(Nur Khasanah, M.Si.)

PERNYATAAN BUKAN PLAGIAT

Yang bertanda tangan di bawah ini:

Nama Mahasiswa 1	:	Nuzul Bragas Sabilillah	NIM: 1051917
Nama Mahasiswa 2	:	Rafda Ardianto	NIM: 1052005

Dengan Judul : Implementasi Pengolah Citra Pada Prototipe Alat Penyortir Lada

Menyatakan bahwa laporan akhir ini adalah hasil kerja kami sendiri dan bukan merupakan plagiat. Pernyataan ini kami buat dengan sebenarnya dan bila ternyata dikemudian hari ternyata melanggar penyataan ini, kami bersedia menerima sanksi yang berlaku.

Sungailiat, 3 Januari 2024

Nama Mahasiswa	Tanda Tangan
1. Nuzul Bragas Sabilillah	.....
2. Rafda Ardianto	.....

ABSTRAK

Penyortiran lada termasuk dalam aktifitas pertanian yang masih melibatkan manusia dalam melakukan prosesnya. Kadang kala kemanpuan manusia memiliki batas dalam performa kinerjanya seiring lamanya waktu kerja. Hal ini berpengaruh terhadap hasil pengidentifikasian lada tersebut. Lada dengan mutu kurang baik yang terlewat saat proses penyortiran lalu tergabung dengan lada yang lain, akan merugikan petani. Sebab itu mempengaruhi persentase kadar yang nantinya berpengaruh terhadap penentuan mutu. Penelitian ini dilakukan bertujuan untuk membuat mengimplementasikan pengolah citra pada pendeteksian lada dengan metode *YOLOv5*. Pengolah citra akan mendeteksi tiap butiran lada dan menentukan mutu tersebut dengan mengunakan parameter lada dengan warna yang kehitam-hitaman. Berdasarkan hasil percobaan diperoleh data *precision* sebesar 1, *recall* sebesar 0,97, *F1 score* sebesar 0,94 dan akurasi dari *confusion matrix* sebesar 63%. Pada penelitian kali ini performa pengolah citra yang dihasilkan sudah baik hanya saja akurasi yang dihasilkan harus dievaluasi lagi pada penelitian selanjutnya. Hal ini berpengaruh dari jumlah data gambar, kualitas data gambar, jumlah epoch saat training dan pemilihan bobot *YOLOv5* yang tepat.

Kata kunci: Lada, Penyortir Lada, Pengolah Citra, *YOLOv5*

ABSTRACT

The sorting of pepper is an agricultural activity in which human beings are still involved in the process. Sometimes human capabilities have a limit to their performance with the length of work time. This has an impact on the results of the identification of the pepper. It is detrimental to the farmer if poor quality peppers are missed during the sorting process and mixed with other peppers. Therefore, it has an impact on the percentage content, which has an impact on the determination of quality. The aim of this research was the implementation of image processing in pepper detection with the YOLOv5 method. The image processor will detect each grain of the pepper and will determine the quality based on the parameters of the pepper with a blackish colour. Based on the experimental results, the data obtained a precision of 1, a recall of 0.97, an F1 score of 0.94 and an accuracy of the confusion matrix of 63%. In this research, the performance of the resulting image processor is good. It's just that the resulting accuracy needs to be re-evaluated in further research. This depends on the amount of image data, the quality of the image data, the number of epochs used to train, and choosing the right YOLOv5 weights.

Keywords: Pepper, Pepper Sorter, Image Processing, YOLOv5

KATA PENGANTAR

Puji syukur kita panjatkan kehadirat Allah SWT atas karunianya, serta shalawat dan salam kepada Rasulullah Muhammad S.A.W, yang telah membawa umat manusia ke dunia yang damai, terang dan penuh ilmu pengetahuan. Sehingga penulis dapat menyelesaikan laporan tugas akhir ini yang berjudul ”Implementasi Pengolah Citra Pada Prototipe Alat Penyortir Lada”. Tujuan penulis membuat laporan tugas akhir ini adalah untuk memenuhi salah satu syarat kelulusan pendidikan Diploma IV di Politeknik Manufaktur Negeri Bangka Belitung. Dalam laporan tugas akhir, penulis membahas tentang hasil penelitian yang penulis laksanakan selama program proyek akhir berlangsung. Adanya implementasi pengolah citra terhadap alat peyortir lada diharapkan dapat mempermudah dan meringkaskan para petani dalam proses pengidentifikasi buah lada.

Pada kesempatan ini, ucapan terima kasih juga disampaikan kepada pihak yang telah banyak membantu serta ikut memberi motivasi, sumbang saran, kritik yang tentunya sangat diharapkan dalam penyelesaian tugas akhir ini. Berikut ini adalah pihak-pihak yang ikut membantu baik secara langsung maupun tidak langsung, diantaranya:

1. Bapak I Made Andik Setiawan, M.Eng, Ph.D selaku Direktur Politeknik Manufaktur Negeri Bangka Belitung.
2. Kepala Jurusan Teknik Elektro dan Informatika di Politeknik Manufaktur Negeri Bangka Belitung.
3. Kepala Program Studi D4 Teknik Elektronika di Politeknik Manufaktur Negeri Bangka Belitung.
4. Bapak Indra Dwisaputra, M.T. selaku dosen pembimbing 1 yang sudah memberikan kesempatan dan peluang untuk membimbing kami sehingga proyek akhir ini bisa terselesaikan.
5. Bapak Ocsirendi, M.T. selaku dosen pembimbing 2 yang sudah memberikan kesempatan dan peluang untuk membimbing kami sehingga proyek akhir ini bisa terselesaikan.

6. Seluruh Staff Komisi Proyek Akhir yang sudah membantu kegiatan Proyek Akhir.
7. Seluruh dosen dan pranata laboratorium pendidikan yang telah banyak membantu dalam penyelesaian proyek akhir ini.
8. Seluruh rekan-rekan mahasiswa kelas angkatan 25, 26, dan 27 Politeknik Manufaktur Negeri Bangka Belitung.
9. Seluruh rekan-rekan mahasiswa Politeknik Manufaktur Negeri Bangka Belitung.
10. Muhammad Irfan Afriyansyah mahasiswa Universitas Pamulang yang telah memberikan ilmunya sehingga penulis dapat menyelesaikan proyek akhir ini.
11. Semua pihak yang tidak dapat penulis sebutkan satu per satu atas segala bantuan unutk memudahkan penulis dalam menyelesaikan proyek akhir ini.

Dalam penyusunan Laporan ini tentunya masih banyak terdapat kekurangan, kesalahan dan kekhilafan karena keterbatasan kemampuan penulis, untuk itu sebelumnya penulis mohon maaf yang sebesar-besarnya. Penulis juga mengharapkan kritik dan saran dari semua pihak demi perbaikan yang bersifat membangun atas laporan ini. Akhir kata dengan segala kerendahan hati penulis mengucapkan rasa terima kasih dan semoga laporan ini dapat bermanfaat bagi penulis maupun kita bersama.

Sungailiat, 15 Januari 2024

Hormat kami,

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Lada yang bernama latin *piper nigrum* adalah hasil perkebunan dengan nilai ekonomi yang tinggi dalam komoditas lokal maupun ekspor. Di Indonesia sebagian wilayahnya menjadikan lada sebagai hasil perkebunan utama. Salah satunya Provinsi Kepulauan Bangka-Belitung. Provinsi ini memiliki predikat salah satu penghasil biji lada terbesar di Indonesia. Ini dibuktikan dengan salah satu kota di Kabupaten Bangka Barat yang memiliki hasil produksi lada yang terkenal hingga mancanegara, yaitu Kota Muntok. Dinamakan sesuai namanya, *Muntok White Pepper* menjadi primadona produk ekspor [1].

Dengan persaingan ekonomi yang terus meningkat seiring berjalannya kebutuhan konsumen. Maka persaingan kualitas produk pun juga ikut andil. Di Indonesia Badan Standarisasi Nasional (BSN) sebagai badan yang berwewenang metetapan standar suatu produk di Indonesia telah menerbitkan standarisasi kualitas mutu lada yang tertuang pada SNI 01-0004-2013 sebagai standarisasi mutu lada putih. Dari landasan inilah para petani dan pengepul lada dapat menentukan mutu lada yang akan dipasarkan. Dari proses ini masalah muncul, ini diakibatkan para petani dan pengepul masih menggunakan keahlian tenaga manusia untuk melakukan proses penyortiran tersebut. Hal ini sangat berpotensi mengalami kesalahan karena biji lada yang kecil dan kempuan manusia yang tidak bisa konsisten secara kontinyu. Dengan tergabungnya lada bermutu kurang baik masuk kedalam lada lain yang akan didistribusikan ini dapat merugikan petani. Sebab akan berpengaruh terhadap penetapan mutu lada dan mempengaruhi nilai harga tersebut [1].

Pada tahun 2023 penelitian dengan judul "Penentu Mutu Lada Berdasarkan Persentase Benda Asing Berbasis Pengolah Citra" oleh Muhammad Erfani dan Siti Barokah dilakukan. Penelitian ini membuat sebuah mesin yang memiliki

kemampuan mendeteksi lada, benda asing dan menghitung jumlah lada yang berada pada wadah secara *real-time* memanfaatkan teknologi pengolah citra menggunakan algoritma *YOLOv3*. Akan tetapi penelitian ini belum mampu mendeteksi lada putih dengan mengidentifikasi warna dari lada putih tersebut. Padahal hal ini sangat penting, karena dalam standar mutu lada putih warna lada yang memiliki warna yang kehitaman menjadi indikator penilaian mutu (SNI 01-0004-2013). Serta dalam penelitian ini masih mengunakan pedekatan dengan versi yang sudah lawas. Ini sejalan dengan perkembangan teknologi yang terus terbarukan [2].

Atas beberapa permasalahan tersebutlah penelitian yang diberi judul "Implementasi Pengolah Citra Pada Prototipe Alat Penyortir Lada" dilakukan. Dengan pengembangan kontruksi berupa pengayak sebagai alat penyortir lada dan memanfaatkan teknologi pengolah citra dengan pendekatan versi yang lebih baru. Pengembangan ini diharapkan membuat proses pendeteksi kualitas lada dengan indikator warna secara *real-time* menjadi lebih baik dan mampu menentukan kualitas lada sesuai dengan SNI 01-0004-2013 secara akurat dengan waktu yang lebih singkat.

1.2 Rumusan Masalah

Permasalahan ada pada pengepul di daerah Bangka ini yang masih melakukan penyortiran dan identifikasi mutu lada dengan cara manual sehingga didapatkan rumusan masalah sebagai berikut:

1. Bagaimana membuat sistem penyortir mutu lada putih berdasarkan bentuk dan memisahkan dengan benda asing?
2. Bagaimana membuat sistem pendeteksi mutu lada berlandaskan SNI 01-0004-2013 berbasis pengolahan citra?
3. Bagaimana membuat sistem dapat menampilkan mutu dan persentase hasil deteksi?

1.3 Tujuan Tugas Akhir

Adapun tujuan penelitian ini berdasarkan rumusan masalah dan Batasan masalah diatas sebagai berikut :

1. Membuat sistem yang mampu menyortir benda asing pada lada.
2. Memimplementasikan metode YOLOv5 pada pendeteksian lada.
3. Membuat sistem yang dapat menampilkan mutu dan persentase pendeteksian lada.

1.4 Batasan Masalah

Batasan masalah pada penelitian ini berdasarkan rumusan masalah diatas adalah sebagai berikut:

1. Sistem pengayak yang belum mampu menyaring dengan bobot lebih dari 1kg.
2. Alat ini masih diperuntukan untuk mendeteksi lada putih saja.
3. Pengolah citra dapat berkerja secara maksimal jika perangkat keras yang digunakan sesuai atau diatas spesifikasi minimum.

BAB II
LANDASAN TEORI

2.1 Penyortiran Lada

Penyortiran lada adalah proses terakhir dari serangkaian proses pengolahan lada mulai dari dipetik hingga dipasarkan. Pada proses penyortiran dalah proses yang krusial ini disebabkan proses ini masih menggunakan tenaga manusia atau dalam Bahasa lain kita sebut manual. Lada yang berada pada wadah pertama-tama akan diayak menggunakan nampan tradisional dan pengerjaan ini memiliki waktu yang fluktuatif. Setelah bagian selain lada terbuang selanjutnya penyortiran dilakukan menggunakan tangan satu persatu. Kadang kala dengan berjalannya waktu, tingkat akurasi dari penyortiran menurun akibat jumlah lada yang dipanen tidak sedikit, bahkan bisa sampai kurang lebih 20kg. Hal ini didasari oleh survey yang dilakukan kepada petani lada didaerah Kampung Kramat Pemali, Kabupaten Bangka. Proses penyortiran ini terkadang menghabiskan waktu yang tidak sebentar. Masalah itu pun muncul dengan tuntutan pasar yang tidak mengenal waktu. Ini merugikan petani karna mereka harus memotong waktu proses lain yang mengakibatkan kualitas mutu lada menurun [1].

Oleh sebab itu dibutuhkan sebuah alat pengayak otomatis untuk melakukan proses penyortiran lada yang dapat bekerja secara efektif untuk menggantikan peran manusia dalam melakukan penyortiran lada. Yang sekaligus bisa berdampak pada hasilnya menjadi lebih baik dengan jumlah bobot yang lebih banyak [3].

2.2 Standar Mutu Lada Indonesia

Di Indonesia penetapan standar atau mutu telah memiliki badan khusus yang menanganinya. Badan Standarisasi Nasional (BSN) adalah komite yang memiliki wewenang dalam menerbitkan standarisasi yang dignakan secara nasional atau disebut Standar Nasional Indonesiaa (SNI). Wewenang ini diberikan mengancu kepada PP Nomor 102 Tahaun 2000 Tentang Standarisasi Nasional.

Termasuk lada sebagai komponen perkebunan mmiliki standar mutu produk yakni SNI 01-0004-2013 & SNI 01-0004-2013 (Amd 1 2015) untuk lada putih. Atas dasar ini dapat diuraikan dalam tabel berikut bagaimana standarisasi diatur.

Tabel 2. 1 SNI 01-0004-2013 Mutu Lada Putih

No	Spesifikasi	Satuan	Persyaratan	
			Mutu I	Mutu II
1	Kerapatan	g/l	Min. 600	Min. 600
2	Kadar Air, (b/b)	%	Maks. 13,0	Maks. 1,0
3	Kadar biji enteng, (b/b)	%	Maks. 1,0	Maks. 2,0
4	Kadar benda asing, (b/b)	%	Maks. 1,0	Maks. 2,0
5	Kadar lada berwarna kehitam-hitaman, (b/b)	%	Maks. 1,0	Maks. 2,0
6	Kadar cemaran kapang, (b/b)	%	Maks. 1,0	Maks. 3,0
7	Salmonela	Detection/25 g	Negatif	Negatif
8	E. coli	MPN/g	< 3	< 3

Pada table 2.1 diatas digunakan untuk penelitian ini. karena disesuaikan dengan kebutuhan penelitian untuk meneliti kualitas lada putih. Untuk menetukan mutu lada kadar lada putih yang berwarna kehitam-kehitaman dinyatakan dalam persentase bobot per bobot dengan perhitungan sebagai berikut:

$$\frac{M_1}{M_0}\ x\ 100$$

Keterangan:

M_0: adalah bobot contoh uji (g)

M_1: adalah bobot lada putih berwarna kehitam-hitaman(g)

Dari hasil perhitungan diatas akan ditemukan hasil berapa besar kandungan lada dengan warna kehitam-hitaman.

2.3 Pengolah Citra

Pengolah citra adalah teknik perangkat lunak yang digunakan untuk memproses dan menganalisis gambar atau citra. Pengolah citra melibatkan berbagai operasi seperti pembersihan citra, perbaikan kualitas citra, ekstraksi fitur, segmentasi objek, dan analis kontur citra [4]. Teknik pengolahan yang mentransformasikan inputan yang berupa citra dan memprosesnya menjadi data numerik. Hal ini berupaya supaya komputer mudah untuk membaca data yang nantinya akan dibandingkan oleh data masukan lain. Ini disebut proses deteksi, dalam proses ini gambar atau citra akan dipahami identitasnya dengan membaca fitur-fitur yang ada dari objek pada gambar. Dari proses tersebutlah perbandingan antara fitur input dan fitur referensi digunakan untuk menentukan hasil yaitu apakah objek yang terdeteksi sama seperti pada fitur refrensi atau tidak [5], [6].

2.4 Convolution Neural Network

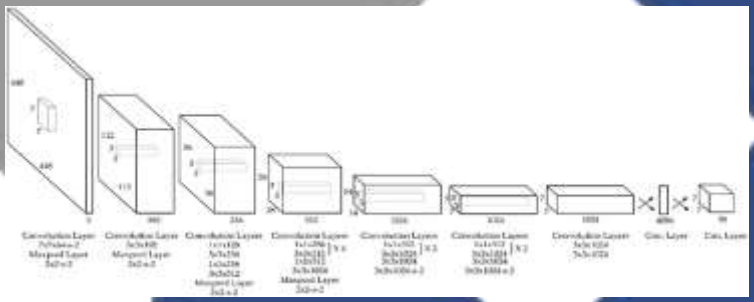
Convolutional Neural Network adalah metode dalam pengolah citra yang mengolah data dua dimensi dalam bentuk citra menggunakan pengembangan dari multilayer perceptron (MLP). CNN pertama kali diperkenalkan oleh Yann LeCun pada tahun 1998 dalam penelitiannya yang berjudul *"Gradient-Based Learning Applied to Document Recognition"*. CNN masuk dalam katergori *Deep Neural Network* karena memiliki kontruksi jaringan yang berlapis-lapis [7].

Gambar 2.1 Arsitektur *Convolutional Neural Network* [6]

Pada gambar 2.1 diatas dapat ditampilkan arsitektur dari *Convolutional Neural Network*. Dengan kontruksi yang berlapis-lapis ini, didalam CNN memiliki beberapa seksi lapisan seperti *Input Layer, Hidden Layer,* dan *Output Layer* [8], [9].

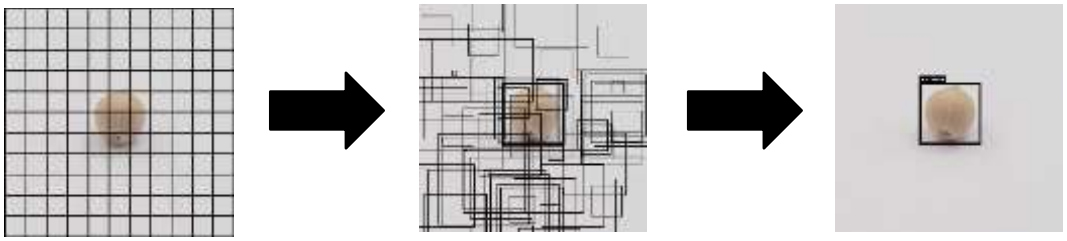
2.5 *You Only Look Once* (*YOLO*)

YOLO adalah salah satu algoritma yang digunakan dalam pendeteksi objek secara real-time. *YOLO* memungkinkan deteksi objek secara *real-time* dengan akurasi yang baik seperti mendeteksi objek yang homogen dan heterogen. Ini dikarenakan *YOLO* memiliki sistem deteksi berbasis CNN [10]. Dengan mengunakan repurpose classifier atau localizer untuk melakukan pendeteksian, model ini diterapkan pada sebuah citra dibeberapa lokasi dan skala. Pada proses ini citra dibagi menjadi beberapa bagian grid [7]. Dan saat pendeteksian daerah dengan skor paling tinggi akan bertanggung jawab sebagai sebuah pendeteksian. Sebelum masuk pada proses pelatihan, gambar yang menjadi bahan untuk *dataset* harus dianotasikan terlebih dahulu membentuk *dataset*. Poses ini adalah guna menetukan nama kelas, menentukan titik koordinat X dan Y pada objek, dan membuat besaran kotak pembatas atau disebut *bounding box* [11].



Gambar 2.2 Arsitektur *YOLO*

Pada gambar 2.2 diatas dapat dijelaskan bahwa *YOLO* memiliki arsitektur yang mirip dengan *Convolutional Neural Network*. Dan untuk *output layer* disesuaikan dengan jumlah kelas dan jumlah kotak prekdisi yang diinginkan[8], [12].



Gambar 2.3 Proses Kerja *YOLO*

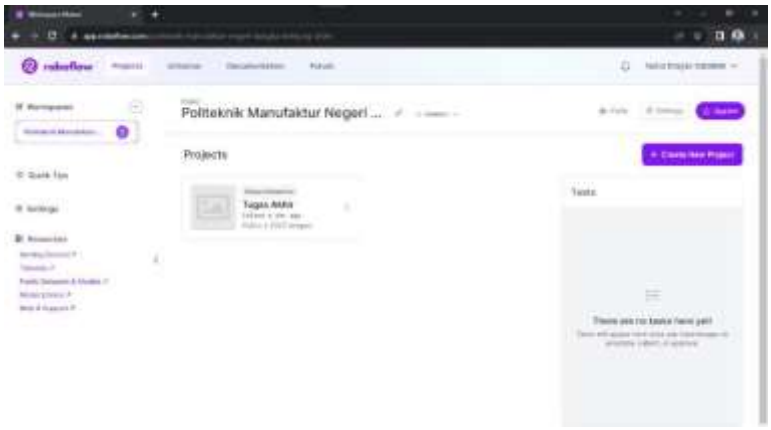
Pada gambar 2.3 diatas dapat dijeaskan sebuah inputan yang kita asumsikan real-time akan dibagi beberapa kali beberapa grid. Ini adalah proses awal dari *YOLO* berkerja, selanjutnya setiap grid akan megetahui apakah titik tengah dari kordinat pada objek dan membuat objek dengan ground trut besar akan membentuk *bounding box* setelah itu bonding box dengan tingkat kebenaran rendah akan dihilangkan dan menampilkan *bounding box* dengan tingkat kebenaran tinggi [6].

2.6 Tools

Dalam pengerjaan penelitian ini tools adalah perangkat yang mampu menungjang dalam membuat penelitian ini. Perangkat yang digunakan adalah perangkat lunak dan perangkat keras. Tools ini digunakan pada pembuatan kode pemograman, melatih model, mengubah bobot sesuai kebutuhan serta digunakan untuk uji coba dan presentasi alat yang sudah jadi [13].

2.6.1 Roboflow

Roboflow adalah *webside* untuk *dataset*. Dalam *webside* ini tugas-tugas untuk membuat *dataset* dapat dilakukan seperti pengumpulan data gambar, labelling, anotasi, augmentasi, *preprocessing* data, dan masih banyak lagi. *Roboflow* memiliki fitur kemudahan seperti *Smart Polygon* untuk menganotasikan objek hanya dengan mengklik titik tengah objek tersebut dan anotasi gambar berjalan otomatis. Dan *Label Assist* dimana fitur ini bisa meng anotasi secara otomatis objek pada gambar tampa harus melakukan apa-apa, yang dilakukan hanya cukup dengan menlanjutkan gambar mana dari kumpulan data gambar yang ingin dianotasikan. *Label Assist* berkerja dari melakukan hasil prediksi mengunakan moden yang sebelumnya sudah dibuat [9].

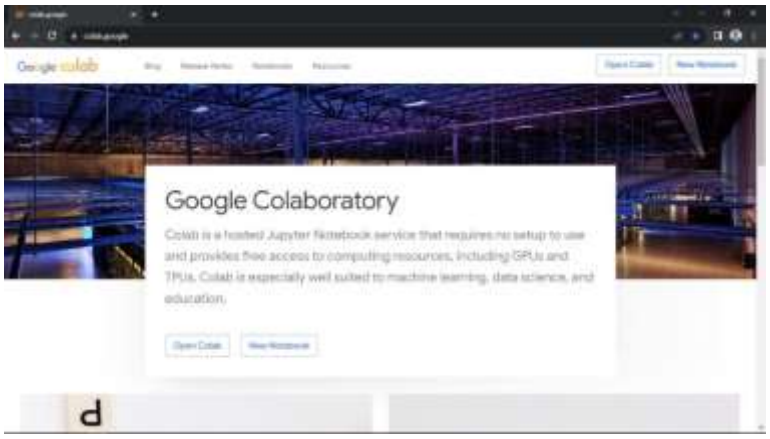


Gambar 2.4 *Webside Roboflow*

Dari gambar 2.4 bisa dilihat tampilan beranda dari *webside Roboflow*.com dimana pada pojok kiri atas dibawah button upload terdapat button *Create New Project*. Pilihan inilah yang pertama kali kita gunakan untuk membuat proyek baru dimana didalamnya akan ada beberapa fitur perintah dan salah satunya fitur perintah upload. Pada fitur perintah tersebut kita kan dibawa kedalam halaman untuk mengupload data gambar yang kita miliki untuk digunakan sebagai data gambar yang akan dilabelling dan anotasi serta dijadikan *dataset* [14].

2.6.2 Google Colaboratory

Google Colaboratory adalah salah satu IDE berbasis *webside* dimana memiliki fungsi membangun, mengeksekusi, dan mengkonfigurasi suatu kode pemograman komputer tanpa perlu menginstall lingkungan pendukung. Dengan berbasis *webside* fitur-fiturnya sudah lengkap dan tidak perlu menyiapkan spesifikasi perangkat yang memadai. Ini karena google colaboratory memiliki Ram, CPU dan *GPU*nya sendiri. Sehingga aktifitas yang diolah didalam *Google Colaboratory* tidak terganggu akibat spesifikasi perangkat yang terbatas[6].



Gambar 2.5 *Webside Google Colaboratory*

Pada gambar 2.5 diatas menampilkan halaman beranda dari *webside Google Colaboratory*. Di*webside* inilah aktifitas pengolahan program dilakukan. Salah satunya untuk *training dataset*. Pada halam beranda terdapat 2 pilihan perintah, yang pertama yaitu *open colab*. Fitur perintah *open colab* adalah digunakan jika sidah memiliki projek yang dibuat dan ingin dilanjutkan lagi progress yang dilakukan. Lalu yang kedua ada fitur perintah *new notebook*, fitur perintah ini digunakan jika penggunakan baru ingin membuat proyek baru [6].

2.6.3 Anaconda Navigator

Anaconda Navigator adalah *Graphical User Interface (GUI) desktop* yang disertakan dalam *Anacoda Distribution* yang dapat digunakan untuk menjalankan aplikasi dan mengelola *conda package*, *environments* dan *channel* tanpa *(CLI)*. Didalam *anaconda navigator* ini sudah ada ekosistem yang terintegrasi. *Software* ini sangat membatu dalam penelitian karena tidak perlu menginstall kebutuhan package secara terpisah tetapi bisa langsung diinstall dalam anaconda navigator tersebut [15].

Gambar 2.6 *Software Anaconda Navigator*

Pada gambar 2.6 kita dapat melihat tampilan halaman beranda dari *software* ini. Dimana didalam *software* ini sudah tersedia aplikasi yang berbagai macam fungsi dan bisa digunakan sesuai dengan jenis pengolahan progam. *Software* memiliki kelebihan yaitu segala fitur yang ada didalam yang sudah disediakan atau ditambah sesuai yang dibutuhkan dan terkoneksi satu sama lain. Sehigga tidak perlu menginstall kebutuhan fitur secara terpisah.

2.6.4 Library

Dalam pemrograman komputer *Library* adalah kumpulan kode yang telah ditulis sebelumnya yang dapat digunakan kembali untuk melakukan tugas-tugas tertentu dalam pengembangan perangkat lunak. *Library* mengandung fungsi-fungsi, kelas-kelas, atau modul-modul yang dapat diintegrasikan ke dalam program. untuk memperluas fungsionalitas atau memecahkan masalah tertentu tanpa perlu menulis kode dari awal. Dengan tersedianya kode yang dapat digunakan kembali memungkinkan untuk dapat menghemat waktu dan upaya dengan tidak perlu menulis ulang fungsi-fungsi atau algoritma-algoritma. Tetapi terkadang ketersediaan *Library* juga harus disesuaikan dengan kebutuhan atau harus diedit Kembali sesuai dengan kebutuhan fungsi dan tugas dari penelitian [16].

2.6.4.1 Ultralytics

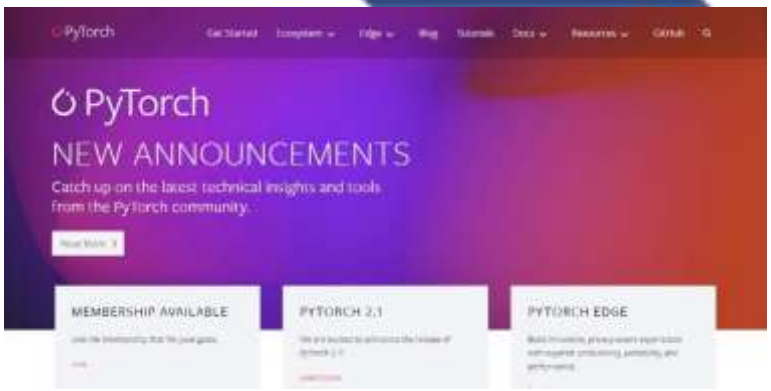
Ultralytics adalah *Library* algoritma *YOLO* yang dibuat oleh *Ultralytics*. Pada *Library* ini terdapat beberapa modul termasuk modul versi dari *YOLO* tersebut. Dalam modul ini juga fitur-fitur yang digunakan dalam memproses *YOLO* terdapat.

Gambar 2.7 *Utralytics*

Pada gambar 2.7 dapat ditampilkan dari *webside Ultralytics*. Dalam *webside* ini terdapat informasi, modul, dan communitas dari developer-developer yang membahas tentang penggunakan *YOLO*[10].

2.6.4.2 PyTorch

PyTorch atau yang sering disebut "*torch*" adalah sebuah libray machine learning yang dikembangakan oleh *Facebook*. *PyTorch* dirancang untuk mendukung pengembangan model-model machine learning dan deep learning.



Gambar 2.8 *PyTorch*

Pada gambar 2.8 diatas dapat ditampilkan *webside PyTorch*. Pada umumnya dalam *webside* ini terdapat sumber yang dapat didapatkan guna menggunakan *Library torch* dari *PyTroch* [7].

2.6.5 Web Camera (Webcam)

Web camera atau *webcam* adalah kamera internal yang menjadi bagian dalam suatu perangkat computer. Tetapi webcam bisa juga dipasang pada computer atau yang sering disebut kamera eksternal. Fungsi kamera webcam biasanya digunakan untuk komunikasi citra antar perangkat atau dalam perangkat computer itu sendiri. Dimana pada penelitian ini sendiri webcam berfungsi sebagai penangkap citra untuk selanjutnya diproses pada algoritma [17].

2.6.5.1 Irium Webcam

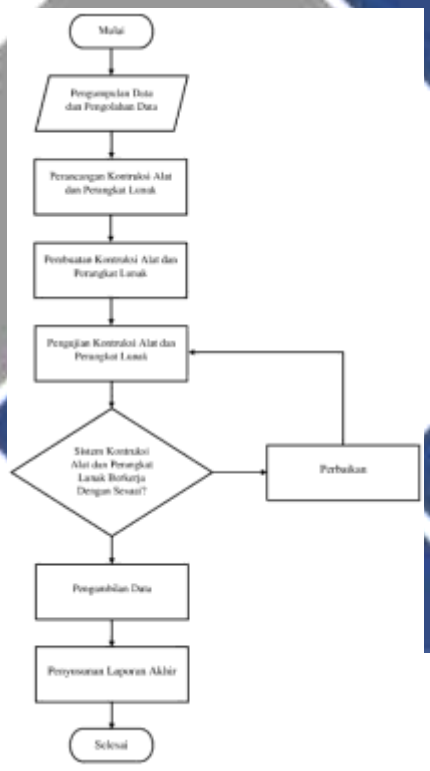
Irium Webcam adalah *software* webcam yang ada di *Windows* dan *Android*. *Software* ini memiliki fungsi menyambungkan kamera smartphone menjadi kamera yang terhubung pada komputer dan mengantikan fungsi *webcam* pada kamera komputer. Kelebihan Irium Webcam terdapat fitur auto fokus pada layanan gratisnya.

Gambar 2.9 Iriun Webcam

Pada gambar 2.9 dapat ditampilkan layar yang menampilkan tangkapan dari kamera eksternal. Terdapat 2 metode sambungan pada *software* Iriun yaitu, *wireless* dan *wired*. Dimana untuk sambungan *wireless* bisa menggunakan *wi-fi* [3].

BAB III
METODE PELAKSANAAN

Dengan metode pelaksaan yang terstruktur maka proses pembuatan proyek akhir akan mencapai pada tujuan maksimal. Proyek akhir yang berjudul "Implementasi Pengolah Citra Pada Prototipe Alat Penyortir Lada" dibuatkanlah alur-alur langkah kerja dimana ini akan menjadi landasan langkah kerja yang dilaksanakan. Alur-alur langkah kerja ini akan secara spesifik dijelaskan dalam bentuk *flowchart* dibawah ini:



Gambar 3.1 *Flowchart* Metode Pelaksanaan

Pada gambar 3.1 dapat dijelakan tahapan metode penelitian dimulai dari pengumpulan dan pengolahan data, proses perancangan kontruksi dan perangkat lunak, pembuatan kontruksi alat dan perangkat unak, pengujian kontruksi dan perangkat lunak, perbaikan jika diperlukan, pengambilan data dan yang terakhir penyusunan laporan akhir.

3.1 Pengumpulan dan Pengolahan Data

Pengumpulan dan pengolahan data adalah langkah awal dari penelitian ini. Dalam garis besarnya pengumpulan data memlalui studi literatur, survei dan konsultasi. Proses ini dilakukan oleh bersama dosen pembimbing dimana dalam hal hal yang dilakukan adalah pengumpulan data kualitatif untuk menentukan bahan baku, material kontruksi, sistem kerja yang digunakan, dan metode penelitian yang dilakukan. Sehingga bahan-bahan yang disediakan sesuai dengan yang dibutuhkan. Survei dilakukan untuk mendapatkan literasi tentang bagaimana para petani memproses lada putih dan aspek-aspek penilaian terhadap idetifikasi lada putih. Hal ini didasari oleh sifat proyek akhir yang tidak boleh memberatkan dalam segi sistem alat dan segi penggunaan biaya.

3.2 Perancangan Kontruksi Alat Dan Perangkat Lunak

Perancangan kontruksi alat dan perangkat lunak langkah yang dilakukan untuk menentukan bagaimana dan seperti apa kontruksi alat yang ingin dibuat. Untuk perancangan kontruksi alat mengunakan *software* menggambar. Gambar dibuat secara 3 dimensi agar mampu menampilkan bentuk secara keseluruhan. Rancangan ini menjadi dasar pembuatan alat dengan dimensi sesungguhnya.

Gambar 3.2 Rancangan Alat Tampak Samping

Pada gambar 3.2 diatas dapat ditampilkan rancangan kontruksi alat dengan perpektif tampak samping. Dalam gambar tersebut dapat dijelaskan beberapa fungsi dari kontruksi yang ditandai oleh anak panah sebagai berikut:

- Tempat masuk lada berfungsi sebagai tempat dimana lada akan dimasukan pada pengayak dan menjadi Langkah awal dalam mengunakan alat tersebut.
- Motor pengayak adalah motor dc yang akan menggerakan pengayak dimana terdapat sistem mekanik pengubah Gerakan mekanik rotasi menjadi Gerakan mekanik linier.
- Roda mekanik berfungsi sebagai media gerak yang menciptakan penurunan gaya gesek menjadi nol.
- Motor konveyor adalah motor dc yang berfungsi sebagai penggerak konveyor.

Pada kontruksi ini kendali motor akan dikendalikan oleh sistem kontrol berbasis mikrokontroller.

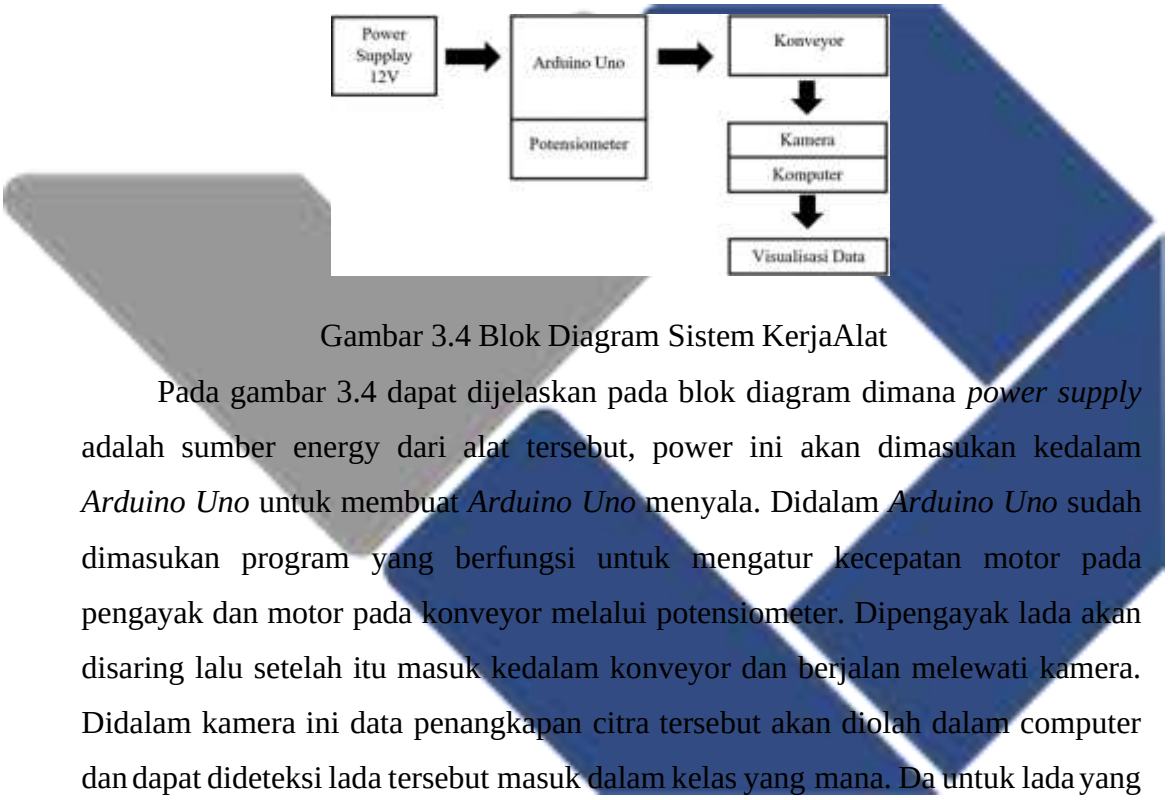
Gambar 3.3 Rancangan Alat Tampak Depan

Pada gambar 3.3 diatas dapat ditampilkan kontruksi alat dengan perpeltif tampak depan dan dijelaskan komponen kontruksi sebagai berikut:

- Corong ke layer 2 adalah kontruksi penyalur lada yang berpindah dari layer 1 ke layer 2.

- Corong ke konveyor adalah komponen kontruksi penyalur laya yang sudah disaring dari pasir dan menuju konveyor.
- Kamera adalah komponen kontruksi yang berfungsi sebagai media pengambilan citra.
- Konveyor adalah media penggerak lada yang nantinya akan melewati area pengambilan citra oleh kamera dan dibawa ke wadah kemas.

Adapun blok diagram rancangan alat ada pada gambar dibawah ini:



Gambar 3.4 Blok Diagram Sistem KerjaAlat

Pada gambar 3.4 dapat dijelaskan pada blok diagram dimana *power supply* adalah sumber energy dari alat tersebut, power ini akan dimasukan kedalam *Arduino Uno* untuk membuat *Arduino Uno* menyala. Didalam *Arduino Uno* sudah dimasukan program yang berfungsi untuk mengatur kecepatan motor pada pengayak dan motor pada konveyor melalui potensiometer. Dipengayak lada akan disaring lalu setelah itu masuk kedalam konveyor dan berjalan melewati kamera. Didalam kamera ini data penangkapan citra tersebut akan diolah dalam computer dan dapat dideteksi lada tersebut masuk dalam kelas yang mana. Da untuk lada yang telah melewati kamera akan masuk kedalam wadah kemas yang nantinya akan dikemas.

3.3 Pembuatan Kontruksi Alat

Pembuatan kontruksi Alat dilakukan diworkshop dan dilaboratorium kampus. Proses yang meliputi pembelian bahan dilakukan secara langsung dan online. untuk perakitan dilakukan diworkshop dan dilaboratorium kampus, meliputi pemasangan rangka alumunium profile untuk perakitan kerangka, pemasangan saringan berupa

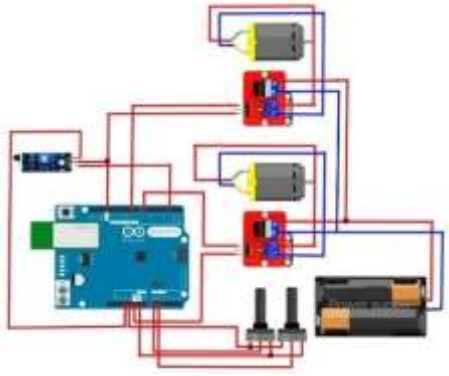
jaring-jaring alumunium dipasang pada kerangka pengayak, dan pemasangan instalasi kelistrikan. Untuk pembuatan skematik dan *sistem kontrol* motor dc mengunakan *software Arduino IDE* karena sistem kontrol mengunakan *Arduino Uno* sebagai mikrokontroller dan mosfet sebagai pengatur tegangan. Pembuatan perangkat lunak berupa program computer mengunakan *Anaconda Navigator*, *Visual Studio Code*, *Google Colaboratory* dan dibantu oleh beberapa *webside* terkait.

3.3.1 Pembuatan Kontruksi Alat

Pembuatan Kontruksi alat meliputi perakitan kerangka dan pembuatan konveyor. Proses ini meliputi pembuatan kerangka, pembuatan konveyor, pemasangan instalasi kelistrikan dan perakitan kontruksi alat.

3.3.2 Pembuatan Rangkaian Kelistrikan Alat

Pembuatan rangkaian kelistrikan berupa rangkaian sistem kontrol dengan mikrokontroller menggunakan *Arduino Uno*. Permograman sistem kontrol dibuat dalam *software Arduino IDE*. Sebelum itu untuk membuat rangkaian guna tidak terjadi kesalahan, digunakanlah aplikasi Tinkercad yang membuat alat dapat diuji terlebih dahulu tanpa harus mengunakan alat sesungguhnya.



Gambar 3.5 Rangkaian Sistem Kontrol

Pada gambar 3.5 diatas dapat dijelaskan rangkaian sistem kontrol pada kontruksi alat yang berfungsi untuk mengatur jumlah kecepatan motor yang ada dipengayak dan konveyor.

3.3.3 Pembuatan Program Kontrol Motor

Gambar 3.6 Program Kontrol Motor

Pada gambar 3.6 dapat dijelaskan program untuk mengatur kecepatan motor yang digunakan pada pengayak dan konveyor. Kecepatan motor akan dipengaruhi oleh besaran value pada potensiometer. Dapat dijelaskan fungsi dari kode program seperti berikut:

```
#include "HCMotor.h"
#define MOTOR_PIN 6
#define POT_PIN A0
```

Gambar 3.7 Kode *Library*

Pada gambar 3.7 diatas dapat dijelaskan program #include “HCMotor.h” yang berfungsi untuk mengimport *Library* HCMotor.h. Kode #define MOTOR_PIN 6 yang berfungsi mendeklarasikan pin 6 sebagai pin digital yang terhubung pada motor. Dan kode #define POT_PIN A0 berfungsi mendeklarasikan pin A0 sebagai pin analog yang terhubung pada inputan dari potensiometer.

```
HCMotor HCMotor;
```

Gambar 3.8 Kode Objek

Pada gambar 3.8 diatas dapat dijelaskan bahwa objek HCMotor dibuat

```
void setup() {
HCMotor.Init();
HCMotor.attach(0, DCMOTOR, MOTOR_PIN);
HCMotor.DutyCycle(0, 100);
}
```

Gambar 3.9 *Void Setup*

Pada gambar 3.9 diatas dapat dijelaskan kode yang berfungsi sebagai seting yang ada pada mikrokontroller dan hanya sekali eksekusi. Pada kode ini terdapat beberapa perintah. Dimana HCMotor.Init(); berfungsi sebagai penginisialisasian objek HCMotor. Kode HCMotor.attach(0, DCMOTOR, MOTOR_PIN); berfungsi mengidentifikasi MOTOR_PIN yang digunakan pada objek HCMotor terhubung pada nomor 0 dan DCMOTOR teridentifikasi mengunaan motor dc. Kode HCMotor.DutyCycle(0, 100) mengatur siklus tugas motor ke 100% untuk motor yag teridentifikasi sebagai nomor 0.

Gambar 3.10 *Void Loop*

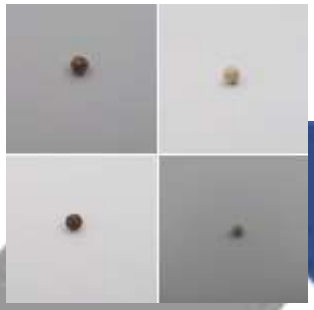
Pada gambar 3.10 diatas dapat dijelaskan kode yang menjalankan secara berulang. Dimana dalam void loop terdapat kode int speed; yang berfungsi mendeklarasikan variable interger bernama speed. Kode Speed = map (analogRead(POT_PIN), 0, 1024, 0, 100); berfungsi membaca nilai analog yang dikeluarkan oleh potensiometer dengan rentan 0 – 1024 menjadi rentan 0 – 100. Kode HCMotor.OnTime(0, Speed); berfungsi mengatur waktu aktif pada motor yaitu on time atau sesuai waktunya pada motor dengan nomor 0.

3.4 Pebuatan Perangkat Lunak

Pembuatan perangkat lunak mengunakan *software* Anaconda Navigator, *Google Colaboratory*oratory, dan *Visual Studio Code*. *Anaconda Navigator* berfungsi sebagai membuat ekosistem yang nantinya fitur-fitur akan digunakan. Fitur-fitur ini meliputi environment, *Library*, dan Visual studio code yang terhubung. Untuk proses pelatihan mengunakan *Google Colaboratory*.

3.4.1 Pengumpulan Data Gambar

Pengumpulan Dataset adalah proses pengumpulan citra berupa gambar yang didapatkan dari pemotretan langsung secara mandiri. Setelah dataset terkumpul selanjutnya dilakukan proses pelebelan dan anotasi gambar. Data gambar lada yang diambil dan dikumpulkan berjumlah sebanyak 548 gambar per kelasnya. Dalam 1 kelas memiliki sebanyak 274 gambar. Beikut gambar dari sempel data gambar lada:



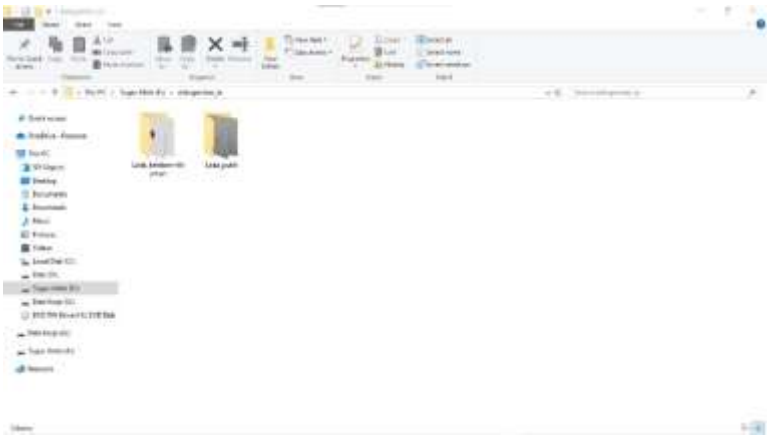
Gambar 3.11 Data Gambar

Pada gambar 3.11 ditampilkan data gambar lada. Dan dijelaskan jumlah lada pada tabel dibawah ini:

Tabel 3.1 Tabel Jumlah Data Gambar

Kelas	Jumlah Gambar	Keterangan
Lada Putih	274	Gambar lada dengan lada yang putih secara keseluruhan
Lada Kehitam-hitaman	274	Gambar Lada memiliki warna kehitam-hitaman dan hitam
Total	**548**	

Tabel 3.4 diatas menjelaskan bahwa banyaknya data gambar disetiap kelas. Lalu ata gambar lada yang sudah dimasukan kedalam masing-masing folder disimpan difolder yang diberi nama sesuai dengan kelasnya. Berikut gambar lokasi penyimpanan folder data gambar lada.



Gambar 3.12 Folder Kelas

Pada gambar 3.12 dapat dijelaskan lokasi penyimpana folder-folder kelas data gambar berada pada Tugas Akhir(E:) dengan folder Bernama datagambar_ta. Penyimpanan ini mengunakan hardisk eksternal guna untuk tidak terlalu banyak memakan penyimpanan internal dari perangkat. Dalam folder datagambar_ta terdapat 2 folder kelas yang diberikan nama sesuai kelasnya lada_putih, dan lada_kehitam-hitaman. Dalam masing-masing folder terdapat file-file gambar dari data gambar tersebut sesuai kelas masing-masing.

3.4.2 Labeling & Anotasi Data Gambar

Poses labeling dan anotasi data gambar mengunakan aplikasi *Roboflow*. Pada proses ini data gambar tersebut akan dilabeli sebagai jenis yang sesuai kelas dari data tersebut. Lalu dianotasikan seperti dimarking untuk menentukan bagian gambar mana yang digunakan sebagai *dataset*. Sebelum tahap ini diakukan seluruh gambar diupload kedalam direktori yang nantikan proses labelling dan anotasi dilakukan [9].

Gambar 3.13 Proses Labeling Dan Anotasi Gambar

Pada gambar 3.13 dapat dijelaskan dari proses labeling dan anotasi. Gambar lada yang sudah dianotasikan mengunakan fitur *smart polygon* yang dimiiki oleh *webside Roboflow* mendapatkan hasil berupa garing anotasi yang mampu mengetahui bentuk dari objek dan mengikutinya. Ini tidak umum dilakukan karna pada kebanyakan orang hanya mengunakan kotak persegi sebagai anotasi.

Selanjutnya hasil anotasi akan berupa file label dimana didalam file tersebut menjelaskan kelas dan koordinat dari objek yang sudah dianotasi didalam gambar tersebut. Isi file tersebut dapat ditampilkan sebagai berikut:

Gambar 3.14 File Label Dan Anotasi Gambar

Pada gambar 3.14 diatas dapat dijelaskan bahwa proses labelling dan anotasi akan membuat file berupa kumpulan angka. Tetapi dari kumpulan angka tersebut dapat dijelaskan bahwa 2 adalah kelas dari objeknya lalu barisan angka 0. Adalah titik dimana anotasi itu dibuat dalam objek. Kalau menggunakan *bounding box* persegi maka hanya ada empat barisan angkta 0,. Tetapi karna penelitian ini tidak menggunakan *bounding box* yang membentuk persegi tetapi mengunakan *smart*

polygon dimana bentuknya menyesuaikan bentuk objeknya. Maka barisan anggka 0, juga mengikuti jumlah titik yang ada pada *bounding box* diobjek. Akan tetapi untuk mengetahui file-file tersebut harus menyelesaikan proses labelling dan anotasi terlebih dahulu. Ini berlaku jika menggunakan *webside Roboflow*.com. ini akan berbeda jika menggunakan media labelling dan anotasi yang berbeda.

3.4.3 Pembuatan *Dataset*

Pembuatan *dataset* dilakukan setelah proses labelling dan anotasi gambar selesai. Maka data gambar siap dibuat menjadi *dataset*. Akan tetapi sebelum masuk dalam pembuatan *dataset*, *dataset* mentah tadi akan diolah lagi dalam beberapa tahap untuk mendapatkan *dataset* yang baik.

3.4.3.1 Pembagian Komponen Data

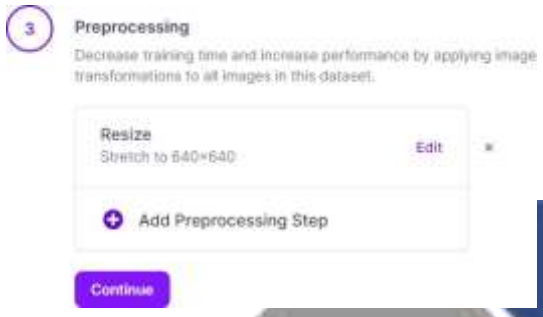
Dataset mentah dibagi lagi menjadi 3 bagian set yaitu *Test Set*, *Valid Set*, dan *Train Set*. Pembagian ini ditentukan sebagai 70% *Test Set*, 20% *Valid Set*, 10% *Train Set*. Walaupun penentuan persentase tersebut tidak baku dan bisa ditentukan sesuai kebutuhan.

Gambar 3.15 Pembagian Komponen Data

Pada gambar 3.15 dijelakan untuk gambar yang sudah dilabeling dan anotasi makas selanjutnya akan dibagi menjadi 3 bagian yaitu *train set*, *valid set*, dan *test set*. Ini digunakan saat pelatihan berlangsung. Jika selesai proses *training* makan ada beberapa data *Result* yang muncul dan menjadi indicator apakah model terbuat dengan baik atau masih harus diperbaiki.

3.4.3.2 Preprocessing

Selanjutnya data gambar tersebut akan diresize. Sehingga data gambar yang memiliki variasi dalam ukuran akan dibentuk dalam satu ukuran, pada penelitian ini resize yang dibuat sebesar 640 x 640 *pixel*.

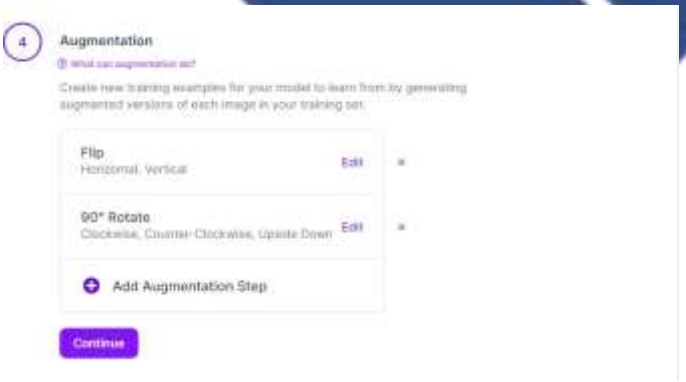


Gambar 3.16 Proses *Preprocessing*

Pada gambar 3.16 dijelaskan Langkah selanjutnya dalam mensetup *dataset* yaitu *preprocessing*. Pada proses ini memiliki beberapa fitur yang dapat digunakan salah satunya adalah fitur resize. Fitur ini memotong gambar dari titik tengah objek yang dianotasikan lalu dipotong sebesar ukuran yang dipilih. Ini berguna supaya benda yang kecil tidak membawa background gambar yang berlebih.

3.4.3.3 Augmentation

Augmentation proses dimana *dataset* akan dilipat gandakan jumlahnya menjadi beberapa kali lipat untuk menambah jumlah *dataset*.



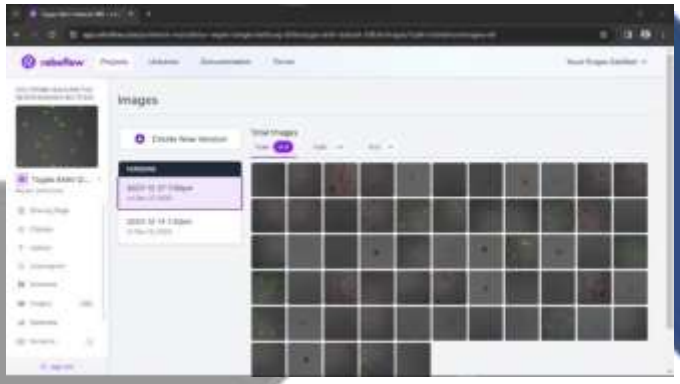
Gambar 3.17 Proses *Augmentation*

Pada gambar 3.17 dapat dijelaskan yaitu proses *Augmentation*. Proses ini membuat lebih banyak gambar atau menyalinya dengan ditambah beberapa

indikator seperti dibalik, diputar searah jarum jam, dan diputar berlawannan jarum jam. Fungsi proses ini tentunya untuk menambah data set yang tersedia sehingga komposisi data gambar menjadi bertambah.

3.4.3.4 Pembuatan *Dataset*

Pembuatan *dataset* yang sudah melalui beberapa tahap dan menghasilkan *dataset* yang siap digunakan dengan jumlah *dataset* menjadi sebanyak 1295 gambar.

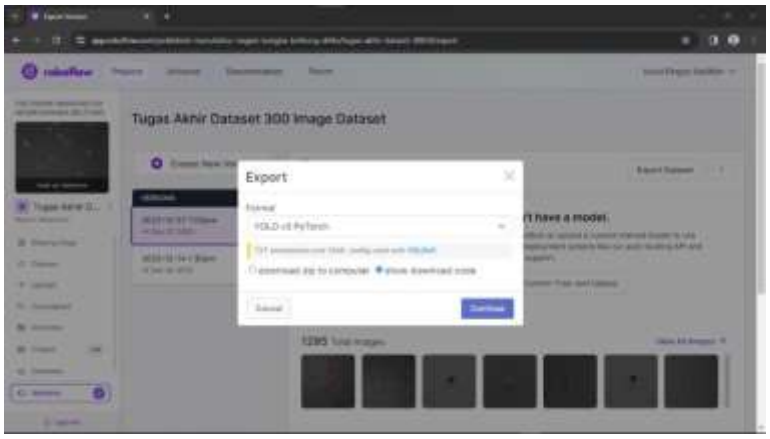


Gambar 3.18 Hasil Pembuatan *Dataset*

Dapat dilihat pada gambar 3.18 diatas bahwa seluruh datagambar sudah diproses menjadi *dataset* dan akan degenerate sesuai dengan algoritma yang digunakan.

3.4.3.5 Export Dataset

Generate dataset adalah proses dimana *dataset* dibuat sesuai dengan algoritma yang digunakan. Sebab setiap algoritma mengunakan tipe *dataset* yang berbeda beda.



Gambar 3.19 *Export Dataset*

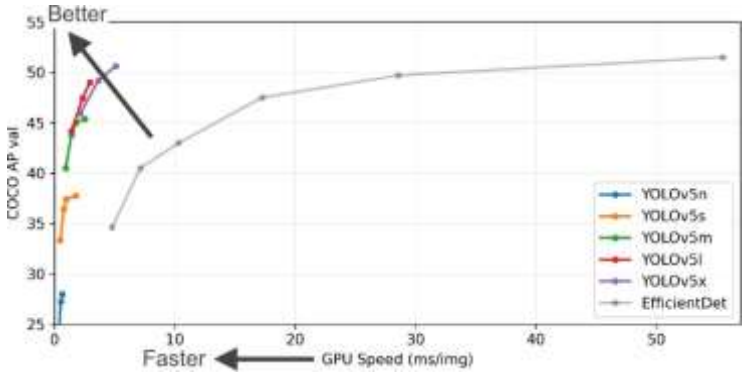
Dapat dilihat pada gambar 3.19 diatas proses *export* dengan tipe *dataset YOLO* v5 *PyTorch*. Ada beberapa tipe *dataset* dari beberapa algoritma yang tersedia sehingga dapat dipilih sesuai dengan kebutuhan yang ada.

3.4.4 Pelatihan *Dataset*

Pelatihan *dataset* dilakukan mengunakan *Google Colaboratoryoratory*. Mengunakan sebanyak 100 epoch dengan bobot *YOLOv5m*. didalam *YOLOv5* memiliki beberapa tingkatan bobot yang dapat digunakan. Tingkatan bobot ini mempengaruhi hasil model jadi terhadap kinerja fitur-fitur pada *YOLOv5*. Dibawah ini dapat dilihat ilustrasi dari konvolusi pada setia bobot dalam *YOLOv5* sebagai berikut:

Gambar 3.20 Visualisasi Fitur Dari Setiap Bobot *YOLOv5*

Pada gambar 3.20 diatas dapat ditampilkan visualisasi fitur dari setiap bobot dalam *YOLOv5*. Dari data tersebut dapat menentukan bobot mana yang cocok dalam melakukan *training* model. Ini menjadi dasar bagaimana sebuah model yang akan di*training* dengan bobot yang sesuai kebutuhan.



Gambar 3.21 Grafik Umum Performa Kecepatan *GPU* Setiap Bobot

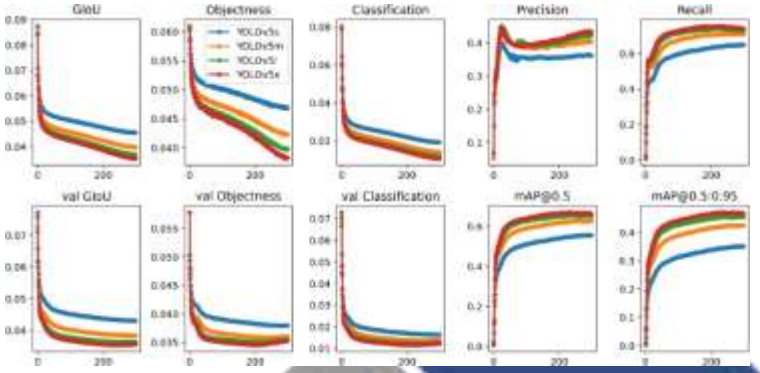
Pada gambar 3.21 diatas dapat dijelaskan grafik umum performa kecepatan penggunakan *GPU* setiap bobot dan perbandingannya.

Tabel 3.2 *Pretrained Checkpoint*

Model	**Size** (*pixels*)	**mAP^{val}** 50-95	**mAP^{val}** 50	**Speed** CPU b1 (ms)	**Speed** V100 b1 (ms)	**Speed** V100 b32 (ms)	**Params** (M)	**FLOPs** @640 (B)
*YOLOv5*n	640	28,0	45,7	45	6,3	0,6	1,9	4,5
*YOLOv5*s	640	37.4	56,8	98	6,4	0,9	7,2	16,5
*YOLOv5*m	640	45,4	64,1	224	8,2	1,7	21,2	49,0
*YOLOv5*l	640	49,0	67,3	430	10,1	2,7	46,5	109,1
*YOLOv5*x	640	50,7	68,9	766	12,1	4,8	86,7	205,7

Pada table 3.2 diatas dapat dijelaskan pretrained check point untuk setiap bobot *YOLO*. Dengan menggunakan bobot yang berlebihan hanya akan membuat penggunaan perangkat keras menjadi lebih besar karena fitur bobot yang besar akan

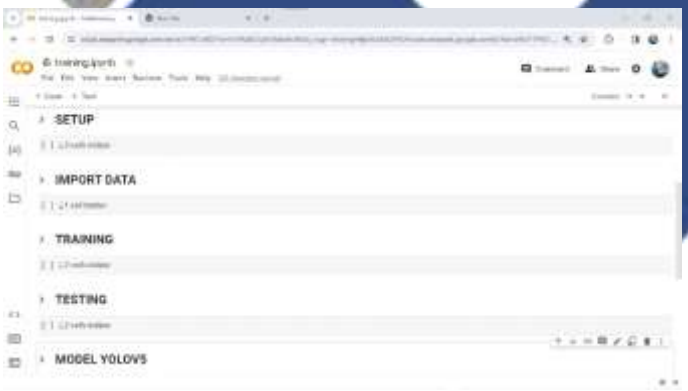
membuat proses *training* membutuhkan ruang yang luas. Hal ini juga dapat dijelaskadengan grafik *Result* spesifikasi dari setiap bobot dibawah ini:



Gambar 3.22 Grafik Performa Dari Setiap Bobot *YOLOv5*

Dapat dilihat pada gambar 3.22 setiap bobot memiliki performanya masing-masing. Dan dari grafik ini dapat ditentukan untuk memilih bobot yang tepat dilihat dari kemampuan perangkat keras, bobot *dataset*, dan jumlah epoch yang akan digunakan.

Untuk menjalankan proses *training* selain bobot juga harus mempersiapkan program untuk melakukan proses *training*.

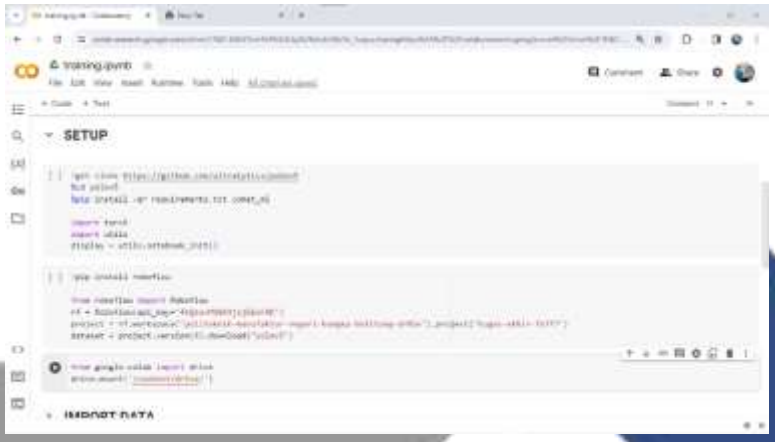


Gambar 3.23 Gambar Setup Program *Training* di *Google Colaboratory*

Pada gambar 3.23 diatas dapat dijelaskan setup program *training* yang dilakukan di*webside Google Colaboratory*. Pada program ini terdapat 3 bagian yaitu SETUP yang berfungsi sebagai setup awal untuk mempersiapkan modul *training* tersebut, IMPORT DATA yang berfungsi untuk mengambil *dataset* yang

dipersiapan, dan *TRAINING* yang berfungsi untuk lokasi *training dataset* dan membuatnya menjadi model.

Pada section SETUP terdapat beberapa kode pemograman yang dapat dilihat sebagai berikut:



Gambar 3.24 SETUP

Pada gambar 3.24 dapat ditampilkan kode pemograman yang terdapat pada section SETUP.

```
!git clone https://github.com/ultralytics/yolov5
%cd yolov5
%pip install -qr requirements.txt comet_ml

import torch
import utils
display = utils.notebook_init()
```

Gambar 3.25 Program Untuk Mengunduh Modul *YOLOv5*

Pada gambar 3.25 dapat dijelaskna program untuk mengunduh Modul *YOLOv5* yang akan digunakan.

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="fkQsovM96K9jsjEExFXK")
project = rf.workspace("politeknik-manufaktur-negeri-bangka-belitung-dr0ie").project("tugas-akhir-dataset-300")
dataset = project.version(1).download("yolov5")
```

Gambar 3.26 Program Untuk Mengunduh *Dataset* Dari *Roboflow*

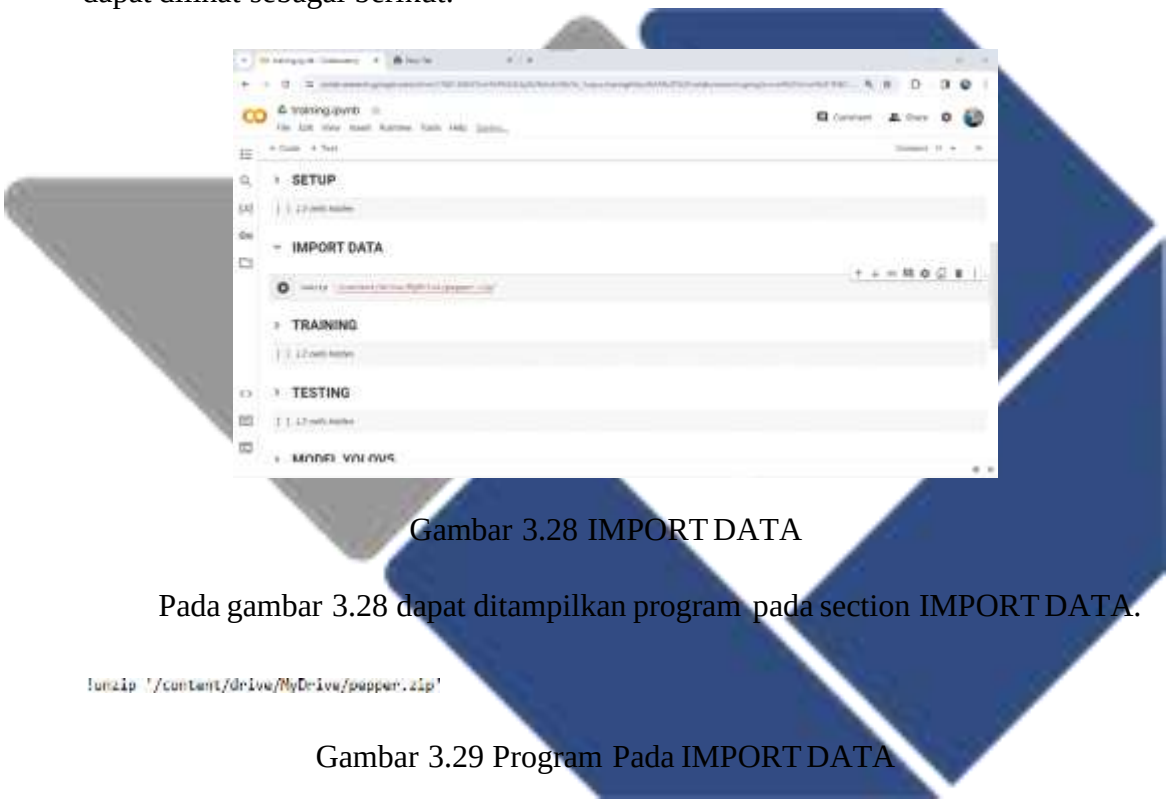
Pada gambar 3.26 diatas dapat dijelaskan program yang berfungsi untuk menguduh *dataset* yang sudah dibuat di*webside Roboflow*.

```
from google.colab import drive
drive.mount('/content/drive/')
```

Gambar 3.27 Program Mengkoneksikan Gdrive

Pada gambar 3.27 diatas dapat dijelaskan program untuk mengkoneksikan *Gdrive* untuk mengunduh beberapa file karena *Google Colaboratory* tidak dapat mengunduh file dikomputer secara langsung.

Pada section IMPORT DATA terdapat beberapa kode pemograman yang dapat dilihat sebagai berikut:



Gambar 3.28 IMPORT DATA

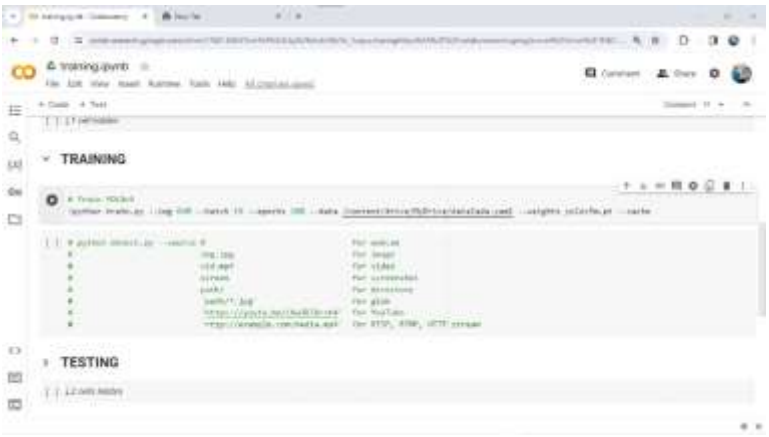
Pada gambar 3.28 dapat ditampilkan program pada section IMPORT DATA.

```
!unzip '/content/drive/MyDrive/pepper.zip'
```

Gambar 3.29 Program Pada IMPORT DATA

Pada gambar 3.29 dapat dijelaskan program untuk mengunzip data yang terdapat di gdrive.

Pada section *TRAINING* terdapat beberapa kode pemograman yang dapat dilihat sebagai berikut:



Gambar 3.30 *TRAINING*

Pada gambar 3.30 diatas dapat ditampilkan program pada section *TRAINING* dimana menjadi lokasi *dataset* dan bobot akan di*training*.

```
# Train YOLOv5
!python train.py --img 640 --batch 16 --epochs 1 --data /content/drive/MyDrive/datalada.yaml --weights yolov5m.pt --cache
```

Gambar 3.31 Program Pada *TRAINING*

Pada gambar 3.31 dapat dijelaskan program untuk men*training dataset*. Pada program ini mengunakan jumlah epoch sebanyak 100 dan mengunakan bobot yolov5m.

Gambar 3.32 Proses *Training Dataset*

Dapat dilihat pada gambar 3.32 diatas proses pelatihan yang berlangsung. Setelah proses *training* selesai output berupa file best.pt akan tersimpan secara sementara difolder run/train/exp seperti ditampilkan dibawah ini:

Gambar 3.33 Folder best.pt

Pada gambar 3.33 diatas dapat ditampilkan folder tempat file best.pt disimpan. Selanjutnya file tersebut akan langsung didownload karena *Google Colaboratory* hanya menyimpan file secara sementara. Dan dengan mengunakan file best.pt tersebut untuk digunakan dalam pengolah citra. Dengan kata lain best.pt adalah file dari model *YOLOv5* yang sudah ditraining.

3.4.5 Implementasi model kedalam program

Implementasi model yang sudah di*training* kedalam program yang dibuat sesuai tugas yang dilaksanakan. Pendeteksian jumlah adalah program yang dibuat untuk pengambilan data dengan variable jumlah lada dan tinggi kamera

Kode program pendeteksian jumlah sebagai berikut:

```
import cv2
import torch
import numpy as np
```

Gambar 3.34 Kode Import *Library*

Pada gambar 3.34 diatas dapat dijelaskan kode tersebut ini berfungsi untuk mengimport *Library* OpenCV sebagai cv2 untuk pemrosesan gambar, *PyTorch Library* yang berfungsi dalam melakukan eksekusi model deep learning dan *numpy* berfungsi sebagai *library* perhitung-hitungan.

```
model = torch.hub.load('ultralytics/yolov5', 'custom', 'epoch100yolov5l.pt')
```

Gambar 3.35 Kode Pengunduhan Model *YOLOv5*

Pada gambar 3.35 diatas dapatdijelaskan kode yang berfungsi sebagai pengunduh model *YOLOv5* dengan modul *YOLOv5* yang dibangun oleh developer

Ultralytics dan mengunduh model yang sudah di*training* berupa file yang Bernama epoch100yolov5l.pt.

```
cap = cv2.VideoCapture(0)
```

Gambar 3.36 Kode Inisialisasi Kamera

Pada gambar 3.36 diatas dapat dijelaskan kode perintah membaca kamera yang menjadi sumber pendeteksian objek. Dimana karna menggunakan kamera webcam maka perintah 0 yang digunakan.

```
while True:
```

Gambar 3.37 Kode Loop Pemrosesan Video

Pada gambar 3.37 dapat dijelaskan kode yang berfungsi sebagai looping program untuk menjalankan program selama kamera aktif.

```
ret, frame = cap.read()
```

Gambar 3.38 Kode Membaca Frame Kamera

Pada gambar 3.38 diatas dapat dijelaksan kode untuk membaca frame dari kamera yang mengunakan perintah ‘cap.read(). Untuk perintah ret akan menjadi variable true jika sumber kamera berhasil terbaca.

```
results = model(frame)
detections = results.pred[0]
classes_detected = detections[:, -1].cpu().numpy().astype(int)
```

Gambar 3.39 Kode Deteksi Objek

Pada gambar 3.39 diatas dijelaskan kode yang berfungsi untuk mengunakan model *YOLOv5* untuk mendeteksi objek dalam frame yang hasil outputnya masuk dalam variable Results. Informasi pendeteksian diambil dari *Results*.pred[0]. Untuk perintah classes_detected untum memasukan variable yang terbaca dalam kelasnya.

```
lada_putih_detections = detections[classes_detected == 0]
lada_hitam_detections = detections[classes_detected == 1]
count_lada_putih = len(lada_putih_detections)
count_lada_hitam = len(lada_hitam_detections)
```

Gambar 3.40 Kode Hitung Jumlah Objek Pada Tiap Kelas

Pada gambar 3.40 diatas dapat dijelasakan kode yang berfungsi untuk menghitung setiap objek dan memasukannya kedalam kelasnya masing-masing.

```
total_detections = len(detections)
percentage_lada_hitam = (count_lada_hitam / total_detections) * 100 if total_detections > 0 else 0
```

Gambar 3.41 Kode Menghitung Persentase

Pada gambar 3.41 diatas dapat dijelaskan kode yang berfungsi untuk menghitung persentase dari lada kehitam-hitaman.

```
if total_detections == 0:
    category = 'Tidak Ada Lada'
elif percentage_lada_hitam < 1:
    category = 'I'
elif 1 <= percentage_lada_hitam < 7:
    category = 'II'
else:
    category = 'Tidak SNI'
```

Gambar 3.42 Kode Untuk Menentukan Kategori Mutu Lada

Pada gambar 3.42 diatas dapat dijelaskan kode yang berfungsi untuk menentukan kategori lada berdasarkan kadar lada kehitam-hitaman.

```
for detection in detections:
    xmin, ymin, xmax, ymax, confidence, class_label = detection.tolist()
    print(f'Kelas {class_label}: {confidence:.2f}')
    class_label = int(class_label)
    color = (0, 255, 0)

    if class_label == 0:
        color = (0, 255, 0)
    elif class_label == 1:
        color = (0, 0, 255)

    cv2.rectangle(frame, (int(xmin), int(ymin)), (int(xmax), int(ymax)), color, 2)
    label_text = f'Class {class_label}: {confidence:.2f}'
    cv2.putText(frame, label_text, (int(xmin), int(ymin) - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
```

Gambar 3.43 Kode Menampilkan *Bounding Box* Pendeteksian

Pada gambar 3.43 diatas dapat dijelaskan fungsi kode untuk menampilkan *bounding box* pendeteksian objek.

```
cv2.putText(frame, f'Total Lada Terdeteksi = {total_detections} buah',
            (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 0, 0), 2)
cv2.putText(frame, f'Lada Putih = {count_lada_putih} buah',
            (20, 50), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 0, 0), 2)
cv2.putText(frame, f'Lada Kehitam-hitaman = {count_lada_hitam} buah',
            (20, 70), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 0, 0), 2)
cv2.putText(frame, f'Persentase Lada Kehitam-hitaman  = {percentage_lada_hitam:.2f}%',
            (20, 90), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 0, 0), 2)
cv2.putText(frame, f'Kategori Mutu: {category}',
            (20, 110), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 0, 0), 2)
```

Gambar 3. 44 Kode Menampilkan Tulisan Keterangan

Pada gambar 3.44 diatas dapat ditampilkan kode yang berfungsi untuk menampilkan tulisan keterangan dipojok kiri atas.

```
cv2.imshow('TUGAS AKHIR 2024', frame)
```

Gambar 3. 45 Kode Membuka Jendela

Pada gambar 3.45 diatas dapat ditampilkan kode yang berfungsi untuk membuka jendela untuk melihat tampilan dari aktivitas program.

```
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
```

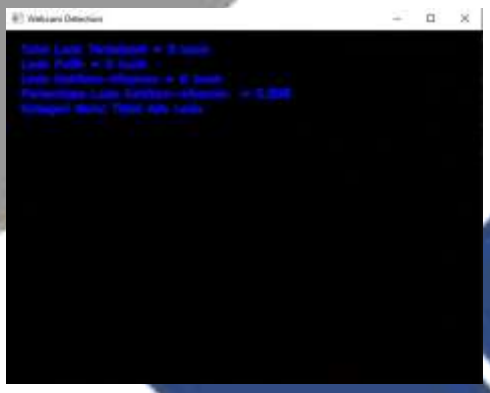
Gambar 3. 46 Kode Untuk Menghentikan Program

Pada gambar 3.46 diatas dapat ditampilkan kode yang berfungsi untuk menutup program beserta jendela yang tadi sudah dibuka.

```
cap.release()
cv2.destroyAllWindows()
```

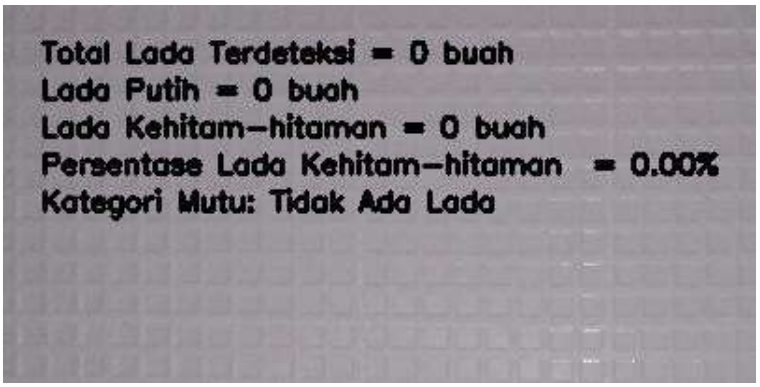
Gambar 3. 47 Kode Untuk Penutup Jendela

Pada gambar 3.47 dapat ditampilkan kode yang berfungsi untuk menutup jendela jika perintah sebelumnya sudah dieksekusi. Dengan menjalankan program diatas maka jendela yang akan terbuka sebagai berikut:



Gambar 3.48 Interface Pendeteksian Jumlah

Pada gambar 3.48 diatas dapat ditampilkan jendela yang menampilkan perhitungan dari ketiga kelas dan bila dijalankan akan menampilkan jendela seperti yang terlihat sebagai berikut:



Gambar 3. 49 Tampilan Saat Menjalankan Program Pendeteksian Jumlah

Pada gambar 3.49 dapat ditampilkan dari tampilan hasil pendeteksian dan perhitungan deteksi lada saat uji coba program. Pada program ini digunakan saat melakukan perhitungan untuk uji ketepatan pendeteksian dan perhitungan sesuai kelasnya dan digunakan dalam pengambilan data akurasi.

3.5 Pengujian & Perbaikan Kontruksi Alat Dan Perangkat Lunak

Pengujian dan perbaikan perangkat lunak adalah proses yang bertujuan untuk menilai sejauh mana alat sudah berkerja dan menentukan perbaikan yang harus dilakukan.

3.5.1 Pengujian Kontruksi Alat

Pengujian kontruksi alat meliputi kerja sistem kontrol untuk mengatur kecepatan berputarnya motor tuas dan motor konveyor. Guna melihat kinerja sistem kontrol dan aspek keselamatan pengunakan alat dan pengunaan ketepatan besaran potensiometer dengan besaran rpm.

3.5.2 Pengujian Perangkat Lunak

Pengujian perangkat lunak dilakukan untuk mengetahui seberapa jauh kinerja pengolah citra untuk menghasilkan tingkat akurasi dan seberapa besar kesalahan yang terjadi pada pengolah citra.

Gambar 3.50 Pengujian Perangkat Lunak

Pada gambar 3.50 dapat ditampilkan saat pengujian perangkat lunak yang dilakukan didalam laboratorium mikrokontroller kampus Politeknik Manufaktur Negeri Bangka Belitung.

3.5.3 Perbaikan

Bila dalam pengujian kontruksi alat maupun perangkat lunak masih memiliki kendala, kesalahan, ataupun kekurangan maka akan dilakukan perbaikan sampai dengan perbaikan sistem dapat diterima.

3.6 Pengambilan Data

Pengambilan data dilakukan setelah proses pengujian selesai dan tidak ada lagi perbaikan yang dilakukan selanjutnya adalah langkah pengambilan data. Pengambilan data berupa kecepatan RPM motor konveyor, RPM motor dc pengayak, tingkat akurasi pengolah citra dibeberapa kondisi, dan nilai bias.

3.7 Penyusunan Laporan Akhir

Penyusunan laporan akhir adalah tahap terakhir dalam proses penelitian aktifitas dan proses dalam metode pelaksanaan akan dijelaskan dalam laporan akhir. Tahap ini bertujuan untuk merangkum semua pembahasan terkait dengan penelitian dan hasil dari penelitian ini.

BAB IV
PEMBAHASAN

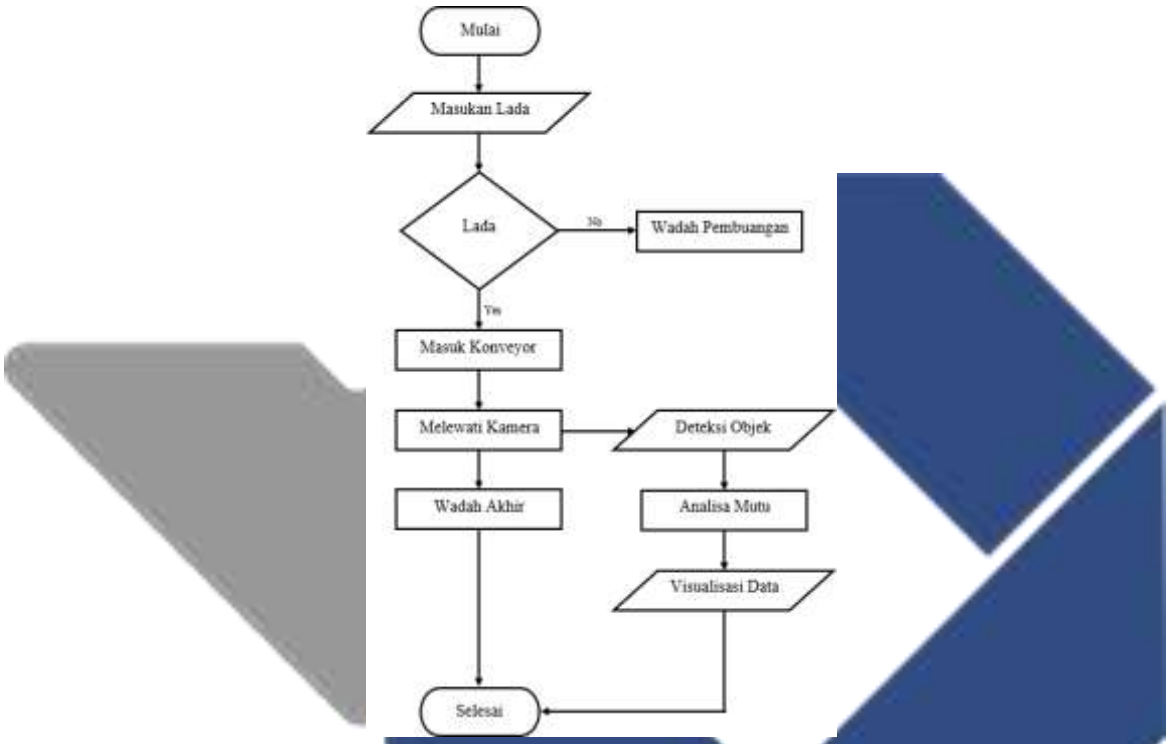
Pada bab ini penulis akan menguraikan hasil-hasil dari proses penelitian yang sudah selesai berdasarkan metode yang telah disebutkan pada bab sebelumnya. Dat-data yang diambil akan menjadi buah kesimpulan yang berikutnya akan dijelaskan pada BAB berikutnya. Berikut adalah pembahasan-pembahasan yang diuraikan seperti dibawah ini:

4.1 Deskripsi Alat

Penelitian yang membuat alat penyortir lada dan pengimplementasian pengolah citra guna mengidentifikasi kualitas mutu lada. Penggunakan pengolah citra dengan algoritma *YOLO* diversi ke-5 mengunakan sebuah kamera. Sebelum lada diidentifikasi, lada akan disaring mengunakan pengayak yang terdiri dari 2 lapis. Lapisan pertama berfungsi sebagai penyortir lada dengan benda asing ukuran besar, lalu lapisan kedua berfungsi untuk menyaring lada dari pasir yang masih tersisa akibat proses penjemuran diruangan terbuka. Cara kerja yang dilakukan oleh pengayak berupa pemanfaatan gerak linier bolak-balik yang terjadi akibat pergerakan motor dc. Dalam motor dc dipasang pulley khusus sehingga pulley tersebut merubah gerak rotasi menjadi gerak linier. Setelah lada bergulir dari lapisan satu kelapisan kedua lalu lada bergulir ke corong yang terhubung pada konveyor. Dikonveyor ini dipasang sebuah kamera yang berfungsi untuk menangkap citra lada yang nantinya melewati kamera. Hasil tangkapan ini akan ditransfer ke komputer, lalu komputer akan mengolah data masukan tersebut menjadi hasil pengidentifikasian. Untuk lada yang sudah melewati kamera akan dibawa kewadah yang sudah disiapkan untuk dikemas.

Diwaktu yang sama hasil penangkapan citra yang dilakukan oleh kamera diproses kedalam model pengolah citra tersebut selanjutnya akan ditentukan dan dihitung. Proses ini akan menghitung lada sesuai hasil pengidentifikasian yang

mana lada akan masuk dalam kualitasnya Putih atau Kehitam-hitaman. lalu jumlahnya akan dihitung tiap kelasnya. Dari indicator ini pula lada akan diidentifiasi kadar mutunya dan ditampilkan hasilnya berupa penentuan Mutu 1, Mutu 2 atau Tidak Masuk SNI.



Gambar 4.1 *Flowchart* alat

4.2 Pengambilan Data

Proses pengambilan data dilakukan apabila kontruksi alat dan perangkat lunak berkerja secara sesuai.

4.2.1 Data Kecepatan Motor

Pengujian kecepatan motor dilakukan mengunakan alat pengukur rpm dan dibandingkan dengan serial monitor pada *software Arduino IDE*.

Tabel 4.1 Tabel Kecepatan Motor Pengayak

No.	Nilai Potensiometer	Selection	RPM

1.	0%	Off	0
2.	1%	1	0
3.	20%	2	0
4.	30%	3	330
5.	40%	4	540
6.	50%	5	600
7.	60%	6	720
8.	70%	7	900
9.	80%	8	1020
10.	90%	9	1200
11.	100%	10	1500

Pada tabel 4.2 diatas dapat ditampilkan tabel dari nilai potensio dan RPM motor pada pengayak.

Tabel 4.2 Tabel Kecepatan Motor Konveyor

No.	**Nilai Potensiometer**	**Selection**	**RPM**
1.	0%	Off	0
2.	1%	1	0
3.	20%	2	0
4.	30%	3	330
5.	40%	4	540
6.	50%	5	600
7.	60%	6	720
8.	70%	7	900
9.	80%	8	1020
10.	90%	9	1200
11.	100%	10	1500

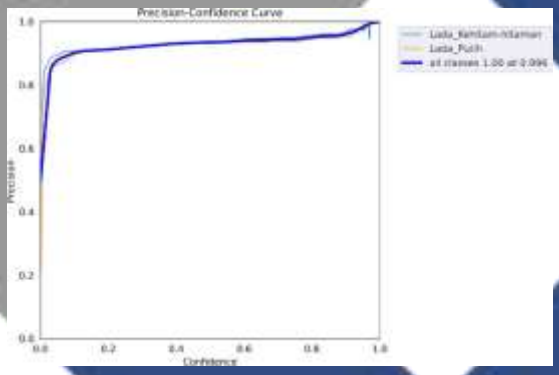
Pada tabel 4.3 diatas dapat ditampilkan tabel dari nilai potensio dan RPM motor pada konveyor.

4.2.2 Data Pasca Pelatihan *Dataset*

Hasil pelatihan *dataset* akan melaporkan beberapa file berupa laporan-laporan hasil *training*. Tetapi file-file tersebut harus diunduh terlebih dahulu dan disimpan dalam folder tersendiri. Berikut beberapa tampilan grafik dari hasil laporan *training*:

4.2.2.1 Grafik *Precision-Confidence Curve*

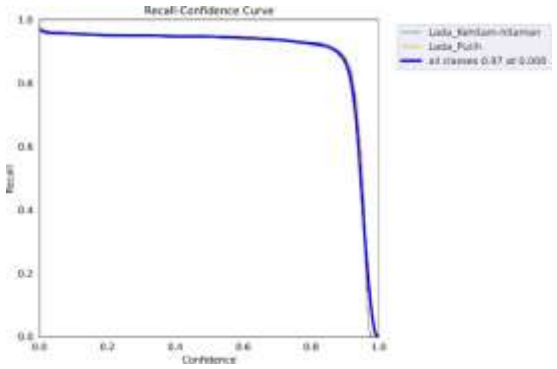
Grafik *Precision* dengan *Confidence*. Grafik ini menjalaskan bagaimana keakuratan terhadap tingkat keyakinan. Grafik menunjukan keakurasian dari tingkat keyakinan terus naik yang artinya tingkat presisi dari tingkat keyakinan bernilai 1.



Gambar 4.2 Hasil Grafik *Precision-Confidence Curve*

4.2.2.2 Grafik *Recall-Confidence Curve*

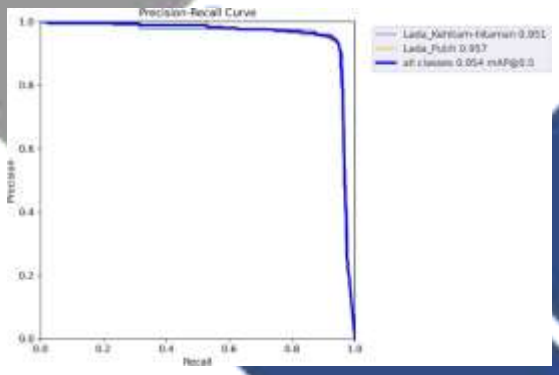
Grafik yang menunjukan tingkat pengambilan objek positif dengan tingkat keyakinan. *Recall* yang artinya rasio objek yang benar terdeteksi dibaningkan dengan total objek yang seharusnya terdeteksi. Dan *Confidence* artinya tingkat keyakinan terhadap pendeteksian yang dibuatnya. Dari hasil *training* kita dapatkan nilai *recall* sebesar 0,97.



Gambar 4.3 Grafik *Recall-Confidence Curve*

4.2.2.3 Grafik *Precision-Recall Curve*

Grafik yang menunjukan keakurasian dari tingkat pengambilan objek positif. *Precision* yang artinya keakurasian dalam mendeteksi objek. Sedangkan *Recall* artinya rasio objek yang benar terdeteksi dibaningkan dengan total objek yang seharusnya terdeteksi. Dari hasil *training* kita dapatkan nilai presisi terhadap *recall* sebesar 0,95.

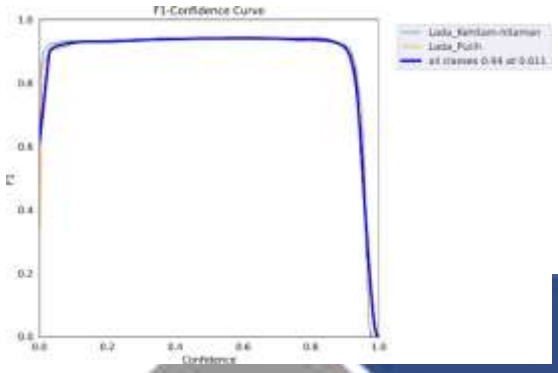


Gambar 4.4 Grafik *Precision-Recall Curve*

4.2.2.4 Grafik *F1-Confidence Curve*

Grafik yang menunjukan *F1* score dengan tingkat keyakinan. *F1* socre yang artinya metrics yang mengukur keseimbangan antara *Precision* dan *Recall*. Sedangkan *Confidence* artinya tingkat keyakinan terhadap pendeteksian yang dibuatnya. Degan kata lain grafik menunjukan hasil dari sejauh mana model mendapatkan keseimbangan optimal antara ketepatan dan pengambilan objek

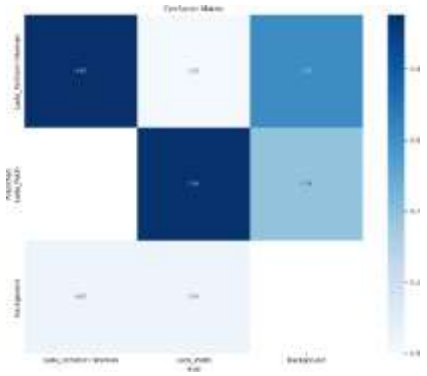
positif dengan mempertimbangkan tingkat keyakinan dalam pendeteksian tersebut. Dari hasil *training* kita dapatkan nilai *F1 Score* sebesar 0,94.



Gambar 4.5 Grafik *F1-Confidence Curve*

4.2.2.5 Grafik *Confusion Matrix*

Pada gambar 4.6 diatas dapat ditampilkan *Confusion Matrix* memberikan informasi hasil klasifikasi yang dilakukan oleh model dengan hasil klasifikasi sebenarnya. Garifk tersebut adalah bentuk representasi visual dari performa model deteksi dengan *dataset* pengujian. Confusion matrik akan membagi hasil prediksi menjadi 4 kategori *yaitu True Positives (TP), False Positives (FP), True Negatives (TN)* dan *False Negatives (FN)*. Dimana *True Positives (TP)* artinya objek positif yang benar-benar terdeteksi sebagai objek positif. *False Negatives (FP)* yang artinya objek negatif yang ikut terdeteksi sebagai objek positif. *True Negatives (TN)* yang artinya objek negatif yang benar-benar terdeteksi sebagai objek negatif. *False Negatives (FN)* yang artinya objek positif yang dideteksi sebagai objek negatif.



Gambar 4.6 *Confusion Matrix*

Pada 4.6 gambar diatas dapat menentukan persentase akurasi dapat dilakukan perhitungan melalui data dari *confusion matrix* dengan persaman sebagai berikut:

$$\text{Akurasi} = \frac{TP+TN}{TP+FP+FN+TN}$$

Dengan mensubtitusikan hasil dari *confusion matrix* kedalam persamaan menjadi,

$$\text{Akurasi} = \frac{(0{,}95+0{,}94)+0{,}02}{(0{,}95+0{,}94)+(0{,}61+0{,}39)+(0{,}05+0{,}04)+0{,}02}$$

$$= \frac{1{,}89+0{,}02}{1{,}89+1+0{,}09+0{,}02}$$

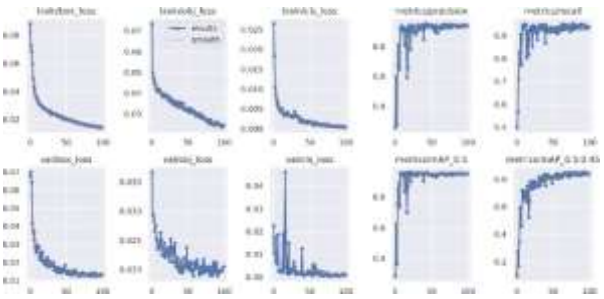
$$= \frac{1{,}91}{3}$$

$$= 0{,}63$$

Kita dapatkan nilai akurasi dari confusion matrix sebesar 63%.

4.2.2.6 Grafik *Result Training Report*

Laporan hasil *training* yang dapat dijelaskan perubahan pada proses *training* berlangsung. Tabel berikut merupakan grafik dari aktifitas *training* yang dilakukan sebanyak 150 epoch.



Gambar 4.7 Grafik *Result Training Report*

4.3 Hasil

Hasil didapatkan dengan melakukan pengujian. Dalampengujian tersebut maka akan disajikan data-data terkait yang dapat menjelaskan informasi sehingga dapat disimpulkan.

4.3.1 Hasil Uji Waktu Pengayakan

Hasil uji waktu pengayakan adalah uji terhadap pengayak dalam memproses lada dalam pengayakan. Dengan bobot lada tertentu dan kecepatan RPM pada motor akan dihitung berapa lama pengayak berkerja. Pengujian Pengayak dilakukan dengan bobot lada sebanyak 1kg.

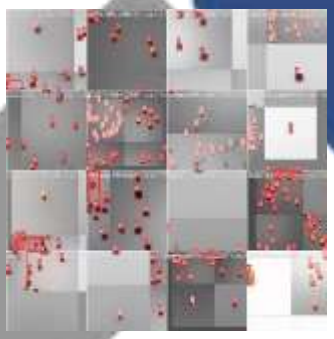
Tabel 4. 3 Uji Waktu Pengayakan

No.	Selection	Waktu (second)	Bobot Lada
1.	Off	0	1000g
2.	1	0	1000g
3.	2	0	1000g
4.	3	0	1000g
5.	4	95	1000g
6.	5	85	1000g
7.	6	85	1000g
8.	7	85	1000g
9.	8	84	1000g

10.	9	84	1000g
11.	10	60	1000g

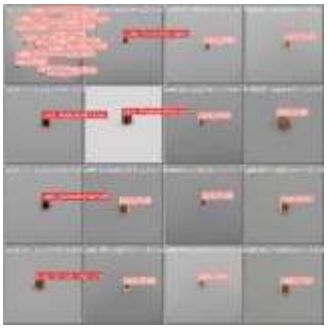
4.3.2 Hasil Pasca *Training*

Hasil pasca *training* adalah laporan terhadap hasil *training* yang dapat dilihat secara langsung terhadap pendeteksian. Hasil ini dijadikan indikator utama apakah pengolah citra sudah sesuai dengan yang diinginkan atau belum. Dan dapat digambarkan bagaimana pengolah citra berkerja nantinya.



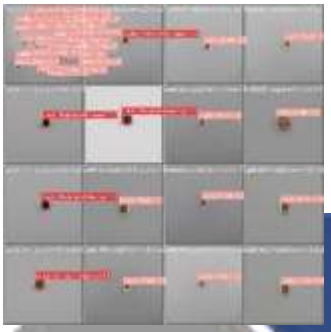
Gambar 4.8 *Train Batch*

Pada gambar 4.8 diatas dapat dijelaskan bagaimana algoritma belajar dan menemukan objek yang keberadaanya diketahui dari hasil labeling dan anotasi. Dimana labeling akan melatih algoritma bahwa objek tersebut masuk dalam kelas mana dan anotasi adalah melatih algoritma mengetahui posisi dari objek tersebut. Dan hasil dari *training* yang dilakukan oleh algoritma dapat diamati seperti pada gambar diatas.



Gambar 4.9 *Valid Labels*

Pada gambar 4.9 diatas dapat dijelaskan dari *valid labels*. Pada kumpulan gambar ini sudah ditemukan oleh algoritma lokasi/koordinat objek yang sebelumnya sudah lebeli dan dianotasikan. Lalu algoritma berhasil menjelaskan label dan dimana objek yang dianotasikan. Dan ditampilkan seperti gambat diatas.



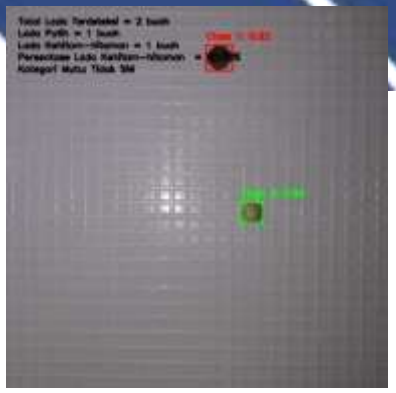
Gambar 4.10 *Valid Prediction*

Pada gambar 4.10 diatas dapat dijelaskan dari kumpulan *valid Prediction*. Pada kumpulan ini sudah diprediksi dan ditampilkan hasil prediksinya seperti yang dapat diamati di gambar 4.9 dan 4.10 diatas.

4.3.3 Hasil Data Pengujian

Pengujin perhitungan dilakukan dengan membandingkan metode perhitungan manual dengan perhitungan pengolah citra dengan mengunakan model YOLOv5.

Tabel 4.4 Data Uji dengan Tinggi Kamera 10cm

No.	Jumlah Aktual	Deteksi YOLOv5
1.	Lada Putih: 1 Lada Kehitam-hitaman: 1	 Lada Putih: 1 Lada Kehitam-hitaman: 1

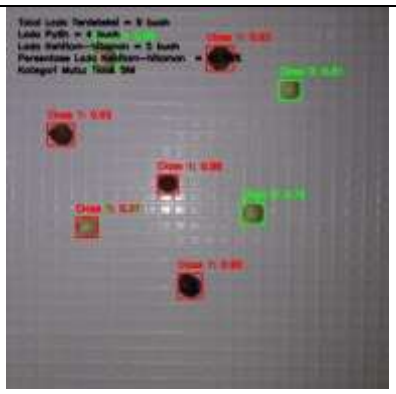
Pada pendeteksian ini, pendeteksian lada berkerja dengan baik dengan mendeteksi 1 lada putih dan 1 lada kehitam-hitaman.

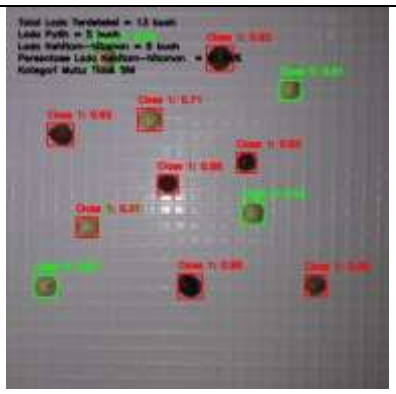
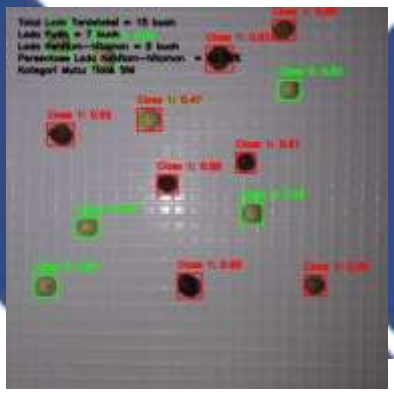
2. Lada Putih: 2
   Lada Kehitam-hitaman: 2

Lada Putih: 2
Lada Kehitam-hitaman: 2
Pada pendeteksian ini, pendeteksian lada berkerja dengan baik dengan mendeteksi 2 lada putih dan 2 lada kehitam-hitaman.

3. Lada Putih: 3
   Lada Kehitam-hitaman: 3

Lada Putih: 3
Lada Kehitam-hitaman: 3
Pada pendeteksian ini, pendeteksian lada berkerja dengan baik dengan mendeteksi 3 lada putih dan 3 lada kehitam-hitaman.

4.	Lada Putih: 4 Lada Kehitam-hitaman: 4	 Lada Putih: 4 Lada Kehitam-hitaman: 5 Pada pendeteksian ini, untuk lada putih sudah benar dengan mendeteksi 4 buah tetapi untuk lada kehitam-hitaman mengalami lebih dalam mendeteksi sebanyak 1 buah, hal ini dikarenakan kurangnya kualitas dari kamera atau atau kurangnya *dataset* yang di*training*.
5.	Lada Putih: 5 Lada Kehitam-hitaman: 5	 Lada Putih: 5 Lada Kehitam-hitaman: 5 Pada pendeteksian ini, pendeteksian lada berkerja dengan baik dengan mendeteksi 5 lada putih dan 5 lada kehitam-hitaman.

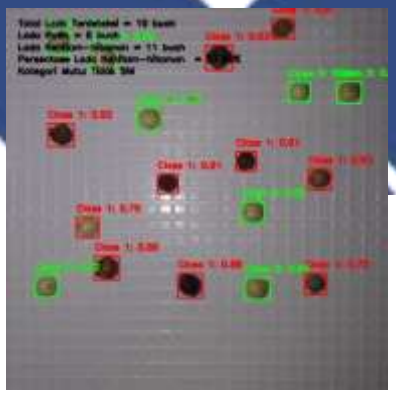
6.	Lada Putih: 6 Lada Kehitam-hitaman: 6	 Lada Putih: 5 Lada Kehitam-hitaman: 8 Pada pendeteksian ini, untuk lada putih mengalami kekurangan deteksi dengan hanya mendeteksi 5 buah tetapi untuk lada kehitam-hitaman mengalami lebih dalam mendeteksi sebanyak 2 buah, hal ini dikarenakan kurangnya kualitas dari kamera atau atau kurangnya *dataset* yang di*training*.
7.	Lada Putih: 7 Lada Kehitam-hitaman: 7	 Lada Putih: 7 Lada Kehitam-hitaman:8 Pada pendeteksian ini, untuk lada putih sudah benar dengan mendeteksi 7 buah tetapi untuk lada kehitam-hitaman mengalami lebih dalam mendeteksi sebanyak 1 buah, hal ini

dikarenakan kurangnya kualitas dari kamera atau atau kurangnya *dataset* yang di*training*.

8. Lada Putih: 8
   Lada Kehitam-hitaman: 10

Lada Putih: 6
Lada Kehitam-hitaman: 10
Pada pendeteksian ini, untuk lada putih mengalami kekurangan deteksi dengan hanya mendeteksi 6 buah tetapi untuk lada kehitam-hitaman sudah benar mendeteksi sebanyak 10 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji, dan kurangnya *dataset* yang di*training*.

9. Lada Putih: 9
   Lada Kehitam-hitaman: 9

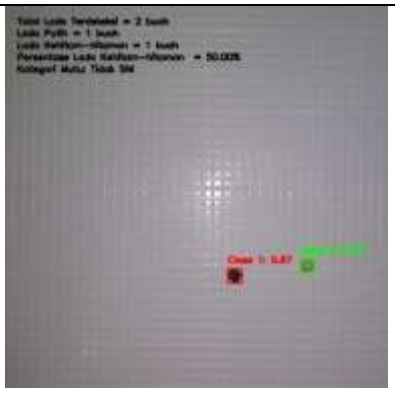
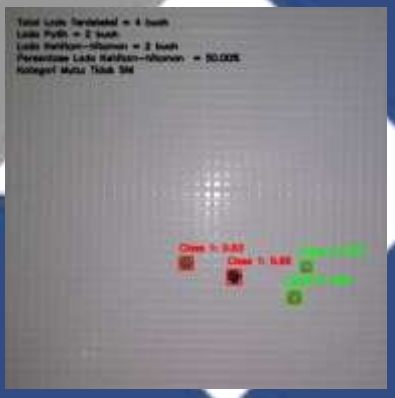


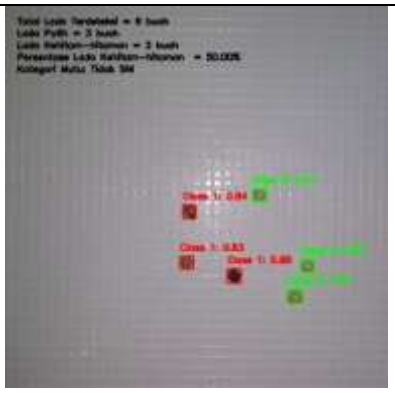
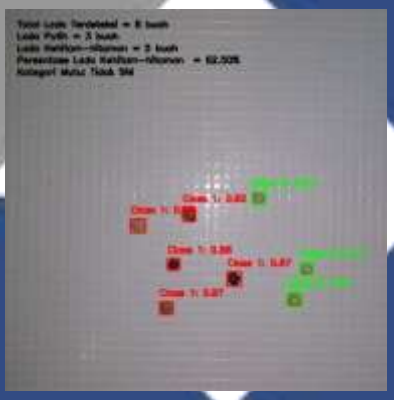
Lada Putih: 8
Lada Kehitam-hitaman: 11
Pada pendeteksian ini, untuk lada putih mengalami kekurangan deteksi dengan hanya

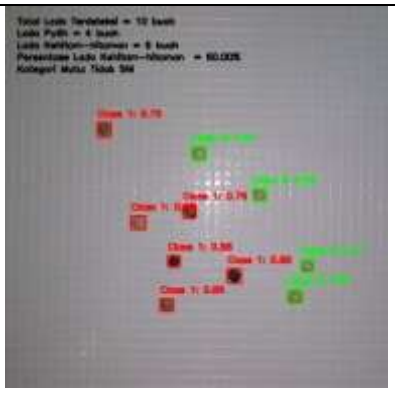
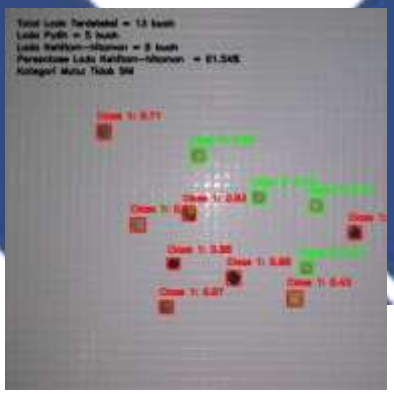
		mendeteksi 8 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi sebanyak 2 buah. Hal ini terjadi karena 53actor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji, dan kurangnya *dataset* yang di*training*.
10.	Lada Putih: 10 Lada Kehitam-hitaman: 10	 Lada Putih: 8 Lada Kehitam-hitaman: 14 Pada pendeteksian ini, untuk lada putih mengalami kekurangan deteksi dengan hanya mendeteksi 8 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi sebanyak 4 buah. Hal ini terjadi karena 53actor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji, dan kurangnya *dataset* yang di*training*.

Tabel 4.5 Data Uji dengan Tinggi Kamera 15cm

No	Jumlah Aktual	Deteksi YOLOv5

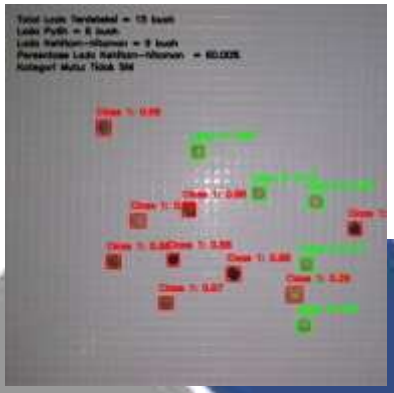
1.	Lada Putih: 1 Lada Kehitam-hitaman: 1	 Lada Putih: 1 Lada Kehitam-hitaman: 1 Pada pendeteksian ini, pendeteksian lada berkerja dengan baik dengan mendeteksi 1 lada putih dan 1 lada kehitam-hitaman.
2.	Lada Putih: 2 Lada Kehitam-hitaman: 2	 Lada Putih: 2 Lada Kehitam-hitaman: 2 Pada pendeteksian ini, pendeteksian lada berkerja dengan baik dengan mendeteksi 2 lada putih dan 2 lada kehitam-hitaman.

3.	Lada Putih: 3 Lada Kehitam-hitaman: 3	 Lada Putih: 3 Lada Kehitam-hitaman: 3 Pada pendeteksian ini, pendeteksian lada berkerja dengan baik dengan mendeteksi 3 lada putih dan 3 lada kehitam-hitaman.
4.	Lada Putih: 4 Lada Kehitam-hitaman: 4	 Lada Putih: 3 Lada Kehitam-hitaman: 5 Pada pendeteksian ini, untuk lada putih mengalami kekurangan dalam mendeteksi dengan hanya mendeteksi 3 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi sebanyak 1 buah. Hal ini terjadi karena 55actor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji atau kurangnya *dataset* yang di*training*.

5.	Lada Putih: 5 Lada Kehitam-hitaman: 5	 Lada Putih: 4 Lada Kehitam-hitaman: 6 Pada pendeteksian ini, untuk lada putih mengalami kekurangan dalam mendeteksi dengan hanya mendeteksi 4 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi sebanyak 1 buah. Hal ini terjadi karena 56actor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji atau kurangnya *dataset* yang di*training*.
6.	Lada Putih: 6 Lada Kehitam-hitaman: 6	 Lada Putih: 5 Lada Kehitam-hitaman: 8 Pada pendeteksian ini, untuk lada putih mengalami kekurangan dalam mendeteksi dengan hanya mendeteksi 5 buah dan lada kehitam-hitaman mengalami kelebihan

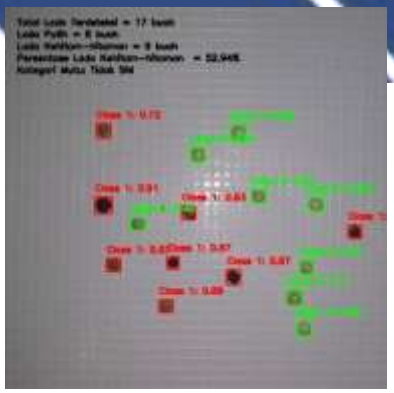
mendeteksi sebanyak 2 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji atau kurangnya *dataset* yang di*training*.

7. Lada Putih: 7
   Lada Kehitam-hitaman: 7



Lada Putih: 6
Lada Kehitam-hitaman: 9
Pada pendeteksian ini, untuk lada putih mengalami kekurangan dalam mendeteksi dengan hanya mendeteksi 6 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi sebanyak 2 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji atau kurangnya *dataset* yang di*training*.

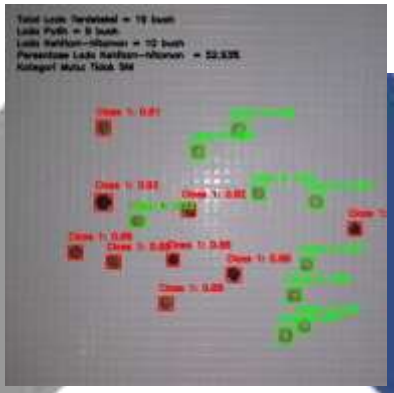
8. Lada Putih: 8
   Lada Kehitam-hitaman: 8



Lada Putih: 8
Lada Kehitam-hitaman: 9

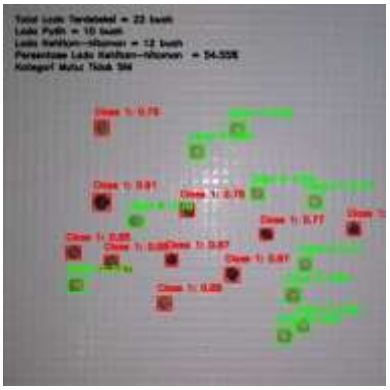
Pada pendeteksian ini, untuk lada putih sudah benar dalam mendeteksi dengan mendeteksi 8 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi sebanyak 1 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji atau kurangnya *dataset* yang di*training*.

9. Lada Putih: 9
   Lada Kehitam-hitaman: 9



Lada Putih: 9
Lada Kehitam-hitaman: 10
Pada pendeteksian ini, untuk lada putih sudah benar dalam mendeteksi dengan mendeteksi 9 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi sebanyak 1 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji atau kurangnya *dataset* yang di*training*.

10. Lada Putih: 10
    Lada Kehitam-hitaman: 10

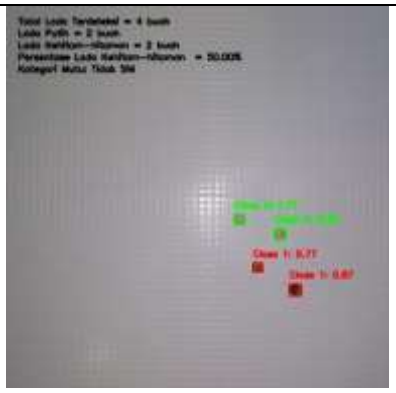


		Lada Putih: 10 Lada Kehitam-hitaman: 12 Pada pendeteksian ini, untuk lada putih sudah benar dalam mendeteksi dengan mendeteksi 10 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi sebanyak 2 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji atau kurangnya *dataset* yang di*training*.

Tabel 4.6 Data Uji dengan Tinggi Kamera 20cm

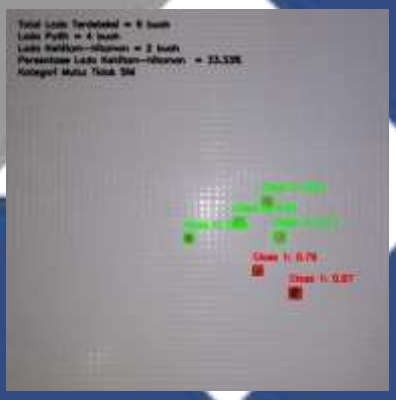
No.	**Jumlah Aktual**	**Deteksi YOLOv5**
1.	Lada Putih: 1 Lada Kehitam-hitaman: 1	 Lada Putih: 1 Lada Kehitam-hitaman: 1 Pada pendeteksian ini, pendeteksian lada berkerja dengan baik dengan mendeteksi 1 lada putih dan 1 lada kehitam-hitaman.

2. Lada Putih: 2
   Lada Kehitam-hitaman: 2



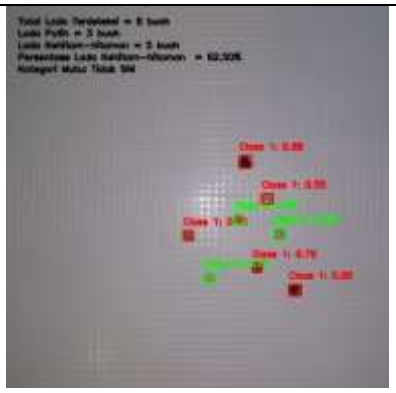
Lada Putih: 2
Lada Kehitam-hitaman: 2
Pada pendeteksian ini, pendeteksian lada berkerja dengan baik dengan mendeteksi 2 lada putih dan 2 lada kehitam-hitaman.

3. Lada Putih: 3
   Lada Kehitam-hitaman: 3



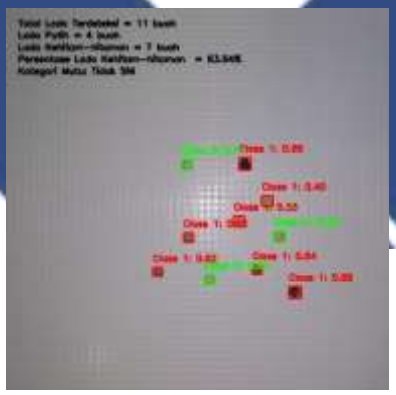
Lada Putih: 4
Lada Kehitam-hitaman: 2
Pada pendeteksian ini, untuk lada putih mengalami kelebihan dalam mendeteksi dengan mendeteksi 4 buah dan lada kehitam-hitaman mengalami kekurangan mendeteksi dengan mendeteksi sebanyak 2 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji dan kurangnya *dataset* yang di*training*.

4. Lada Putih: 4
   Lada Kehitam-hitaman: 4



Lada Putih: 3
Lada Kehitam-hitaman: 5
Pada pendeteksian ini, untuk lada putih mengalami kekurangan dalam mendeteksi dengan hanya mendeteksi 3 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi dengan mendeteksi sebanyak 5 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji dan kurangnya *dataset* yang di*training*.

5. Lada Putih: 5
   Lada Kehitam-hitaman: 5



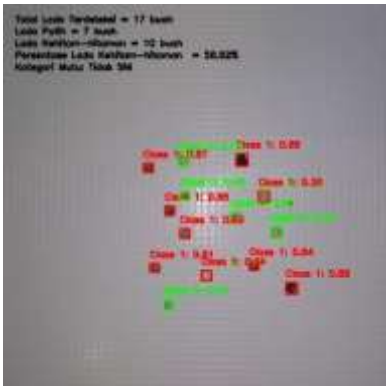
Lada Putih: 4
Lada Kehitam-hitaman: 7
Pada pendeteksian ini, untuk lada putih mengalami kekurangan dalam mendeteksi dengan hanya mendeteksi 4 buah dan lada

kehitam-hitaman mengalami kelebihan mendeteksi dengan mendeteksi sebanyak 7 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji, jarak lada yang berdekatan dan kurangnya *dataset* yang di*training*.

6. Lada Putih: 6
   Lada Kehitam-hitaman: 6

Lada Putih: 6
Lada Kehitam-hitaman: 9
Pada pendeteksian ini, untuk lada putih benar dalam mendeteksi dengan mendeteksi 6 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi dengan mendeteksi sebanyak 9 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji dan kurangnya *dataset* yang di*training*.

7. Lada Putih: 7
   Lada Kehitam-hitaman: 7



Lada Putih: 7

Lada Kehitam-hitaman: 10

Pada pendeteksian ini, untuk lada putih benar dalam mendeteksi dengan mendeteksi 7 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi dengan mendeteksi sebanyak 10 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji dan kurangnya *dataset* yang di*training*.

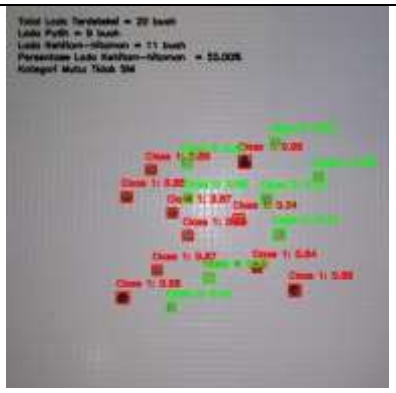
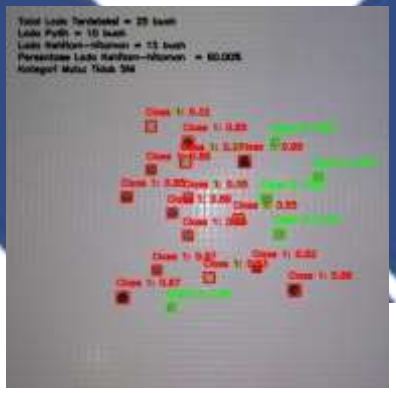
8. Lada Putih: 8

   Lada Kehitam-hitaman: 8

Lada Putih: 8

Lada Kehitam-hitaman: 10

Pada pendeteksian ini, untuk lada putih benar dalam mendeteksi dengan mendeteksi 8 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi dengan mendeteksi sebanyak 10 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji dan kurangnya *dataset* yang di*training*.

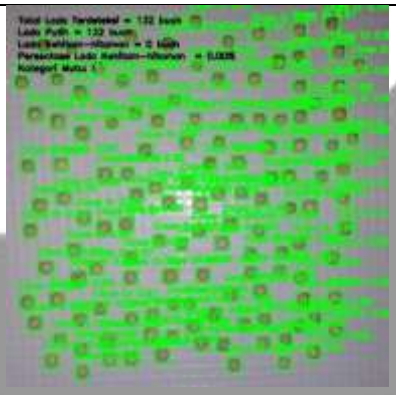
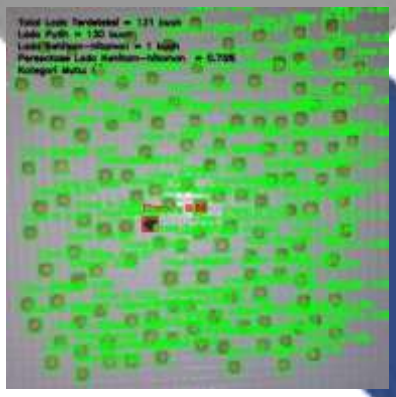
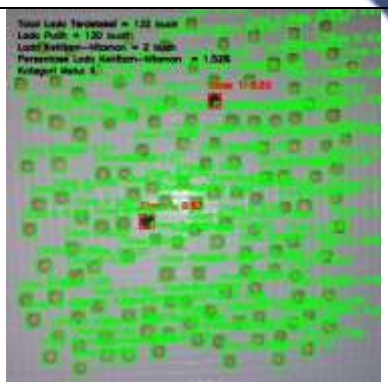
9.	Lada Putih: 9 Lada Kehitam-hitaman: 9	 Lada Putih: 9 Lada Kehitam-hitaman: 11 Pada pendeteksian ini, untuk lada putih benar dalam mendeteksi dengan mendeteksi 9 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi dengan mendeteksi sebanyak 11 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji dan kurangnya *dataset* yang di*training*.
10.	Lada Putih: 10 Lada Kehitam-hitaman: 10	 Lada Putih: 10 Lada Kehitam-hitaman: 15 Pada pendeteksian ini, untuk lada putih benar dalam mendeteksi dengan mendeteksi 10 buah dan lada kehitam-hitaman mengalami kelebihan mendeteksi dengan mendeteksi

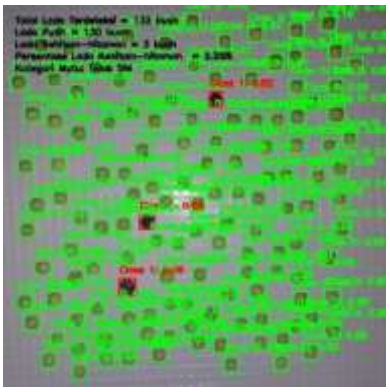
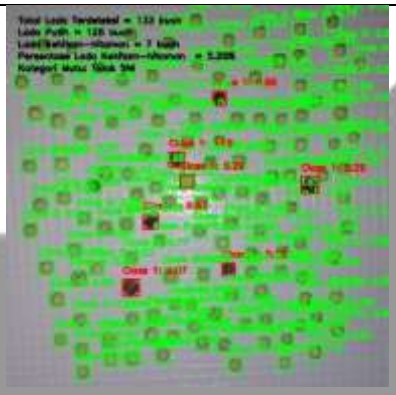
	sebanyak 15 buah. Hal ini terjadi karena faktor kurangnya kualitas dari kamera, kurangnya pencahayaan lokasi uji dan kurangnya *dataset* yang di*training*.

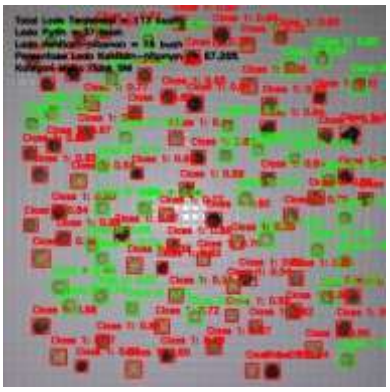
Pada tabel 4.5, tabel 4.6 dan tabel 4.7 diatas dapat ditampilkan hasil pengujian dimana didapatkan hasil yang ditampilkan pada layar monitor. Dimana objek yang terdeteksi pada bounding box berwarna hijau adalah kelas Lada Putih dan yang berwarna merah adalah kelas Lada Kehitam-hitaman. Tulisan yang berada pada pojok kiri atas berwarna hitam akan menampilkan banyaknya semua lada yang terdeteksi, banyaknya lada putih yang terdeteksi, banyakya lada kehitam-hitaman yang terdeteksi, persentasi banyaknya kadar lada kehitam-hitaman dan kategori mutu dari lada tersebut. Perbedaan terdapat pada setiap tabel dimana tabel 4.5 pengujian dilakukan diketinggian kamera sebesar 10cm, pada tabel 4.6 pengujian dilakukan pada ketinggian kamera sebesar 15cm dan pada tabel 4.7 pengujian dilakukan pada ketinggian kamera sebesar 20cm.

Tabel 4.7 Uji Penentuan Mutu Dan Persentase

No.	Hasil Uji	Keterangan
1.		Pada uji kali ini, media dikosongkan dari objek atau lada guna melihat kinerja 0 pada algoritma.

2.		Pada uji kali ini, objek lada diberikan 1 lada putih guna dapat ditampilkan bahwa Lada Kehitam-hitaan terdeteksi 0, Persentase dari Lada Kehitam-hitaman 0,00% dan Kategori Mutu masuk pada Mutu 1.
3.		Pada uji kali ini, objek lada diberikan 132 lada putih guna dapat ditampilkan bahwa Lada Kehitam-hitaan terdeteksi 0, Persentase dari Lada Kehitam-hitaman 0,00% dan Kategori Mutu masuk pada Mutu 1.
4.		Pada uji kali ini, objek lada diberikan 130 lada putih dan 1 Lada Kehitam-hitaman guna dapat ditampilkan bahwa Lada Kehitam-hitaan terdeteksi 1 dan Persentase dari Lada Kehitam-hitaman 0,76% dan Kategori Mutu masuk pada Mutu 1.
5.		Pada uji kali ini, objek lada diberikan 130 lada putih dan 2 Lada Kehitam-hitaman guna dapat ditampilkan bahwa Lada Kehitam-hitaan terdeteksi 2 dan Persentase dari Lada Kehitam-hitaman 1,52% dan Kategori Mutu masuk pada Mutu 2.

6.		Pada uji kali ini, objek lada diberikan 130 lada putih dan 3 Lada Kehitam-hitaman guna dapat ditampilkan bahwa Lada Kehitam-hitaan terdeteksi 3 dan Persentase dari Lada Kehitam-hitaman 2,26% dan Kategori Mutu masuk pada Tidak SNI.
7.		Pada uji kali ini, objek lada diberikan 127 lada putih dan 7 Lada Kehitam-hitaman guna dapat ditampilkan bahwa Lada Kehitam-hitaan terdeteksi 7 dan Persentase dari Lada Kehitam-hitaman 5,26% dan Kategori Mutu masuk pada Mutu Tidak SNI.
8.		Pada uji kali ini, objek lada diberikan 47 lada Kehitam-hitaman guna dapat ditampilkan bahwa Lada Kehitam-hitaan terdeteksi 47 dan Persentase dari Lada Kehitam-hitaman 100,00% dan Kategori Mutu masuk pada Mutu Tidak SNI.
9.		Pada uji kali ini, objek lada diberikan 1 Lada Putih dan 49 lada Kehitam-hitaman guna dapat ditampilkan bahwa Lada Kehitam-hitaan terdeteksi 49 dan Persentase dari Lada Kehitam-hitaman 98,00% dan Kategori Mutu masuk pada Mutu Tidak SNI.

10.		Pada uji kali ini, objek lada diberikan 37 Lada Putih dan 76 lada Kehitam-hitaman guna dapat ditampilkan bahwa Lada Kehitam-hitaan terdeteksi 72 dan Persentase dari Lada Kehitam-hitaman 67,26% dan Kategori Mutu masuk pada Mutu Tidak SNI.

Pada tabel 4.8 diatas dapat ditampilkan hasil pengujian dimana didapatkan hasil bahwa program computer untuk menentukan persentase lada kehitam-hitaman sudah berhasil. Dan untuk program komputer yang berfungsi untuk menentukan mutu dengan indikator lada kehitam-hitaman juga berfungsi dengan baik. Dapat dilihat pada tabel 4.8 dimana saat kadar lada putih berada dibawah 1% maka kategori mutu menentukan bahwa itu masuk pada mutu 1, lalu pada saat kadar lada kehitam-hitaman berada dibawah 2% maka kategori mutu menentukan bahwa itu masuk pada mutu 2 dan pada saat kadar lada kehitam-hitaman berada pada persentase diatas 2% maka kategori mutu menentukan bahwa itu masuk pada kategori mutu Tidak SNI. Pengujian ini dilakukan pada kondisi tinggi kamera 15cm. ini didasari oleh kebutuhan pengujian yang membutuhkan lada sebanyak 133 butir sehingga lada butuh lahan yang cukup luas untuk membuat butiran lada tidak terlalu berdekatan dan tangkapan kamera menjadi lebih besar.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari poses yang sudah diselesaikan dan pengujian terhadap kontruksi alat serta sistem yang sudah dilakukan, penelitian ini berhasil mengimplementasikan pengolah citra pada prototipe alat penyortir lada. Kemudian kesimpulannya dapat dijabarkan sebagai berikut:

1. Pada pengujian alat penyortir lada yang memiliki dimensi sebesar 8 mm^2 akan lolos dalam penyaringan. Ini berlaku juga tehadap benda asing dimana hanya benda asing yang berukuran lebih besar dari 8 mm^2 yang akan tertahan oleh penyaring. Tetapi proses ini akan mengalami 2 kali penyringan dimana pada penyaringan kedua hanya benda yang berukuran 1mm^2 yang akan terbuang dan lada yang berukuran lebih dari 1 mm^2 akan tertahan pada penyaringan ini dan selanjutnya akan digulirkan kekonveyor. Kesimpulannya untuk alat penyortir ini berkerja sesuai dengan yang direncanakan.
2. Pada pengujian pengolah citra sudah dapat mendeteksi lada dengan 2 parameter yaitu lada putih dan lada kehitam-hitaman dengan jumlah yang bervariasi. Walaupun demikian, performa dari pengolah citra akan berpengaruh terhadap jumlah bobot lada, area tangkap kamera dan pencahayaan lingkungan.
3. Pada pengujian pengolah citra untuk perhitungan persentase lada kehitam-hitaman dan penentuan mutu lada keseluruhan berdasarkan jumlah lada kehitam-hitaman sudah berhasil. Dengan ditampilkannya visualisasi data yang menampilkan persentase kadar lada kehitam-hitaman dan penentuan mutu yang berlandaskan SNI 01-0004-2013, program dapat menentukan kadar lada yang dideteksinya masuk dalam kategori mutu 1, mutu 2 atau tidak SNI.

4. Pada pengujian pengolah citra, jumlah gambar pada *dataset* untuk proses *training* sangat mempengaruhi hasil dari performa pengolah citra. Semakin tepat dalam memilih jumlah gambar untuk *dataset* maka semakin baik performa pengolah citra yang dihasilkan. Pada dasarnya jika jumlah gambar pada *dataset* terlalu sedikit maka tingkat performa juga kurang maksimal dimana akan mempengaruhi tingkat akurasi. Akan tetapi dengan jumlah *dataset* yang terlalu jika parameter yang digunakan sedikit seperti pada penelitian ini, maka itu hanya akan memperlambat proses penelitian terutama pada lebeling anotasi gambar dan proses *training dataset*.
5. Untuk pemilihan jumlah *epoch* pada saat proses *training* juga harus tepat, hal ini dapat diamati pada parameter jumlah dataset atau dilihat pada *report training* pada saat proses berlangsung. jika diparameter *precision, recall* dan *mAP50* mengalami saturasi dimana data yang dihasilkan tidak mengalami perubahan yang signifikan. Bisa juga jika hasil dari parameter tersebut sudah mendekati 1 maka dengan jumlah *epoch* tersebut sudah cukup dalam men*training dataset*. Hal ini karena dengan memilih *epoch* yang terlampau banyak, hal ini akan membuat model bukan mempelajari suatu citra tetapi membuat model menjadi menghafal dataset.
6. Pada penelitian ini didapatkan hasil dari *precision* sebesar 1, *recall* sebesar 0,97, *F1 score* sebesar 0,94 dan akurasi *confusion matrix* sebesar 63%. Dari hasil tersebut untuk performa YOLOv5 dalam mendeteksi dan kebenaran pendeteksian yaitu terciptanya bounding box dan penentuan kelas terhadap objek lada sudah baik sehingga dapat dilihat dari hasil pengujian terhadap beberapa jumlah lada bahwa lada terdeteksi semua, tetapi untuk akurasi masih harus dikembangkan kembali, karena dengan hasil akurasi 63% masih ada beberapa situasi dimana YOLOv5 mengalami kekeliruan dalam menentukan pendeteksiannya, seperti saat objek sangat banyak atau pencahayaan lingkungan tangkapnya kurang.

5.2 Saran

Dari hasil proyek akhir ini penulis memiliki beberapa saran untuk pengembangan yang lebih lanjut sebagai berikut:

1. Memilih dengan tepat jumlah data gambar untuk dataset, membuat data gambar dengan resolusi yang baik, memilih variasi data gambar yang tepat sehingga tidak ada data gambar kembar, dan memanfaatkan framework untuk dataset karena itu juga dapat membantu mempersingkat proses pengumpulan dataset.
2. Memilih epoch yang tepat saat proses training sehingga dapat menghasilkan model yang baik performanya tetapi tidak membuah waktu terlalu lama untuk proses training.
3. Dapat memanfaatkan berbagai jenis framework dimana itu dapat membantu dalam proses pembuatan dengan perangkat yang kurang spesifikasinya.
4. Menggunakan algoritma deteksi objek yang lainnya seperti YOLOv8 yaitu YOLO dengan versi yang terbaru karena diharapkan dengan menggunakan algoritma dengan versi terbaru akan memanfaatkan fitur-fitur terbarunya tau bahkan performa yang terbaru pula.
5. Menggunakan kamera dengan spesifikasi yang sesuai karena ada beberapa fitur yang dapat membantu menghasilkan tangkapan citra yang baik tanpa harus mengeluarkan biaya yang berlebih.

DAFTAR PUSTAKA

[1] K. Kamila, R. Kusmiadi, and S. N. Aini, "PENGARUH PENUNDAAN WAKTU PERENDAMAN DAN PELUKAAN MEKANIS TERHADAP KUALITAS LADA PUTIH MUNTOK," *JURNAL BIOINDUSTRI*, vol. 1, no. 2, pp. 213–228, Jul. 2019, doi: 10.31326/jbio.v1i2.349.

[2] I. Dwisaputra, S. Barokah, M. E. Ramadhani, and O. Ocsirendi, "Determinants of Pepper Quality Based on the Percentage of Foreign Objects Based You Only Look Once (YOLO)," *Jurnal Elektronika dan Telekomunikasi*, vol. 23, no. 1, p. 37, Aug. 2023, doi: 10.55981/jet.525.

[3] A. B. Pulungan and Z. Nafis, "Rancangan Alat Pendeteksi Benda dengan Berdasarkan Warna, Bentuk, dan Ukuran dengan Webcam," *JTEIN: Jurnal Teknik Elektro Indonesia*, vol. 2, no. 1, pp. 49–54, Feb. 2021, doi: 10.24036/jtein.v2i1.111.

[4] Y. Mufidah, R. Noah, L. Lawalatta, and N. Bragas, "PENGARUH TINGKAT AKURASI DALAM IDENTIFIKASI GEJALA DAN TANDA PENYAKIT PADA TANAMAN," *Jurnal Informatika Progres*, vol. 14, no. 1, pp. 11–15, Apr. 2022, doi: 10.56708/progres.v14i1.301.

[5] A. S. Riyadi, I. P. Wardhani, M. S. Wulandari, and S. Widayati, "Perbandingan Metode ResNet, YoloV3, dan TinyYoloV3 pada Deteksi Citra dengan Pemrograman Python," *PETIR*, vol. 15, no. 1, pp. 135–144, Jan. 2022, doi: 10.33322/petir.v15i1.1302.

[6] L. Rahma, H. Syaputra, A. H. Mirza, and S. D. Purnamasari, "Objek Deteksi Makanan Khas Palembang Menggunakan Algoritma YOLO (You Only Look Once)," *Jurnal Nasional Ilmu Komputer*, vol. 2, no. 3, pp. 213–232, Nov. 2021, doi: 10.47747/jurnalnik.v2i3.534.

[7] T. Abuzairi, Nurdina Widanti, Arie Kusumaningrum, and Yeni Rustina, “Implementasi Convolutional Neural Network Untuk Deteksi Nyeri Bayi Melalui Citra Wajah Dengan YOLO,” *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 4, pp. 624–630, Aug. 2021, doi: 10.29207/resti.v5i4.3184.

[8] K. Maulana Azhar, I. Santoso, D. Yosua, and A. A. Soetrisno, “IMPLEMENTASI DEEP LEARNING MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK DAN ALGORITMA YOLO DALAM SISTEM PENDETEKSI UANG KERTAS RUPIAH BAGI PENYANDANG LOW VISION,” 2021. [Online]. Available: https://ejournal3.undip.ac.id/index.php/transient

[9] S. Pokuciński and D. Mrozek, “Object Detection with YOLOv5 in Indoor Equirectangular Panoramas,” *Procedia Comput Sci*, vol. 225, pp. 2420–2428, 2023, doi: 10.1016/j.procs.2023.10.233.

[10] D. I. Mulyana and M. A. Rofik, “Implementasi Deteksi Real Time Klasifikasi Jenis Kendaraan Di Indonesia Menggunakan Metode YOLOV5.”

[11] L. Agustien, T. Rahman, and A. W. Hujairi, “Real-time Deteksi Masker Berbasis Deep Learning menggunakan Algoritma CNN YOLOv3,” 2021. [Online]. Available: https://doi.org/10/25047/jtit.v8i2.246

[12] Q. Aini, N. Lutfiani, H. Kusumah, and M. S. Zahran, “Deteksi dan Pengenalan Objek Dengan Model Machine Learning: Model Yolo,” *CESS (Journal of Computer Engineering, System and Science)*, vol. 6, no. 2, p. 192, Jul. 2021, doi: 10.24114/cess.v6i2.25840.

[13] M. H. Alhayani, “Real-Time Objects Detection, Tracking, and Counting Using Image Processing Techniques,” *Al-Nahrain Journal for Engineering Sciences*, vol. 26, no. 1, pp. 24–30, Feb. 2023, doi: 10.29194/NJES.26010024.

[14] S. Cakic, T. Popovic, S. Krco, D. Nedic, and D. Babic, “Developing Object Detection Models for Camera Applications in Smart Poultry Farms,” in *2022 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*, IEEE, Aug. 2022, pp. 1–5. doi: 10.1109/COINS54846.2022.9854975.

[15] C. Kurniawan Umbu Nggiku, A. Rabi, and S. Subairi, “Deteksi Kantuk Untuk Keamanan Berkendara Berbasis Pengolahan Citra,” *Jurnal JEETech*, vol. 4, no. 1, pp. 48–56, Sep. 2023, doi: 10.32492/jeetech.v4i1.4107.

[16] H. Muchtar and R. Apriadi, “Implementasi Pengenalan Wajah Pada Sistem Penguncian Rumah Dengan Metode Template Matching Menggunakan Open Source Computer Vision Library (Opencv),” *RESISTOR (elektRonika kEndali telekomunikaSI tenaga liSTrik kOmputeR)*, vol. 2, no. 1, p. 39, May 2019, doi: 10.24853/resistor.2.1.39-42.

[17] F. F. Aiyub and M. Munawir, “Kontrol Mouse Menggunakan Webcam Berdasarkan Deteksi Warna,” *JTIM : Jurnal Teknologi Informasi dan Multimedia*, vol. 1, no. 1, pp. 73–77, May 2019, doi: 10.35746/jtim.v1i1.18.

LAMPIRAN

LAMPIRAN 1

DAFTAR RIWAYAT HIDUP

1. **Data Pribadi**

Nama Lengkap	: Nuzul Bragas Sabilillah
Tempat & Tanggal lahir	: Jakarta, 15 Januari 1999
Alamat	: Jl. Raya Malaka No.70
	RT 002 RW 08 Perumnas
	Klender, Malaka Sari, Duren
	Sawit, Jakarta Timur
	Telp: -
	Hp: 085764365662
	E-mail: nuzulbragass@gmail.com
Jenis Kelamin	: Laki-laki
Agama	: Islam

2. **Riwayat Pendidikan**

SDN Malaka Sari 04 Pagi	Lulus	Tahun 2010
SMP N 213 Jakarta	Lulus	Tahun 2013
SMA Muhammadiyah 23 Jakarta	Lulus	Tahun 2016

3. **Riwayat Pendidikan Non-Formal**

-

Sungailiat, 15 Januari 2024

Nuzul Bragas Sabilillah

DAFTAR RIWAYAT HIDUP

1. **Data Pribadi**

   Nama Lengkap : Rafda Ardianto

   Tempat & Tanggal lahir : Kampung Baru, 31 Mei 2001

   Alamat : Dusun Kampung Baru Timur

   Telp: -

   Hp: 082181439245

   E-mail: rafdaa7@gmail.com

   Jenis Kelamin : Laki-laki

   Agama : Islam

2. **Riwayat Pendidikan**

SDN 13 Parit Tiga Jebus	Lulus	Tahun 2014
SMPN 1 Jebus	Lulus	Tahun 2017
SMA 1 Parit Tiga	Lulus	Tahun 2020

3. **Riwayat Pendidikan Non-Formal**

   -

Sungailiat, 15 Januari 2024

Rafda Ardianto

LAMPIRAN 2

PROGRAM KESELURUHAN

```
#include "HCMotor.h"

#define MOTOR_PIN_1 6

#define MOTOR_PIN_2 9

#define POT_PIN_1 A0

#define POT_PIN_2 A1

#define IR_PIN 2

HCMotor HCMotor;

volatile unsigned int counter = 0;

unsigned long previousMillis = 0;

unsigned int rpm = 0;

void setup() {

  HCMotor.Init();

  HCMotor.attach(0, DCMOTOR, MOTOR_PIN_1);

  HCMotor.DutyCycle(0, 100);

  HCMotor.attach(1, DCMOTOR, MOTOR_PIN_2);

  HCMotor.DutyCycle(1, 100);

}

void loop() {

   unsigned long currentMillis = millis();

  if (currentMillis - previousMillis >= 1000) {

    detachInterrupt(digitalPinToInterrupt(IR_PIN));
```

```
    rpm = (counter / 2) * 60;
    counter = 0;
    attachInterrupt(digitalPinToInterrupt(IR_PIN), countPulse, FALLING);
    previousMillis  = currentMillis;
    Serial.print("RPM: ");
    Serial.println(rpm);
  int Speed_1, Speed_2;
  // Membaca nilai dari potensiometer pertama
  Speed_1 = map(analogRead(POT_PIN_1), 0, 1024, 0, 100);
  // Membaca nilai dari potensiometer kedua
  Speed_2 = map(analogRead(POT_PIN_2), 0, 1024, 0, 100);
  // Mengatur kecepatan untuk motor pertama
  HCMotor.OnTime(0, Speed_1);
  // Mengatur kecepatan untuk motor kedua
  HCMotor.OnTime(1, Speed_2);
}
void countPulse() {
  counter++;
}
!git clone https://github.com/ultralytics/yolov5
%cd yolov5
%pip install -qr requirements.txt comet_ml
```

```
import torch

import utils

display = utils.notebook_init()

!pip install roboflow

from roboflow import Roboflow

rf = Roboflow(api_key="fkQsovM96K9jsjEExFXK")

project = rf.workspace("politeknik-manufaktur-negeri-bangka-belitung-dr0ie").project("tugas-akhir-dataset-300")

dataset = project.version(4).download("yolov5")

!unzip '/content/drive/MyDrive/pepper2class.zip'

!python train.py --img 640 --batch 16 --epochs 1 --data /content/drive/MyDrive/data.yaml --weights yolov5l.pt --cache

import cv2

import torch

import numpy as np

model = torch.hub.load('ultralytics/yolov5', 'custom', '/content/drive/MyDrive/epoch100yolov5l.pt')

cap = cv2.VideoCapture(0)

while True:

    ret, frame = cap.read()

    results = model(frame)

    detections = results.pred[0]

    classes_detected = detections[:, -1].cpu().numpy().astype(int)
```

```
    lada_baik_detections = detections[classes_detected == 0]
    lada_buruk_detections = detections[classes_detected == 1]
    count_lada_baik = len(lada_baik_detections)
    count_lada_buruk = len(lada_buruk_detections)
    total_detections = len(detections)
    percentage_lada_buruk = (count_lada_buruk / total_detections) * 100 if
total_detections > 0 else 0
    if total_detections == 0:
        category = 'Tidak Ada Lada'
    elif percentage_lada_buruk < 1:
        category = 'I'
    elif 1 <= percentage_lada_buruk < 2:
        category = 'II'
    else:
        category = 'Tidak SNI'
    for detection in detections:
        xmin, ymin, xmax, ymax, confidence, class_label = detection.tolist()
        class_label = int(class_label)
        color = (0, 255, 0)
        if class_label == 0:
            color = (0, 255, 0)
        elif class_label == 1:
            color = (0, 0, 255)
```

```
        cv2.rectangle(frame, (int(xmin), int(ymin)), (int(xmax), int(ymax)), color, 2)
        # Display class label and confidence
        label_text = f'Class {class_label}: {confidence:.2f}'
        cv2.putText(frame, label_text, (int(xmin), int(ymin) - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
    cv2.putText(frame, f'Total Lada Terdeteksi = {total_detections} buah', (20, 30),
cv2.FONT_HERSHEY_SIMPLEX, 0.5,
                (0, 0, 0), 2)
    cv2.putText(frame, f'Lada Putih = {count_lada_baik} buah', (20, 50),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2)
    cv2.putText(frame, f'Lada Kehitam-hitaman = {count_lada_buruk} buah', (20,
70), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
                (0, 0, 0), 2)
    cv2.putText(frame, f'Persentase Lada Kehitam-hitaman =
{percentage_lada_buruk:.2f}%', (20, 90),
                cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2)
    cv2.putText(frame, f'Kategori Mutu: {category}', (20, 110),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2)
    cv2.imshow('Webcam Detection', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```