TUGAS AKHIR

PERANCANGAN SISTEM OBJECT DETECTION PADA INDUSTRI ALAT BERAT MENGGUNAKAN DEEP LEARNING

Dikky Kurniawan
NIM. 04211020

Amalia Rizqi Utami, S.T., M.T.
Barokatun Hasanah, S.T, M.T.

Program Studi Teknik Elektro
Jurusan Teknik Elektro, Informatika, dan Bisnis
Fakultas Sains dan Teknologi Informasi
Institut Teknologi Kalimantan
Balikpapan, 2025

TUGAS AKHIR

PERANCANGAN SISTEM OBJECT DETECTION PADA INDUSTRI ALAT BERAT MENGGUNAKAN DEEP LEARNING

Dikky Kurniawan
NIM. 04211020

Amalia Rizqi Utami, S.T., M.T.
Barokatun Hasanah, S.T, M.T.

Program Studi Teknik Elektro
Jurusan Teknik Elektro, Informatika, dan Bisnis
Fakultas Sains dan Teknologi Informasi
Institut Teknologi Kalimantan
Balikpapan, 2025

PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul **PERANCANGAN SISTEM OBJECT DETECTION PADA INDUSTRI ALAT BERAT MENGGUNAKAN DEEP LEARNING** adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diizinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri. Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Balikpapan, 5 Juli 2025

Dikky Kurniawan

NIM. 04211020

Halaman ini sengaja dikosongkan

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Institut Teknologi Kalimantan, saya yang bertanda tangan di bawah ini:

Nama : Dikky Kurniawan

NIM : 04211020

Program Studi : Teknik Elektro

Jurusan : Teknik Elektro, Informatika dan Bisnis

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Institut Teknologi Kalimantan Hak Bebas Royalti Non-ekslusif (***Non-exclusive Royalty Free Right***) atas karya ilmiah saya yang berjudul:

PERANCANGAN SISTEM OBJECT DETECTION PADA INDUSTRI ALAT BERAT MENGGUNAKAN DEEP LEARNING

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-eksklusif ini, Institut Teknologi Kalimantan berhak mentimpan, mengalihmediakan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya,

Balikpapan, 5 Juli 2025

Dikky Kurniawan
NIM. 04211020

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas berkat dan anugerah-Nya sehingga penulis dapat menyelesaikan laporan tugas akhir yang berjudul :

”PERANCANGAN SISTEM OBJECT DETECTION PADA INDUSTRI ALAT BERAT MENGGUNAKAN DEEP LEARNING”

Laporan tugas akhir ini merupakan salah satu syarat yang harus ditempuh untuk menyelesaikan Program Sarjana di Program Studi Teknik Elektro Jurusan Teknologi Industri dan Proses, Institut Teknologi Kalimantan (ITK) Balikpapan. Untuk itu penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Allah SWT atas Rahmat dan Anugerah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir.
2. Keluarga Penulis yang selalu mendukung dan mendoakan.
3. Ibu Amalia Rizqi Utami, S.T., M.T. selaku Dosen Pembimbing Utama dan Ibu Barokatun Hasanah, S.T, M.T. selaku Dosen Pembimbing Pendamping.
4. Bapak Kharis Sugiarto, SST., M.T. selaku Koordinator Tugas Akhir Program Studi Teknik Elektro Jurusan Teknologi Industri dan Proses ITK.
5. Bapak Kharis Sugiarto, SST., M.T. selaku Koordinator Program Studi Teknik Elektro Jurusan Teknologi Industri dan Proses ITK.
6. Bapak Adi Mahmud Jaya Marindra, S.T., M.Eng.,Ph.D. selaku dosen wali penulis yang selalu membantu dan mendukung seluruh aktivitas penulis di kampus.
7. Nuri Dwi Safitri yang selalu mendukung dan menemani penulis selama menyelesaikan tugas akhir.
8. Serta semua pihak yang terlibat dalam penyusunan Tugas Akhir ini yang tidak bisa disebutkan satu-persatu.

Kami menyadari bahwa penyusunan laporan tugas akhir ini masih jauh dari sempurna, karena itu kami mengharapkan segala kritik dan saran yang membangun. Semoga tugas akhir ini dapat bermanfaat bagi kita semua. Atas perhatiannya kami ucapkan terima kasih.

Halaman ini sengaja dikosongkan

PERANCANGAN SISTEM *OBJECT DETECTION* PADA INDUSTRI ALAT BERAT MENGGUNAKAN *DEEP LEARNING*

Nama Mahasiswa	: Dikky Kurniawan
NIM	: 04211020
Dosen Pembimbing Utama	: Amalia Rizqi Utami, S.T., M.T.
Dosen Pembimbing Pendamping	: Barokatun Hasanah, S.T, M.T.

ABSTRAK

Proses identifikasi komponen alat berat secara manual seringkali memakan waktu dan rentan terhadap kesalahan, terutama saat teknisi harus memastikan kesesuaian suku cadang. Hal ini dapat memengaruhi efisiensi kerja dan kelancaran operasional industri. Penelitian ini menawarkan solusi berupa sistem otomatis yang mampu mengidentifikasi komponen alat berat dengan cepat dan akurat menggunakan teknologi deep learning. Sistem dikembangkan menggunakan algoritma YOLOv11 dan diimplementasikan pada perangkat Raspberry Pi 4 Model B dan Camera Pi sebagai input citra. Dataset terdiri dari 950 gambar yang mencakup delapan kelas komponen alat berat utama dan tiga objek tambahan sebagai gangguan (distractor). Dataset telah melalui proses augmentasi hingga menghasilkan 2.307 gambar yang dibagi menjadi data pelatihan, validasi, dan pengujian. Hasil deteksi objek disimpan secara otomatis di Firebase dan ditampilkan melalui dashboard web. Berdasarkan hasil pengujian, sistem mampu mendeteksi beberapa objek dalam satu frame dengan nilai mAP50 sebesar 93,2% dan mAP50-95 sebesar 64%. Namun, rata-rata confidence antar kelas saat kondisi multi-objek berada di rentang 0,55%–0,88%, yang masih berada di bawah target minimum 0,9. Beberapa kesalahan klasifikasi juga ditemukan, seperti deteksi background yang salah dikenali sebagai objek. Hasil ini menunjukkan bahwa sistem masih perlu ditingkatkan pada aspek kualitas dataset, penyetelan parameter model, serta stabilisasi perangkat keras. Penelitian ini diharapkan menjadi dasar dalam pengembangan sistem cerdas untuk mendukung efisiensi operasional industri alat berat.

Kata Kunci : *YOLOv11*, *Raspberry P*i, *Firebase*, Komponen Alat Berat

Halaman ini sengaja dikosongkan

OBJECT DETECTION SYSTEM DESIGN IN HEAVY EQUIPMENT INDUSTRY USING DEEP LEARNING

By	: Dikky Kurniawan
Student Identify Number	: 04211020
Supervisor	: Amalia Rizqi Utami, S.T., M.T.
Co-Supervisor	: Barokatun Hasanah, S.T, M.T.

ABSTRACT

The manual process of identifying heavy equipment components is often time-consuming and prone to errors, especially when technicians have to ensure the suitability of spare parts. This can affect work efficiency and smooth industrial operations. This study offers a solution in the form of an automated system that is able to identify heavy equipment components quickly and accurately using deep learning technology. The system was developed using the YOLOv11 algorithm and implemented on Raspberry Pi 4 Model B and Camera Pi devices as image input. The dataset consists of 950 images covering eight main classes of heavy equipment components and three additional objects as distractors. The dataset has gone through an augmentation process to produce 2,307 images divided into training, validation, and testing data. The object detection results are automatically stored in Firebase and displayed via a web dashboard. Based on the test results, the system is able to detect multiple objects in one frame with an mAP50 value of 93.2% and an mAP50-95 of 64%. However, the average confidence between classes when multi-object conditions are in the range of 0.55%–0.88%, which is still below the minimum target of 0.9. Some misclassifications were also found, such as background detection that was mistakenly recognized as an object. These results indicate that the system still needs to be improved in terms of dataset quality, model parameter tuning, and hardware stabilization. This research is expected to be the basis for developing intelligent systems to support the operational efficiency of the heavy equipment industry.

Keywords : *YOLOv11*, Raspberry Pi, Firebase, Heavy Equipment Components

Halaman ini sengaja dikosongkan

DAFTAR ISI

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB I

PENDAHULUAN

1.1 Latar Belakang

Alat berat merupakan kendaraan yang dirancang untuk melakukan pekerjaan berat seperti konstruksi, pertambangan dan pertanian. Alat berat berfungsi untuk memudahkan manusia dalam menyelesaikan pekerjaan yang membutuhkan tenaga besar, kecepatan, dan efesiensi yang tidak dapat dicapai secara manual (Afni et al., 2019). Alat berat terdiri dari ekskavator, bulldozer, dan creane. Setiap jenis alat berat memiliki peran penting dalam operasional di berbagai sektor industri. Contohnya, eksavator digunakan untuk menggali dan memindahkan tanah, sementara *bulldozer* untuk meratakan permukaan tanah dan mengangkat material berat, serta *creane* diperlukan untuk mengangkat dan memindahkan beban berat (M Tumengkol et al., 2021.

Pada alat berat terdapat komponen yang sangat penting antara lain *engine*, *pompa hydrolic*, kontrol *falet*, *motor hydrolic, silinder, final akuator,* dan *redaction* (Lase et al., 2020). *Engine* menyediakan tenaga utama, sementara pompa hidrolik menggerakkan cairan untuk menghasilkan tenaga mekanis. Kontrol *valve* mengatur aliran hidrolik ke motor dan silinder, yang mengubah tenaga tersebut menjadi gerakan. *Final actuator* mengendalikan bagian akhir seperti *bucket* atau *blade*, dan reduksi mengurangi kecepatan rotasi *engine* untuk stabilitas dan torsi optimal. Komponen-komponen ini bekerja sama untuk memastikan alat berat beroperasi efisien dan aman.

Identifikasi komponen dalam alat berat diperlukan untuk mendokumentasikan nama komponen dan nomor *part* secara tepat. Hal ini penting agar teknisi tidak perlu repot mencari tahu nama atau spesifikasi komponen ketika melakukan perawatan atau penggantian. Dengan data yang lengkap dan akurat, proses perbaikan menjadi lebih cepat dan efisien, karena teknisi dapat langsung menemukan komponen yang diperlukan tanpa menunda operasional alat berat. Identifikasi yang baik juga membantu dalam manajemen inventaris, memastikan ketersediaan suku cadang yang sesuai saat dibutuhkan (Lase et al., 2020).

Untuk mengatasi tantangan dalam mengidentifikasi komponen alat berat dengan cepat dan akurat, solusi yang diusulkan adalah penggunaan teknologi *deep learning*. Melalui sistem *object detection* berbasis *deep learning*, kamera atau sensor dapat mendeteksi dan mengenali setiap komponen alat berat secara otomatis, termasuk nama dan nomor partnya. Teknologi ini memungkinkan proses identifikasi yang lebih cepat dan mengurangi potensi kesalahan manual. Sistem tersebut akan mempelajari pola dan karakteristik visual dari setiap komponen melalui algoritma, sehingga teknisi hanya perlu menggunakan perangkat untuk mendapatkan informasi yang dibutuhkan secara instan (Azis, 2021). Dengan demikian, *deep learning* dapat mengotomatisasi proses pendataan komponen, meningkatkan efisiensi dan akurasi identifikasi.

Salah satu keuntungan utama dari penggunaan *deep learning* dalam sistem *object detection* adalah kemampuannya untuk memproses data dalam jumlah besar dengan cepat dan akurat. Teknologi ini menggunakan berbagai teknik salah satunya *YOLO* yang merupakan bagian dari algoritma *deep learning CNN* yaitu mendeteksi objek dengan fitur *bounding box* dan mengklasifikasikan gambar atau video. Metode ini memisahkan gambar ke dalam bentuk matriks SxS dengan setiap nilai pemisah kotak dengan m sebagai bentuk *bounding box* untuk melakukan klasifikasi kelas, memprediksi probabilitas dan nilai offset, sehingga *bounding box* yang melebihi nilai ambang batas probabilitas kelas dipilih untuk menemukan objek gambar(Cheng et al., 2025). *YOLO* memiliki beberapa versi dan *YOLOv11* merupakan versi terbaru dalam akurasi, kecepatan, dan efesiensi(Li et al., 2024).

Oleh karena itu, penelitian ini dilakukan untuk mengidentifikasi komponen-komponen alat berat yang memungkinkan sistem untuk mengenali berbagai komponen alat berat, termasuk nama dan nomor part-nya, melalui teknologi *deep learning* yang di implementasikan dalam sistem *object detection* menggunakan model *YOLOv11*(Huang et al., 2025). Dengan penerapan sistem ini, diharapkan teknisi dapat menemukan komponen yang diperlukan dengan cepat, tanpa perlu mencari secara manual, sehingga meningkatkan produktivitas dan efisiensi operasional.

1.2 Rumusan Masalah

Rumusan masalah pada penelitian ini sebagai berikut.

1. Bagaimana merancang sistem *object detection* yang dapat mendeteksi dan mengidentifikasi komponen alat berat dengan nilai akurasi ≤90?
2. Bagaimana mengimplementasikan metode *YOLOv11* ke dalam sistem *object detection*?
3. Bagaimana hasil analisis performansi deteksi dan identifikasi komponen alat berat menggunakan metode *YOLOv11*?
4. Bagaimana mengelola hasil deteksi objek ke dalam bentuk *dashboard* di *website*?

1.3 Tujuan Penelitian

Tujuan yang akan dilaksanakan pada penelitian ini sebagai berikut.

1. Merancang sistem *object detection* untuk mendeteksi dan mengidentifikasi komponen alat berat.
2. Mengimplementasikan metode *YOLOv11 (You Only Look Once)* ke dalam sistem *object detection*.
3. Mendapatkan hasil performansi deteksi dan identifikasi komponen alat berat menggunakan metode *YOLOv11* dengan nilai akurasi ≤90.
4. Mendapatkan hasil deteksi yang dapat dikelola dan divisualisasikan dalam bentuk dashboard pada website.

1.4 Manfaat

Manfaat yang dapat diambil dari tugas akhir ini sebagai berikut.

1. Penelitian ini bisa mempermudah proses identifikasi komponen alat berat, sehingga mengurangi kesalahan dalam pemilihan suku cadang.
2. Penelitian ini diharapkan bisa menjadi prototipe sistem yang membantu teknisi dalam mengenali dan mengidentifikasi komponen alat berat secara otomatis dan efisien.
3. Sistem ini dapat dijadikan dasar untuk pengembangan penelitian lebih lanjut di bidang teknologi deteksi dan industri 4.0.

4. Sistem ini bisa menjadi inovasi teknologi yang berpotensi meningkatkan efisiensi perawatan dan perbaikan, serta mengurangi waktu downtime peralatan.

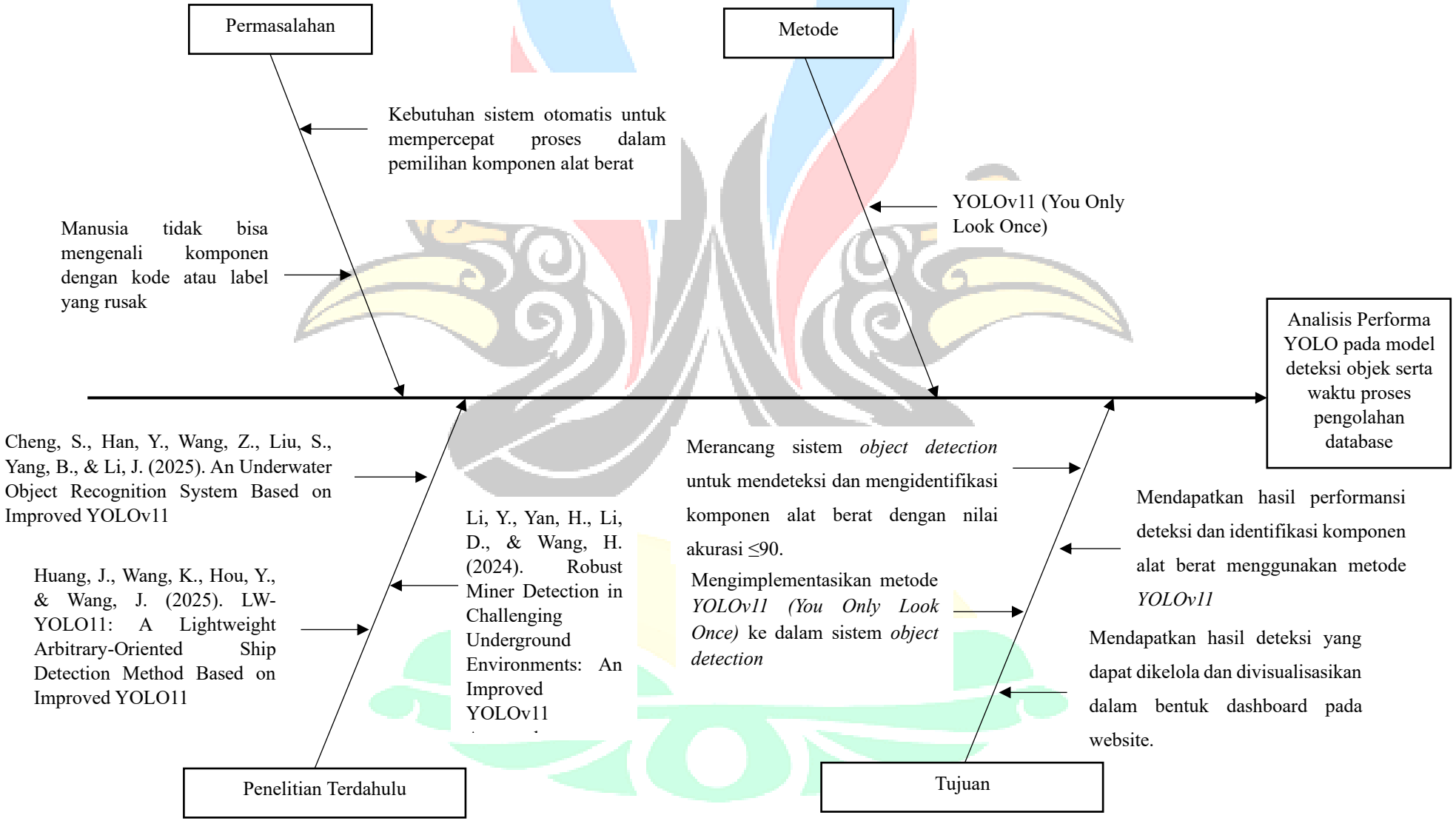
1.5 Batasan Masalah

Batasan masalah pada penelitian ini sebagai berikut.

1. Perancangan sistem hanya mengidentifikasi satu jenis komponen besar alat berat yang *available* untuk dijadikan objek.
2. Jumlah objek untuk deteksi sebanyak 5 kelas termasuk part penyusunnya.
3. Algoritma model yang digunakan pada penelitian ini adalah *YOLOv11 (You Only Look Once)*.
4. Dataset yang digunakan sebanyak 50 gambar per kelas termasuk komponen penyusun didalamnya.
5. Pengujian deteksi objek dilakukan pada jarak 1 sampai 2 m terhadap komponen.

1.6 Kerangka Penelitian

Penelitian Tugas Akhir ini memiliki kerangka penelitian sebagai berikut:



Gambar 1. 1 Kerangka Penelitian

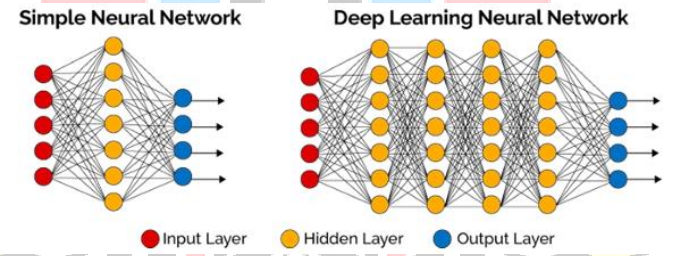
Halaman ini sengaja dikosongkan

BAB II

TINJAUAN PUSTAKA

2.1 Deep Learning

Deep learning adalah bagian algoritma dari machine learning yang terdiri dari beberapa lapisan tesembunyi dan merupakan salah satu cabang dalam kecerdasan buatan. Algoritma ini memungkinkan komputer untuk mempelajari pola yang kompleks dari data besar dengan menggunakan transformasi non-linear dan abstraksi model tingkat tinggi. Metodologi *deep learning* dapat melakukan suatu tugas seperti pengenalan gambar, pemrosesan suara, dan analisis teks yang sebelumnya sulit dicapai dengan algoritma konvesional (Mubin, 2021).



Gambar 2. 1 *Deep Leraning Layers (Dan & Mubin, 2021).*

Pada gambar 2.1 menunjukkan perbandingan antara jaringan saraf sederhana dan jaringan saraf mendalam, menunjukkan jumlah lapisan tersembunyi yang berbeda dalam setiap proses. *Deep learning* memiliki beberapa *library* pada Bahasa pemrograman python antara lain *tensorflow*, keras, *theano*, dan *pytorch*. *Deep learning* juga memiliki banyak model, yang memiliki algoritma dan fungsi yang berbeda. Berikut merupakan jenis model *deep learning* (Mubin, 2021):

1. *Convolutional Neural Network* (*CNN*)
2. *You Only Look Once (YOLO)*
3. *Artificial Neural Network* (ANN)
4. *Recurrent Neural Network* (RNN)
5. *Deep Belief Network (*DBN)

2.2 Perpustakaan *OpenCV*

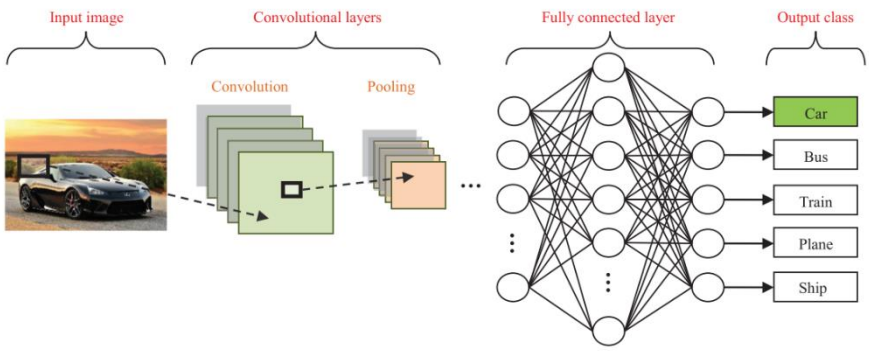
OpenCV (Open Computer Vision) adalah sebuah *library* perangkat lunak yang utamanya berfokus pada pemrosesan visi komputer secara *realtime*. Visi komputer merupakan bidang dalam kecerdasan buatan yang memungkinkan komputer untuk mengolah dan menafsirkan citra visual. Melalui kamera dan video sebagai input, visi komputer memungkinkan mesin (komputer) memahami, mengidentifikasi, serta mengklasifikasikan objek, kemudian menghasilkan output yang sesuai. *OpenCV*, sebagai *library Python* yang bersifat *open-source*, dapat digunakan secara gratis oleh para pengembang. Keunggulan dari *library* ini adalah kemampuannya dalam mengelola memori secara efisien, yang sangat berguna dalam memproses algoritma klasifikasi gambar atau video tanpa perlu khawatir dengan masalah dealokasi memori (Putra et al., 2023).

OpenCV mendukung beberapa bahasa pemrograman seperti *C++, Python, Java*, dan lainnya, serta tersedia pada berbagai sistem operasi termasuk *Windows, Linux, OS X, Android, dan iOS*. *OpenCV-Python* adalah API yang menghubungkan API C++ *OpenCV* dengan *Python*. Untuk menggunakan *OpenCV* dalam *Python*, pengguna dapat menginstalnya dengan "pip," yakni melalui perintah “pip install opencv-contrib-python” di *command line*, yang akan menginstal versi *OpenCV* terbaru, yaitu *cv2. OpenCV* juga menyediakan metode untuk mengonversi gambar berwarna menjadi *grayscale. Grayscale* adalah format citra di mana setiap *pixel* berada dalam rentang gradasi dari hitam hingga putih, dikenal sebagai derajat keabuan karena menampilkan variasi warna abu-abu antara nilai minimum (hitam) dan maksimum (putih). Umumnya, konversi warna ke *grayscale* dilakukan dengan memberikan bobot pada warna dasar merah, hijau, dan biru. Namun, cara yang lebih sederhana adalah dengan menghitung rata-rata dari ketiga warna tersebut dan menggunakannya sebagai nilai yang seragam untuk warna dasar dalam gambar (Wardoyo et al., 2014).

2.3 *Convolution Neural Network (CNN)*

Convolutional Neural Network (CNN) adalah jenis neural network yang dirancang khusus untuk memproses data berbentuk gambar. Ketika gambar diberikan sebagai *input, CNN* akan mengolahnya melalui beberapa lapisan untuk

mengekstraksi informasi yang relevan, sehingga gambar tersebut dapat dikenali dan diklasifikasikan. *CNN* terdiri dari tiga lapisan utama: *convolution layer* untuk mendeteksi fitur, *pooling layer* untuk mengurangi dimensi data, dan *fully-connected layer* yang berfungsi menggabungkan semua informasi untuk klasifikasi akhir (Mailoa & Santoso, 2022).



Gambar 2. 2 Arsitektur *Convolutional Neural Network* (Rawat & Wang, 2017)

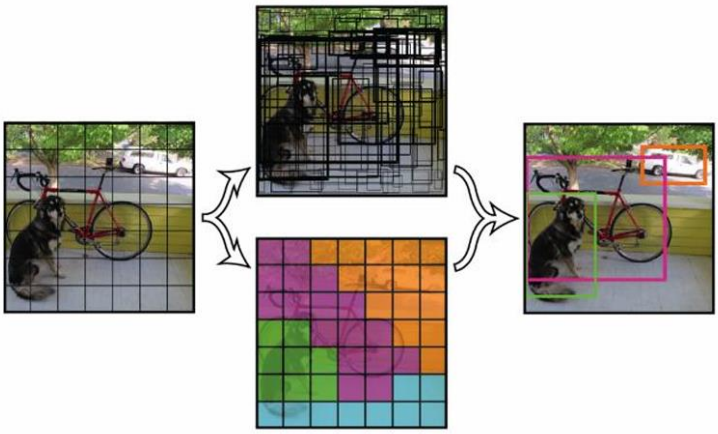
Arsitektur *Convolutional Neural Network (CNN)* terdiri dari beberapa lapisan penting yang berfungsi untuk mengekstrak dan mengklasifikasikan fitur dari gambar. Pada *convolution layer*, konvolusi dilakukan menggunakan filter berbentuk matriks, biasanya berukuran 3x3, untuk mengenali fitur-fitur utama dalam gambar yang diinput. Di layer ini, parameter *stride* mengatur seberapa jauh filter bergerak pada matriks, sedangkan *padding* (atau *zero padding*) digunakan untuk mempertahankan ukuran asli gambar. Selanjutnya, pada *pooling layer*, ukuran matriks diperkecil dengan mengambil nilai tertentu dari hasil *convolution layer*, menghasilkan satu nilai yang mewakili fitur dari kelompok yang diambil. Setelah melewati *pooling layer*, input diteruskan ke *fully connected layer*, yang merupakan lapisan akhir dari arsitektur *CNN*. Di sini, klasifikasi dilakukan terhadap fitur yang telah diproses; sebelum itu, fitur-fitur dari lapisan sebelumnya harus diubah menjadi vektor (*reshape* atau *flatten*) agar dapat digunakan sebagai input. Pada akhir proses, fungsi aktivasi seperti *softmax* atau *sigmoid* diterapkan untuk menentukan kategori berdasarkan nilai tertinggi yang dihasilkan (Anggarkusuma et al., 2020).

Dalam arsitektur *Convolutional Neural Network* (*CNN*), terdapat *feature extractor* untuk mengenali pola atau fitur utama dalam gambar, seperti tepi dan tekstur, melalui proses konvulasi pada gambar input. Setiap filter memiliki langkah

yang menentukan jarak perpindahan filter pada gambar, dan *padding* dapat melakukan penambahan piksel di sekitar tepi gambar agar dimensi output tetap sesuai dengan input. *Pooling layer* juga terdapat pada arsitektur berfungsi untuk mengurangi dimensi data dan menghasilkan fitur yang lebih kokoh. Ada dua teknik dalam pooling, yaitu *max pooling* yang memiliki nilai maksimum dari setiap area fitur, dan *average pooling* yang mengambil rata-rata dalam area fitur. Setelah fitur diekstraksi, fungsi aktivasi seperti *ReLU* (*Rectified Linear Unit*) diterapkan untuk menambahkan elemen non-linear, yang memungkinkan jaringan untuk mengenali pola yang lebih rumit. Untuk mempercepat pelatihan dan menjaga kestabilan distribusi data di setiap lapisan, digunakan teknik *batch normalization*. Teknik ini membantu mengurangi masalah pergeseran data internal, sehingga proses pelatihan menjadi lebih efisien dan stabil (Rawat & Wang, 2017).

2.4 *You Only Look Once (YOLO)*

You Only Look Once (YOLO) adalah salah satu metode yang umum digunakan untuk mendeteksi objek dalam sebuah gambar. Proses deteksi objek dimulai dengan mengidentifikasi lokasi dari setiap objek yang ingin dideteksi dalam gambar tersebut. *YOLO* menggunakan penerapan jaringan saraf tunggal *(single neural network)* untuk menganalisis keseluruhan gambar. Jaringan ini membagi gambar menjadi beberapa wilayah *(grid),* lalu melakukan prediksi untuk setiap wilayah *(bounding box)* guna mengklasifikasikan apakah itu objek atau bukan objek (Abby et al., 2023).



Gambar 2. 3 Proses Deteksi YOLO (Mailoa & Santoso, 2022)

YOLO membagi gambar input menjadi grid dengan ukuran S x S. Setiap sel dalam grid akan memprediksi *bounding box* dan nilai terkaitnya. Atribut yang dihasilkan mencakup empat koordinat (x, y, w, dan h) dari *bounding box*, serta *confidence score* yang menunjukkan probabilitas adanya objek di posisi tersebut. Koordinat x dan y menunjukkan posisi pusat dari *bounding box*, sementara w dan h merepresentasikan lebar dan tinggi perkiraan objek dalam gambar. *Confidence score* dihitung dengan mengalikan probabilitas keberadaan objek di grid dengan *Intersection Over Union* (IOU) antara *bounding box* prediksi dan *IOU ground truth* (Mailoa & Santoso, 2022).

Dalam mendeteksi sebuah objek menggunakan metode *YOLO* perlu dilakukannya tahapan, yaitu (Nazilly et al., 2020):

1. Membagi citra dalam grid untuk mendeteksi objek. Setiap grid akan diprediksi *bounding* box dan nilai confidence yang diperoleh melalui persamaan berikut.

$$Conf\,(class) = \Pr\,(class) \times IOU_{Pred}^{Truth}$$

   Pr(class) adalah probabilitas objek dalam suatu region dan IOU_{Pred}^{Truth} adalah rasio tumpang tindih (*Intersection Over Union*) antara kontak perdiksi dan kontak *ground truth*. *Pred* adalah luar area dalam kotak prediksi, *Truth* adalah area dalam *ground truth.* Makin besar nilai *IOU*, maka makin tinggi tingkat akurasi pendeteksiannya.
2. Setiap *bounding* terdapat nilai informasi yaitu *x,y,w,h dan c*. Pada nilai x dan y adalah koordinat titik tengah *bounding* box yang terprediksi, nilai w dan h adalah rasio ukuran lebar dan tinggi relatif terdahap grid, dan c adalah nilai *confidence bounding box* tersebut.
3. Setiap grid akan memprediksi nilai *class probabilitas* jika diprediksi terdapat objek di dalamnya.

2.4.1 *YOLOv11*

YOLOv11 merupakan versi terbaru dari keluarga algoritma You Only Look Once (YOLO) yang dikenal luas dalam bidang deteksi objek real-time. Dibandingkan dengan pendahulunya seperti YOLOv5 dan YOLOv7, YOLOv11

menghadirkan peningkatan signifikan baik dari sisi kecepatan inferensi maupun akurasi deteksi. Model ini dirancang untuk mengatasi tantangan pada deteksi objek kecil, kondisi occlusion (tumpang tindih antar objek), serta pencahayaan dan sudut pandang yang bervariasi di lingkungan nyata. Arsitektur YOLOv11 secara umum terdiri atas tiga bagian utama, yaitu backbone, neck, dan head. Pada bagian backbone, digunakan jaringan seperti Improved CSPNet yang dilengkapi dengan mekanisme Efficient Channel Attention (ECA) guna mengekstraksi fitur penting dari citra input secara efisien dan mendalam, khususnya untuk menangkap karakteristik visual objek-objek kecil. Selanjutnya, bagian neck menggunakan pendekatan seperti PANet atau BiFPN yang memungkinkan penggabungan fitur dari berbagai tingkat resolusi, sehingga informasi spasial dan kontekstual dari citra tetap terjaga dengan baik. Pada bagian head, YOLOv11 mengadopsi teknik Dynamic Head yang mampu menyesuaikan jalur pemrosesan fitur secara adaptif, menghasilkan prediksi akhir berupa bounding box, confidence score, dan label kelas objek dengan akurasi yang lebih baik. Selain itu, YOLOv11 dibekali fitur tambahan seperti adaptive anchor boxes yang secara otomatis menyesuaikan bentuk dan ukuran anchor terhadap distribusi data pelatihan, serta fungsi loss yang telah ditingkatkan menggunakan pendekatan Complete IoU (CIoU) untuk mempertimbangkan aspek posisi, ukuran, dan rasio aspek bounding box secara lebih presisi. Model ini juga menerapkan decoupled head untuk memisahkan proses klasifikasi dan regresi, serta menggunakan algoritma Non-Maximum Suppression (NMS) untuk menghindari duplikasi deteksi objek yang sama. Dengan arsitektur yang modular dan efisien, YOLOv11 sangat cocok digunakan pada perangkat dengan keterbatasan komputasi seperti Raspberry Pi, serta memungkinkan pengembangan dan fine-tuning yang fleksibel pada berbagai jenis dataset industri. YOLOv11 menjadi pilihan ideal dalam penerapan sistem deteksi objek berbasis deep learning yang membutuhkan akurasi tinggi dan kecepatan real-time, seperti dalam sistem identifikasi komponen alat berat pada lingkungan operasional yang dinamis (Husaini dkk, 2025).

2.5 *Object Detection*

Object detection adalah salah satu cabang dari *computer vision* yang bertujuan untuk mendeteksi dan mengklasifikasikan objek-objek tertentu dalam

sebuah gambar atau video secara otomatis(Bastian et al., 2016). Teknologi ini memiliki peran penting dalam berbagai aplikasi, mulai dari pengawasan keamanan, deteksi kendaraan, robotika, hingga industri berat, seperti alat berat. Pada dasarnya deteksi objek dilakukan menggunakan dua parameter untuk pengenalannya, yaitu *shape-color recognition* dan *color-size recognition* atau pengenalan bentuk dan warna. Kedua parameter ini dapat di implementasikan pada deteksi objek, akan tetapi proses pemilahan objek akan lebih selektif dengan menggunakan parameter jenis komponen. (Basrah Pulungan et al., 2021).

Pada industri alat berat, *object detection* memungkinkan sistem otomatis untuk mengenali dan melacak alat-alat berat yang beroperasi di lapangan. Hal ini sangat bermanfaat untuk mengurangi kecelakaan kerja, meningkatkan efisiensi operasional, serta mempermudah pengawasan dalam lingkungan kerja yang sulit dijangkau oleh manusia secara langsung. Penggunaan *object detection* pada industri ini juga bisa berkolaborasi dengan teknologi lain seperti *Internet of Things (IoT)* dan sensor untuk menciptakan sistem otomatisasi yang lebih canggih (Ren et al., 2015).

Object detection berbasis *deep learning* telah berkembang pesat berkat algoritma-algoritma seperti *YOLO (You Only Look Once), SSD (Single Shot Multibox Detector), dan Faster R-CNN.* Model-model ini mampu bekerja dengan sangat cepat dan akurat dalam mengenali objek dari berbagai gambar, bahkan dalam kondisi yang kurang ideal, seperti pencahayaan yang buruk atau sudut pandang yang tidak biasa (Redmon et al., 2016).

Proses utama dalam *object detection* meliputi tiga tahapan penting (Leibe et al., 2016):

1. Ekstraksi fitur dari gambar menggunakan algoritma *deep learning* seperti *Convolutional Neural Networks (CNN) dan YOLO (You Only Look Once).*
2. Klasifikasi objek yang terdeteksi di gambar.
3. Penentuan *bounding* box, yaitu wilayah gambar yang mengelilingi objek yang terdeteksi.

2.6 *Object Tracking*

Object tracking merupakan proses pergerakan objek yang terdeteksi dalam suatu video secara *real-time* dari satu *frame* ke *frame* berikutnya. Dalam bidang visi *computer*, tujuan dari *object tracking* adalah untuk mengenali posisi, orientasi, dan identitas objek yang sudah terdeteksi agar dapat terus dipantau dalam bentuk rekaman video. Teknik pelacakan objek memerlukan metode *deep learning* seperti *YOLO (You Only Look Once)* atau *CNN (Convolutional Neural Networks)* (Prabowo et al., 2018).

Object tracking sering kali digunakan dalam industri, khususnya pada pengawasan visual lalu lintas untuk deteksi, pelacakan, dan perhitungan kendaraan. Proses pelacakan objek dapat dilakukan melalui sistem berbasis *YOLO* dan *Convolutional Neural Network* (*CNN*). Algoritma *YOLO* digunakan untuk membagi citra grid dan memprediksi *bounding* box pada objek dengan *confidence* yang tinggi, sementara *CNN* bertugas melakukan ekstraksi fitur untuk klasifikasi objek. Pada *object tracking* juga juga terdapat tantangan dalam pelacakan, seperti perubahan nilai *confidence* akibat variasi pencahayaan, penutupan objek, dan perbedaan sudut pandang yang mempengaruhi akurasi deteksi. Untuk mengevaluasi performa sistem, digunakan metrik seperti *Intersection over Union (IoU)* dan *Mean Average Precision (mAP),* yang memberikan ukuran tingkat keberhasilan pelacakan dan perhitungan objek dalam berbagai kondisi(Arif Hudaya et al., 2020).

2.7 *Raspberry Pi*

Raspberry Pi adalah perangkat multifungsi seukuran kartu kredit, juga dikenal sebagai *single-board computer (SBC).* Perangkat komputasi dengan komputasi rendah ini dapat menjalankan berbagai aplikasi, antara lain pemutaran musik,video ,pengerjaan dokumen, media pembelajaran, permainan, dan aplikasi *IoT*. Perangkat ini berjalan pada sistem operasi *Raspberry Pi OS* berbasis *Linux Debian.*

Gambar 2. 4 *Raspberry Pi 4* (Pi, 2020)

Tabel 2. 1 Spesifikasi lengkap *Raspberry Pi 4 model B*

No.	Komponen	Spesifikasi lengkap
1.	Prosesor	Quad core Cortex-A72 (ARM v8), Broadcom BCM2711 64-bit @1.5GHz
2.	Memori	8GB LPDDR4-3200 SDRAM
3.	Konektivitas	Bluetooth 5.0, , 2 USB 3.0 ports; 2 USB 2.0, 2.4/5.0 GHz IEEE 802.11ac wireless, dan Gigabit Ethernet BLE
4.	Akses	40 Kepala GPIO
5.	Video & Suara	Konektor 3.5 mm, 1 MIPI CSI camera, 2 Mikro HDMI (4K60), 1 MIPI DSI display
6.	Video Grafis	Broadcom VideoCore IV @500 MHz
7.	Input Daya	PoE, 5V/3A DC (GPIO), 5V/3A DC (USB-C)
8.	Multimedia	Vulkan 1.0, H.265, OpenGL ES 3.1, H264
9.	SD Card	Maksimal 2 TB
10.	Daya	700 mA (3,5W)

Sumber: Datasheet Raspberry Pi 4,2023

2.8 *Rapberyy Pi Camera* Modul

Modul versi 3 *raspberry pi* ini diumumkan pada januari 2023 sebagai penerus dari versi sebelumnya. Peningkatan paling jelas yaitu pada sektor sensor, yang awalnya menggunakan sensor *OmniVision* OV5647 5 megapiksel kini diganti dengan sensor *Sony* IMX219 12 megapiksel. Modul ini dapat digunakan baik pemula maupun untuk tingkat lanjut, karena menawarkan banyak fitur salah satunya pengambilan gambar atau video definisi tinggi.

Gambar 2. 5 *Pi Camera Module* 3 (Rapberry Pi, 2023)

2.9 *Database System*

Basis data adalah sekumpulan data yang disimpan secara sistematis di komputer sehingga dapat diperiksa dengan program yang mengambil data dari basis data tersebut. Perangkat lunak yang mengelola dan mengeksekusi kueri terhadap basis data dikenal sebagai sistem manajemen basis data (DBMS). Data berbasis sistem dipelajari dalam ilmu informasi. Istilah "*database"* berasal dari bidang ilmu komputer (Andaru, 2018).

Basis data pada dasarnya adalah kumpulan catatan, atau potongan pengetahuan. Jenis fakta yang disimpan dalam database dijelaskan dengan cara yang terstruktur: skema adalah nama yang diberikan untuk deskripsi ini. Objek yang diwakili oleh database dan dijelaskan dalam skema. Memodelkan struktur basis data atau menyiapkan skema dapat dilakukan dengan berbagai cara: model basis data atau model data adalah dua nama untuk ini. Model relasional saat ini merupakan model yang paling banyak digunakan. Menurut istilah *layman*, model ini merepresentasikan semua informasi dalam bentuk tabel yang saling berhubungan dengan baris dan kolom di setiap tabel yang saling berhubungan dengan baris dan kolom di setiap tabel (Andaru, 2018).

Basis data adalah kumpulan file data yang saling terkait dan didata sedemikian rupa sehingga data tersebut dapat diakses dengan cepat dan mudah serta diolah menjadi informasi yang lebih bermanfaat. Database menyimpan data yang ada selain media penyimpanan; di sisi lain, data dikelola dan di proses oleh sistem basis data yang dikenal sebagai Sistem Manajemen Basis Data (Dwi Wijaya & Wardah Astuti, 2019).

2.10.1 *Database SQL*

Basis data relasional (DBR) adalah sistem penyimpanan data yang mengorganisir informasi dalam bentuk tabel yang terdiri dari baris dan kolom. Setiap tabel memiliki nama unik dan berisi data yang terstruktur, sehingga memudahkan pengaksesan dan pengelolaan informasi tanpa perlu mengubah struktur tabel tersebut. Dalam DBR, setiap baris diidentifikasi oleh kunci utama yang unik, yang memungkinkan hubungan antar tabel melalui kunci asing, sehingga data dapat saling terhubung dan diakses secara efisien (Silalahi & Wahyudi, 2018). Model ini sangat berguna untuk menjaga integritas data dan mengurangi redundansi melalui proses normalisasi, di mana data dipecah menjadi tabel-tabel kecil yang saling terkait. Beberapa contoh sistem manajemen basis data relasional (DBMS) yang populer termasuk *Oracle, DB2, Sybase, MySQL, MS SQL Server, dan MS Access.* DBR banyak digunakan dalam berbagai aplikasi bisnis dan keuangan karena kemampuannya dalam menangani volume data besar secara konsisten dan teratur.

2.10.2 *Database NoSQL (Not Only SQL)*

Perusahaan-perusahaan besar di era *Web 2.0* telah mulai mengembangkan atau menggunakan berbagai jenis basis data *NoSQL* untuk mengelola data mereka yang terus berkembang. Contoh basis data *NoSQL* yang populer antara lain *Google BigTable, LinkedIn Voldemort, Facebook Cassandra, dan Amazon Dynamo* (Silalahi & Wahyudi, 2018). Penggunaan *NoSQL* memungkinkan para pengembang untuk menciptakan aplikasi tanpa perlu mengubah struktur data di memori menjadi format relasional.Basis data *NoSQL* dirancang untuk menangani tantangan *Big Data* dengan memanfaatkan sistem yang terdistribusi. Mereka sering disebut juga sebagai basis data cloud dan dikembangkan untuk mengatasi masalah performa yang dihadapi oleh basis data relasional saat memproses data yang tidak terstruktur dan terus bertambah, seperti dokumen, email, multimedia, atau konten media sosial. *NoSQL* menyimpan dan mengambil data dalam berbagai format. Ada empat kategori utama basis data *NoSQL* (Silalahi & Wahyudi, 2018):

1. *Key-Value Store*: Ini adalah jenis penyimpanan data *NoSQL* yang paling sederhana, di mana setiap elemen data disimpan dalam pasangan kunci-

nilai. Penyimpanan ini memungkinkan fleksibilitas tinggi karena tidak memerlukan struktur tertentu untuk data yang disimpan.

2. *Column-Family*: Sistem ini menggunakan matriks jarang dan menyimpan data dalam bentuk baris dan kolom sebagai kunci. Data dikelompokkan berdasarkan kolom yang relevan.

3. *Graph Databases*: Data disimpan dalam bentuk grafis dengan entitas (*nodes*), atribut (*properties*), dan hubungan (*edges*) antar entitas. Ini sangat berguna untuk analisis hubungan kompleks antara berbagai elemen.

4. *Document Stores*: Mekanisme ini menyimpan struktur data hierarkis langsung ke dalam basis data, biasanya menggunakan format seperti JSON atau XML, sehingga mendukung penyimpanan informasi yang lebih kompleks dan terstruktur.

2.10 *Firebase*

Firebase adalah layanan *cloud* yang menyediakan solusi *backend* untuk pengembangan aplikasi, baik mobile maupun web. Didirikan oleh Andrew Lee dan James Tamplin pada tahun 2011 di San Francisco, California, *Firebase* meluncurkan *database real-time* pertamanya pada tahun 2012. Pada Oktober 2014, perusahaan ini diakuisisi oleh *Google*. Layanan yang ditawarkan oleh *Firebase* memungkinkan pengembang untuk membuat API yang dapat disinkronkan dengan berbagai klien dan menyimpan data di *cloud*. *Firebase* mendukung banyak bahasa pemrograman, termasuk *Android, iOS, JavaScript, Java, Objective-C, dan Node.js*. Beberapa fitur utama *Firebase* meliputi (Sonita, 2018):

1. Akses Offline: *Database real-time* dilengkapi dengan SDK yang memungkinkan penyimpanan data secara lokal. Ini berarti pengguna masih bisa mengakses data meskipun tidak terhubung ke internet.
2. Keamanan Tinggi: Dengan akuisisi oleh *Google*, *Firebase* menawarkan tingkat keamanan yang kuat untuk melindungi data pengguna.
3. Layanan Gratis: *Firebase* menyediakan opsi gratis bagi pengembang aplikasi, meskipun dengan beberapa batasan.

2.11 Penelitian Terdahulu

Tabel 2. 2 Penelitian Terdahulu

Peneliti	Judul	Hasil
Cheng, S., Han, Y., Wang, Z., Liu, S., Yang, B., & Li, J. (2025).	An Underwater Object Recognition System Based on Improved YOLOv11	Penelitian ini mengembangkan sistem pengenalan target bawah air dengan Jetson Xavier NX dan STM32 menggunakan algoritma DD-YOLOv11 mencapai akurasi 87,8% pada dataset RUOD.
Huang, J., Wang, K., Hou, Y., & Wang, J. (2025).	LW-YOLO11: A Lightweight Arbitrary-Oriented Ship Detection Method Based on Improved YOLO11	Penelitian ini dilakukan metode deteksi kapal ringan berbasis YOLOv11 yang ditingkatkan diajukan untuk mengatasi masalah ketidakseimbangan antara akurasi dan efisiensi deteksi. Metode ini menggunakan jaringan fusion fitur ringan, modul MSDA. Hasil eksperimen menunjukkan kinerja deteksi unggul dengan mAP@0.5 sebesar 97,6% pada dataset HRSC2016 dan 90,1% pada MMShip.
Li, Y., Yan, H., Li, D., & Wang, H. (2024).	Robust Miner Detection in Challenging Underground Environments: An Improved YOLOv11 Approach	Penelitian ini memperkenalkan dataset untuk deteksi penambang di tambang batubara bawah tanah, dengan mengatasi tantangan seperti kondisi pencahayaan rendah, gangguan cahaya terang, dan oklusi. Dengan mengintegrasikan mekanisme Efficient Channel Attention (ECA) ke dalam YOLOv11, kemampuan model untuk mendeteksi fitur utama ditingkatkan, sehingga meningkatkan akurasi dalam kondisi sulit.

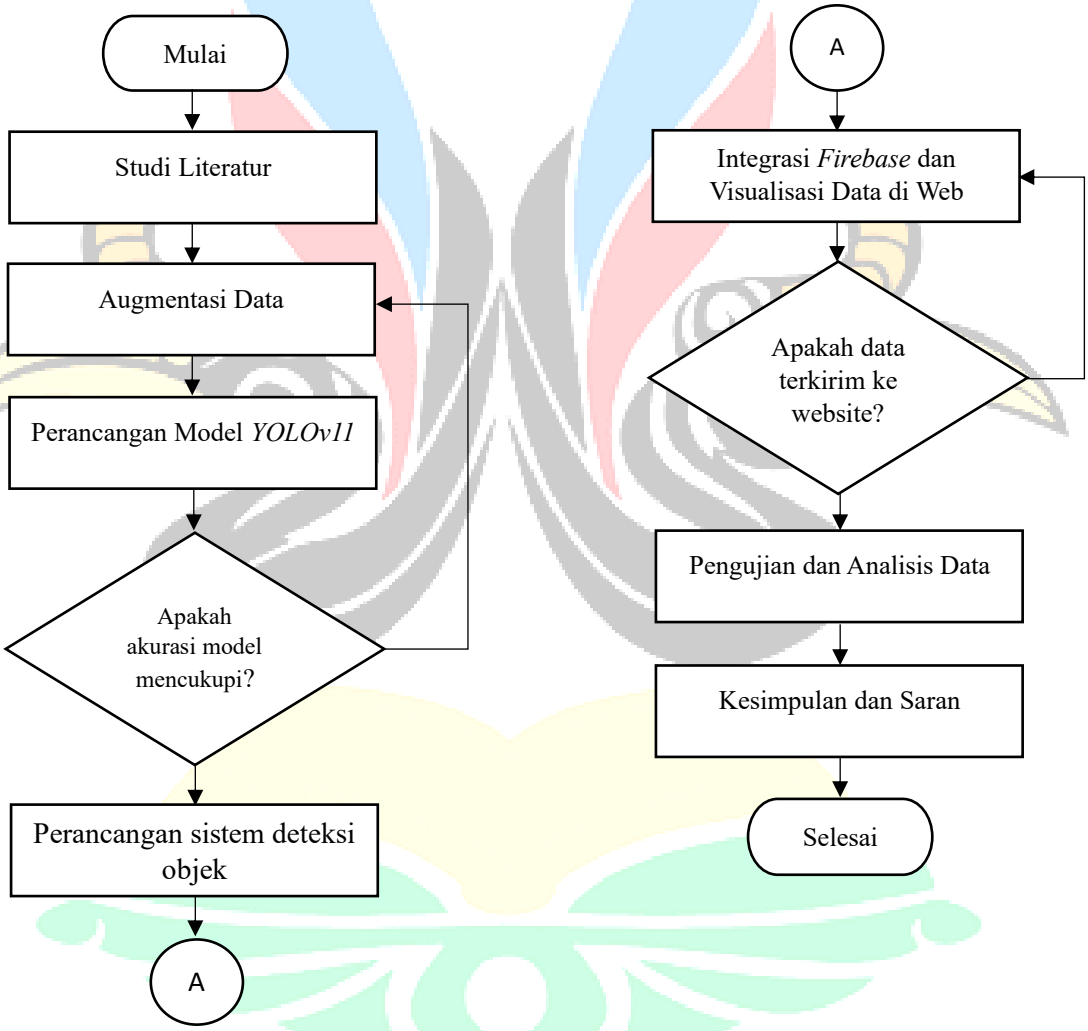
Halaman ini sengaja dikosongkan

BAB III

METODE PENELITIAN

3.1 Prosedur Penelitian

Prosedur penelitian diawali dengan menentukan metode penelitian, studi literatur, variabel penelitian, persiapan alat dan bahan perancangan deteksi objek, pengumpulan data primer, menganalisis hasil data dan pembahasan, serta penarikan kesimpulan dan saran.



Gambar 3. 1 Diagram Alir Penelitian (Penulis, 2025).

3.1.1 Studi Literatur

Tahap ini dilakukan pencarian serta pengkajian literatur baik berupa buku maupun hasil penelitian yang digunakan dalam pengerjaan tugas akhir. Teori- teori penunjang yang digunakan yaitu:

1. *Object Detection*
2. Perancangan Website
3. *YOLOv11*

3.1.2 Augmentasi Data

Augmentasi data adalah proses peningkatan variasi dataset dengan memodifikasi gambar asli menggunakan beberapa teknik yakni, rotasi. *flip*, blur, dan *noise*. Proses ini dilakukan untuk meningkatkan kemampuan generalisasi model *YOLOv11* agar dapat mengenali objek meskipun ada variasi dalam pencahayaan, posisi, atau skala objek dalam kondisi nyata. Dalam penelitian ini, beberapa teknik augmentasi data diterapkan antara lain rotasi gambar dengan sudut -15° hingga 15°, *flipping* horizontal dan vertikal untuk membalik gambar, blur dengan nilai 2,5px, dan noise sebesar 1,6%. Proses augmentasi ini dilakukan menggunakan platform Roboflow karena sudah mendukung untuk train ke model *YOLOv11.*

3.1.3 Perancangan Model *YOLOv11*

Perancangan model dalam penelitian ini dilakukan menggunakan pendekatan algoritma YOLOv11 (You Only Look Once Version 11), yang merupakan salah satu metode deteksi objek berbasis deep learning dengan kemampuan deteksi real-time dan akurasi tinggi. Proses perancangan model ini melibatkan beberapa tahap utama, yaitu persiapan dataset, pelabelan dan augmentasi data, pelatihan model, konversi model, hingga implementasi ke perangkat edge (Raspberry Pi) dan integrasi dengan sistem backend dan frontend. Dataset yang digunakan dalam penelitian ini merupakan citra komponen alat berat yang diperoleh dari hasil pengambilan gambar langsung di lapangan. Gambar-gambar tersebut diklasifikasikan ke dalam empat kelas utama, yaitu bolt, piston, cylinder head, dan plate. Seluruh gambar diberi label menggunakan platform Roboflow yang juga digunakan untuk melakukan augmentasi data guna

meningkatkan keberagaman dataset dan mengurangi risiko overfitting. Teknik augmentasi yang digunakan meliputi flip horizontal, rotasi ±15°, blur, dan penambahan noise.

Setelah proses anotasi dan augmentasi selesai, dataset diekspor ke format YOLOv11 dan digunakan untuk proses pelatihan model di lingkungan Google Colab dengan memanfaatkan GPU agar proses komputasi berjalan lebih cepat dan efisien. Proses pelatihan dilakukan dengan parameter yang telah disesuaikan, seperti batch size, epoch, learning rate, dan anchor box yang diadaptasi dari distribusi dataset. Model yang telah dilatih kemudian dievaluasi menggunakan metrik mAP50 dan mAP50-95 untuk mengukur akurasi deteksi. Model akhir diekspor dalam format ".pt" dan di-deploy ke perangkat Raspberry Pi 4 Model B, yang dilengkapi dengan Camera Pi sebagai sensor input citra. Deteksi objek dilakukan secara real-time di perangkat tersebut menggunakan Python dengan pustaka Ultralytics YOLOv11. Hasil deteksi, seperti nama objek, confidence score, dan timestamp, dikirim secara otomatis ke layanan Firebase (Firestore dan Storage). Data ini kemudian ditampilkan dalam bentuk dashboard web berbasis HTML, CSS, dan JavaScript agar dapat diakses oleh teknisi secara real-time melalui jaringan lokal atau internet. Perancangan model ini tidak hanya mempertimbangkan akurasi, tetapi juga efisiensi dan kompatibilitas terhadap perangkat keras terbatas, mengingat sistem ditujukan untuk digunakan di lapangan dengan infrastruktur minimal. Dengan rancangan ini, sistem diharapkan mampu membantu teknisi dalam mengidentifikasi komponen alat berat dengan lebih cepat, akurat, dan efisien.

3.1.4 Perancangan Sistem Deteksi Objek

Perancangan sistem yang dilakukan yaitu perancangan pengenalan citra objek pada komponen alat berat dengan hasil model yang telah dipilih melalui tingkat akurasi yang tinggi untuk diimplementasikan. Berikut alat dan komponen yang digunakan untuk perancangan sistem:

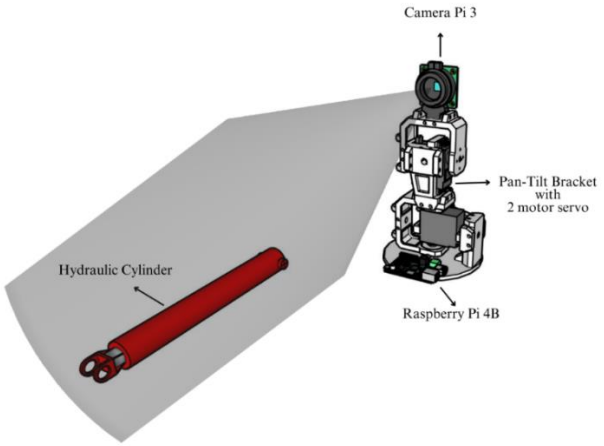
1. Raspberry Pi 4
2. Raspberry Pi Cam 3
3. Adaptor Raspberry Pi 5V/3A

Setelah menentukan alat dan komponen yang dibutuhkan, langkah selanjutnya adalah tahapan perancangan sistem deteksi objek. Peracangan ini meliputi beberapa tahapan yang bertujuan untuk memastikan setiap komponen berfungsi sesuai dengan sepsifikasinya yang dapat dilihat pada Gambar 3.3. Adapun tahapan-tahapan perancangan hardware yang dilakukan sebagai berikut.

1. Siapkan alat dan komponen
2. Mengubungkan ribbon cable Camera Pi ke port Csi Raspberry Pi
3. Memasang motor servo pada pan-tilt bracket
4. Menghubungkan motor servo ke pin GPIO Raspberry Pi
5. Menghubungkan adaptor 5V/3A ke raspberry Pi

Setelah dilakukan perancangan hardware, langkah berikutnya adalah instalasi dan konfigurasi perangkat lunak. Adapun tahapan konfigurasi perangkat lunak sebagai berikut.

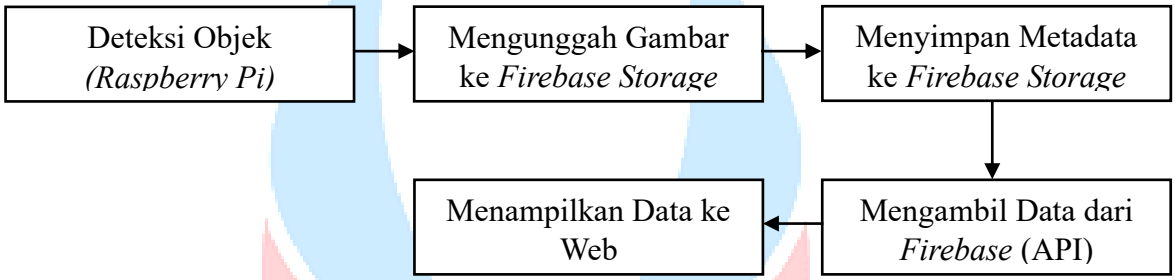
1. Mengintalasi library Rpi.GPIO, pigpio, dan OpenCV untuk pengendalian servo dan pemrosesan gambar.
2. Mengaktifkan modul kamera melalui konfigurasi Raspberry Pi dengan *command-line* (sudo raspi-config)
3. Menginstalasi dan konfigurasi Firebase API untuk menyimpan serta mengelola data hasil deteksi
4. Menghubungkan Firebase API ke dashboard web untuk menampilkan hasil deteksi



Gambar 3.2 Perancangan Sistem Deteksi Objek (Penulis, 2025)

3.1.5 Integrasi Firebase dan Visualisasi Data di Website

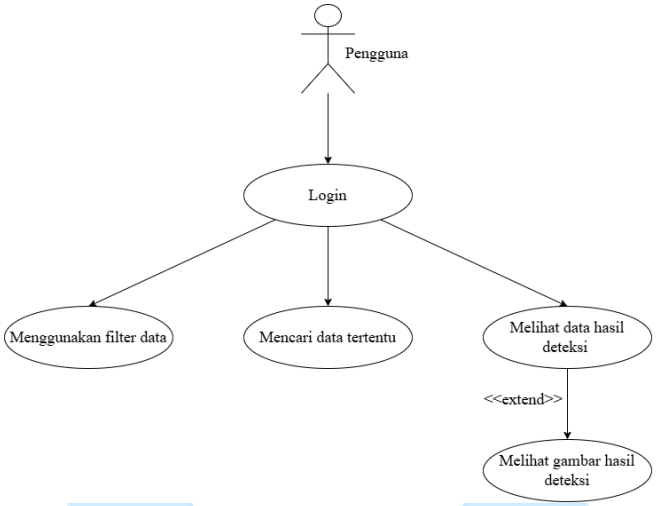
Integrasi *firebase* dilakukan untuk memastikan bahwa hasil deteksi objek tersimpan dan dikelola oleh *firebase storage*, serta dapat ditampilkan melalui website. Tedapat langkah langkah proses dalam pengintegrasian firbase.



Gambar 3. 3 Blok Diagram Integrasi *Firebase* (Penulis, 2024)

Berdasarkan Gambar 3.4, menjelaskan alur proses integrasi hasil deteksi dengan *firebase* dan data yang ditampilkan di *website*. Integrasi dimulai setelah perancangan model dan hardware selesai, dari perancangan tersebut dilakukan deteksi objek untuk menangkap gambar yang terdeteksi. Kemudian, hasil gambar yang berhasil terdeteksi diunggah ke *firebase storage* untuk menyimpan gambar komponen ataupun *metadata* seperti nama gambar komponen, waktu deteksi, dan *part number*. Tahapan terakhir dilakukan pengambilan data dari *firebase API* untuk menghubungkan server *firebase* dan *website* untuk menampilkan hasil data visual di *website*.

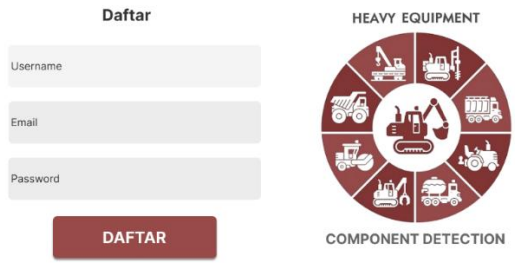
Visualisasi data di web dilakukan untuk menampilkan hasil deteksi objek yang disimpan di *firebase*. Data hasil deteksi seperti gambar dan informasi metadata (nama objek, waktu deteksi, dan tingkat akurasi) diambil dari *firebase firestore* dan *firebase storage.* Terdapat langkah proses dalam visualisasi data di *website*.

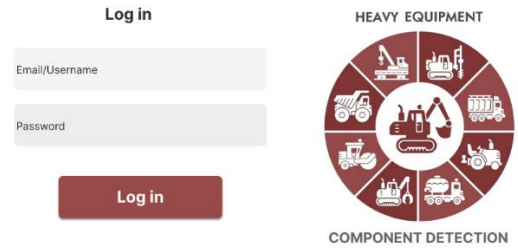
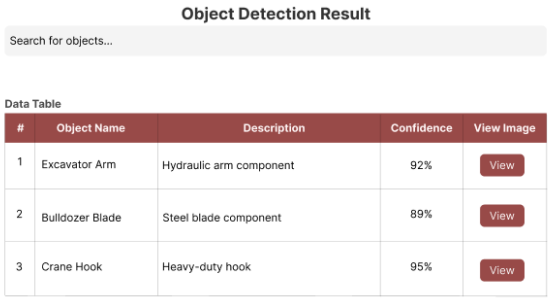


Gambar 3.4 *Use Case* Diagram Visualisasi data di *Web* (Penulis, 2024)

Pada Gambar 3.5 menggambarkan proses visualisasi data di website yang didalamnya terdapat beberapa fitur untuk pengguna untuk dapat melihat hasil deteksi objek dalam bentuk *dashboard*. Fitur dalam proses ini dilakukan login terlebih dahulu sebelum melihat hasil deteksi objek. Setelah login pengguna dapat melihat hasil deteksi objek dengan 3 fitur yaitu, pengguna dapat melihat hasil deteksi menggunakan filter data seperti waktu deteksi, data tertentu yaitu nama komponen atau *part number*, dan hasil deteksi dalam bentuk gambar. Berikut merupakan *design website* yang ditampilkan.

Tabel 3. 1 *Design Website*

Visualisasi Gambar pada Website	
Tampilan *Register*	

Visualisasi Gambar pada Website	
Tampilan *Log in*	
Tampilan Menu	

Sumber: Penulis, 2025

3.1.6 Pengujian Dan Analisis Data

Proses pengujian dan analisis data dilakukan untuk mengevaluasi kinerja sistem deteksi objek berbasis *deep learning* dan menentukan model untuk di implementasikan. Berikut merupakan proses pengujian dan analisis data.

3.1.6.1 Pengujian Sistem

Pengujian sistem dilakukan dari *functional test* pada sistem deteksi objek untuk memastikan sistem dapat melakukan deteksi. Kemudian, pengujian pada model *YOLOv11* dalam mendeteksi dan mengidentifikasi objek dan pengujian pada data yang diolah ke dalam bentuk *dashboard website* untuk melihat hasil deteksi objek.

3.1.6.2 Analisis Data

Analisis data dilakukan dengan fokus pada akurasi, *F1 score,* dan *Frame per second* untuk dibagian sistem *object detection*. Kemudian, analisis data juga dilakukan pada hasil deteksi objek yang diintegrasikan dengan *firebase* pada *dashboard website*. Dalam analisis ini juga memiliki kriteria yakni pada sistem

object detection nilai akurasi minimal ≤ 85%, *F1 Score* nilai minimal ≤ 85% dan FPS ≤ 20 $^{frame}/_{s}$.

3.2 Variabel Penelitian

Adapun variabel penelitian yang digunakan dalam penelitian ini terdiri dari variabel bebas dan variabel terikat. Berikut merupakan tabel variabel penelitian.

Tabel 3. 2 Variabel Penelitian

Variabel Bebas	Variabel Terikat
Deep Learning Model: YOLOv11	• *Accuracy (%)* • *F1 Score* • *Frame Rate* (Fps)
Koneksi internet	• Pengiriman data ke *website* • Rendering data ke *website*

Sumber: Penulis, 2025

BAB IV

HASIL DAN PEMBAHASAN

4.1 Sistem *Object Detection*

Sistem *object detection* merupakan inti dari proses identifikasi dan pelacakan objek dalam citra atau video secara otomatis. Dalam penelitian ini, sistem *object detection* dirancang untuk mendeteksi dan mengklasifikasikan berbagai komponen alat berat berdasarkan citra yang ditangkap oleh kamera. Teknologi ini memungkinkan sistem untuk tidak hanya mengenali keberadaan suatu objek, tetapi juga menentukan posisi objek tersebut dalam bentuk koordinat *bounding box*. Pengembangan sistem *object detection* melibatkan beberapa tahapan penting, mulai dari perancangan perangkat keras (*hardware*) sebagai alat pendukung akuisisi data, hingga pengolahan data yang bertujuan menghasilkan model deteksi yang andal.

Setiap tahap memiliki peran strategis dalam menjamin kualitas hasil deteksi, baik dari segi akurasi, kecepatan pemrosesan, maupun kemampuan generalisasi terhadap kondisi nyata di lapangan. Tahapan-tahapan utama yang dilakukan dalam pengembangan sistem ini meliputi:

a. Perancangan *Hardware*, yang mencakup pemilihan dan konfigurasi perangkat seperti kamera *Raspberry Pi,* sistem servo *pan-tilt*, dan unit pemroses data.

b. Pengambilan *Dataset*, yaitu proses mengumpulkan citra objek target dalam berbagai kondisi untuk memastikan keberagaman data.

c. Pengolahan *Dataset*, yang mencakup anotasi data, augmentasi, dan konversi data agar sesuai dengan format pelatihan pada model deteksi.

Dengan tahapan-tahapan tersebut, sistem *object detection* yang dibangun diharapkan mampu bekerja secara optimal dalam mengenali komponen alat berat dan memberikan hasil deteksi yang akurat untuk mendukung proses identifikasi di lingkungan industri.

4.1.1 Perancangan *Hardware*

Perancangan *hardware* dilakukan dengan menggunakan *microcomputer* berupa *Raspberry Pi 4* yang dilengkapi dengan kamera *Pi* sebagai input citra dan

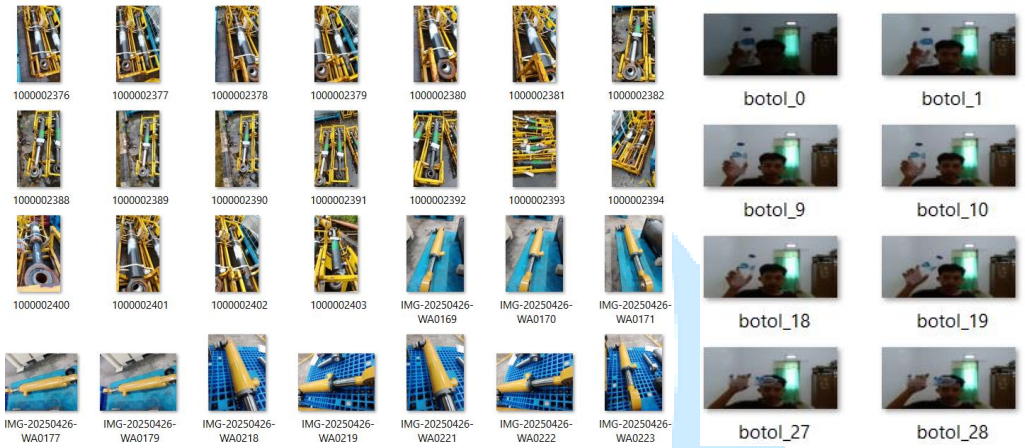
MicroSD sebagai penyimpanan lokal. Kemudian, diperlukan adaptor daya 5 Volt 2 Ampere sebagai pendukung operasional perangkat secara *realtime*.

Gambar 4. 1 Sistem *Object Detection*

Berdasarkan gambar 4.1, menunjukkan rangkaian sistem *object detection* yang dilengkapi dengan *pan-tilt HAT* untuk menambahkan sistem *object tracking*. *Pan-tilt HAT* memiliki dua servo yang menggerakkan kamera pada dua sumbu, yaitu *horizontal (pan)* dan *vertical (tilt)*. Kamera *Pi* terpasang pada dudukan *pan-tilt* yang terhubung langsung ke *raspberry Pi* melalui port CSI. Sistem ini dirancang untuk mendukung fungsi *object detection* dan *tracking*. Ketika objek terdeteksi, servo akan menggerakkan kamera untuk mengikuti posisi objek secara *realtime.*

4.1.2 Pengambilan Dataset

Proses pengambilan data untuk keperluan dataset dilakukan menggunakan kamera ponsel, dengan cara mengambil gambar komponen alat berat secara langsung di lapangan. Dataset yang dihasilkan terdiri dari dua kategori, yaitu 800 gambar untuk dataset komponen alat berat dan 150 gambar untuk dataset *custom* objek. Sehingga, total seluruh dataset yang akan digunakan sebanyak 950 data. Gambar dan tabel berikut ini menunjukkan hasil pengambilan data yang dilakukan di lingkungan industri alat berat.



Gambar 4. 2 Pengambilan Dataset (Penulis, 2025)

Tabel 4. 1 Jumlah Data

Nama Kelas	Jumlah Data
Bolt	100
Cylinder Barel	100
Cylinder Head	100
Nut	100
Piston	100
Plate	100
Rod	100
Steering Cylinder	100
Botol	50
Phone	50
Pen	50
Total Data	**950**

Sumber: Penulis,2025

Kemudian data akan dibagi menjadi tiga bagian, yakni data train, data validasi, dan data test. Pembagian ini dilakukan agar pengujian dapat menghasilkan nilai akurasi yang tinggi.

4.1.3 Pengolahan Dataset

Pengolahan dataset dilakukan untuk mempermudah sistem dalam mengenali objek komponen alat berat secara akurat. Proses ini dimulai dengan pengumpulan gambar dari berbagai sudut dan kondisi pencahayaan, kemudian dilakukan pelabelan (anotasi) menggunakan platform Roboflow sesuai dengan kelas objek seperti bolt, piston, cylinder head, dan plate. Setelah proses anotasi selesai, dilakukan augmentasi data untuk meningkatkan variasi visual dalam dataset dan mengurangi risiko overfitting saat pelatihan model.

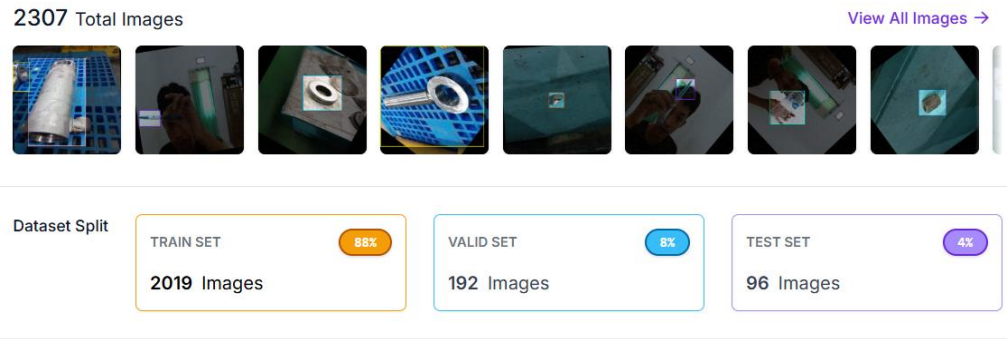
Teknik augmentasi yang digunakan meliputi flip, rotasi (±15°), blur, dan penambahan noise. Setiap teknik augmentasi bertujuan untuk meniru kemungkinan variasi kondisi nyata yang terjadi di lapangan, seperti perubahan orientasi komponen, getaran kamera, atau kualitas gambar yang menurun. Augmentasi juga berperan dalam meningkatkan ketahanan model terhadap perubahan lingkungan visual saat implementasi di perangkat seperti *Raspberry Pi*. Berikut ini adalah contoh hasil dari masing-masing metode augmentasi yang diterapkan.

Tabel 4. 2 Hasil Augmentasi

Gambar Asli	Flip	Rotasi	Blur	Noise
				

Sumber: Penulis, 2025

Tabel 4.2 menunjukkan hasil dari masing-masing metode augmentasi yang diterapkan pada citra komponen alat berat, dengan kelas bolt sebagai contoh. Teknik flip mencerminkan kondisi orientasi objek dari sisi berlawanan, sementara rotasi sebesar ±15° mensimulasikan perubahan sudut pengambilan gambar di lapangan. Teknik blur diterapkan untuk merepresentasikan kondisi ketajaman citra yang rendah akibat getaran atau fokus kamera yang kurang, dan penambahan noise digunakan untuk menguji ketahanan model terhadap gangguan visual yang mungkin terjadi dalam lingkungan industri. Dengan adanya variasi citra hasil augmentasi ini, diharapkan model YOLOv11 mampu belajar lebih baik dan memberikan hasil deteksi yang lebih akurat serta stabil meskipun menghadapi kondisi pencitraan yang tidak ideal. Proses augmentasi ini juga turut meningkatkan ukuran dataset secara signifikan, sehingga mendukung proses pelatihan model yang lebih general dan tidak mudah *overfitting*.



Gambar 4. 3 Split Dataset Setelah Augmentasi (Penulis, 2024)

Berdasarkan gambar 4.3, melalui proses augmentasi yang dilakukan dari data awal, jumlah dataset pelatihan berhasil diperbanyak. Hasil akhir diperoleh total 2307 gambar, yang dibagi menjadi tiga kelompok, yaitu :

a. 88% untuk data pelatihan (2019 gambar)
b. 8% untuk data validasi (192 gambar)
c. 4% untuk data pengujian (96 gambar)

4.1.4 Training Model

Pelatihan model dilakukan menggunakan *google colab* dengan dataset yang telah di proses melalui *roboflow* dan diunduh dalam format zip. Kemudian file di upload ke dalam internal *google colab* untuk dilatih menggunakan model *YOLOv11*. Untuk melatih dataset ini cukup menggunakan file data.yaml, kemudian konfigurasi dengan menjalankan perintah *YOLOv11* di *notebook* (colab). Setelah file dataset di konfigurasi dengan model *YOLOv11*, *google colab* akan melatih dataset tersebut.

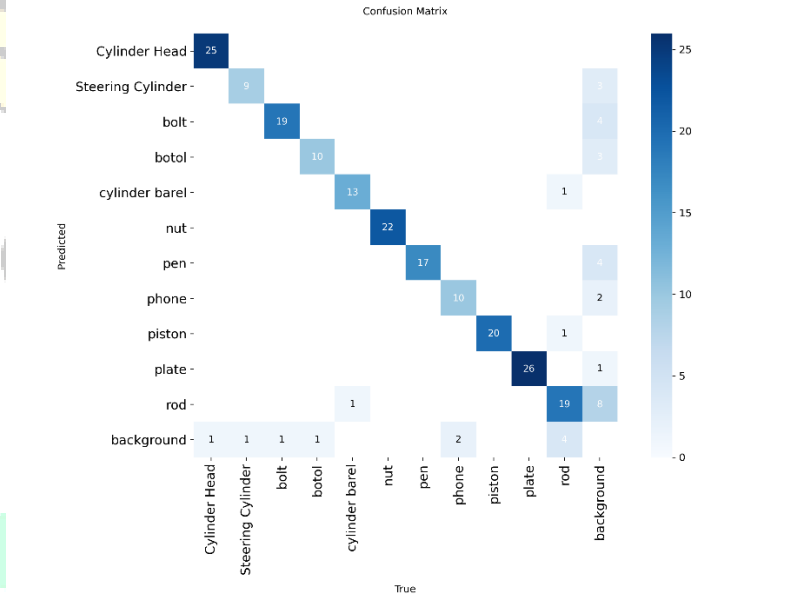
```
100 epochs completed in 1.005 hours.
Optimizer stripped from runs/detect/train/weights/last.pt, 5.5MB
Optimizer stripped from runs/detect/train/weights/best.pt, 5.5MB

Validating runs/detect/train/weights/best.pt...
Ultralytics 8.3.138 🚀 Python-3.11.12 torch-2.6.0+cu124 CUDA:0 (Tesla T4, 15095MiB)
YOLO11n summary (fused): 100 layers, 2,584,297 parameters, 0 gradients, 6.3 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 6/6 [00:04<00:00,  1.47it/s]
                   all        192        203      0.903      0.894      0.932       0.64
          Cylinder Head         26         26          1      0.956      0.995      0.774
      Steering Cylinder          9         10      0.887      0.783      0.932      0.382
                  bolt         17         20      0.883       0.95      0.917      0.522
                botol          11         11       0.87      0.909      0.949      0.699
         cylinder barel         14         14      0.919      0.929      0.961      0.571
                   nut         22         22      0.982          1      0.995      0.811
                   pen         17         17      0.879      0.854      0.969       0.53
                 phone         12         12      0.837      0.857      0.945      0.736
                piston         20         20      0.971          1      0.995      0.871
                 plate         26         26       0.97          1      0.995      0.865
                   rod         25         25      0.734        0.6      0.603      0.274
Speed: 0.3ms preprocess, 4.0ms inference, 0.0ms loss, 4.8ms postprocess per image
Results saved to runs/detect/train
```

Gambar 4. 4 Hasil *Training* (Penulis, 2025)

Berdasarkan gambar 4.4, pelatihan model telah berhasil dilakukan dengan selang waktu 1,005 *hours* untuk mencapai 100 *epochs*. Hasil pelatihan menunjukkan bahwa model mencapai performa yang baik, dengan nilai *mean average precision* (mAP) pada IoU 0,5 (mAP50) sebesar 93,2%, dan mAP50-95 sebesar 64% untuk seluruh kelas. Selain itu terdapat hasil proses pelatihan lainnya sebagai berikut.

A. Confusion Matrix

Confusion matrix merupakan hasil tampilan visual berbentuk matrix yang digunakan untuk mengevaluasi kinerja model dalam mengklasifikasi objek ke dalam kelas yang di tentukan. Pada sistem deteksi objek, *confusion matrix* berfungsi untuk menggambarkan jumlah prediksi yang benar dan salah terhadap setiap kelas objek yang terdeteksi oleh model. Matriks menampilkan hubungan antara label *ground trouth* dan hasil prediksi model untuk mengukur tingkat akurasi, presisi, dan kesalahan klasifikasi pada setiap kelas.



Gambar 4. 5 *confusion matrix YOLOv11* (Penulis, 2025)

Berdasarkan gambar 4.5, menunjukkan kinerja model dalam klasifikasi masing-masing komponen alat berat yang terdeteksi. Setiap baris merupakan hasil perdiksi model, sedangkan setiap kolom menunjukkan jumlah prediksi yang lebih tinggi, dan pada warna lebih terang adalah frekuensi yang lebih rendah. Kemudian terlihat bahwa model memiliki performa yang baik dalam mengklasifikasikan kelas

objek. Hal ini terlihat dari banyaknya nilai prediksi yang berada di diagonal utama matriks. Terdapat beberapa keberhasilan klasifikasi yang tinggi antaranya:

1. *Cylinder Head* berhasil di prediksi dengan benar sebanyak 25 kali tanpa prediksi yang salah ke kelas lain.
2. *Plate* diprediksi dengan benar sebanyak 26 kali dengan hanya satu *instance* yang salah di klasifikasikan sebagai kelas *rod*.
3. *Nut* menunjukkan klasifikasi yang baik dengan 22 prediksi benar.

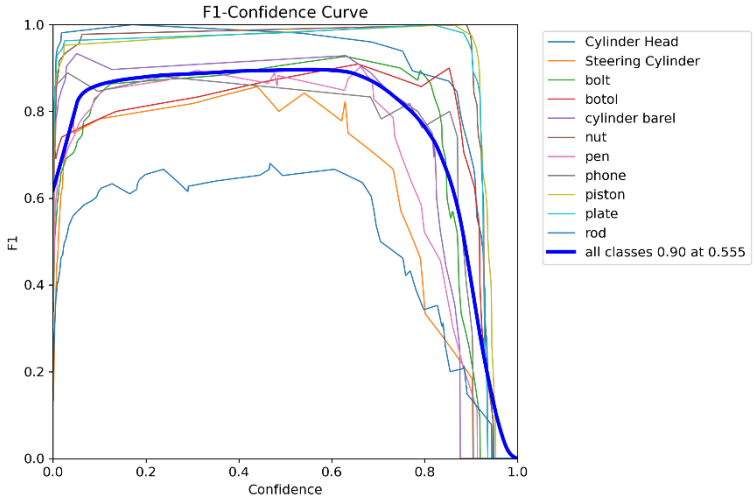
Namun demikian, masih terdapat beberapa kesalahan klasifikasi (*misclassification)*, antara lain:

1. *Rod* diprediksi dengan benar sebanyak 19 kali, namun salah diklasifikasikan sebagai *background* sebanyak 4 kali dan satu kali sebagai *piston*.
2. *Phone* mengalami salah klasifikasi ke pen sebanyak 2 kali
3. *Pen* diklasifikasikan ke *background* sebanyak 2 kali serta memiliki beberapa prediksi kesalahan ke kelas lain.

Selain itu, label Background yang semestinya tidak mengandung objek tertentu, justru beberapa kali salah diklasifikasikan sebagai objek, seperti Cylinder Head, Bolt, dan Nut, masing-masing sebanyak satu kali. Kondisi ini mengindikasikan adanya kemungkinan terjadinya overfitting atau ketidakseimbangan data pada kategori latar belakang (background) selama proses pelatihan model.

B. F1 *Curve*

F1 curve adalah grafik yang menggambarkan nilai dari *f1 score* dari model selama pelatihan. *F1 score* merupakan rata-rata kesesuaian dari *precision, recall*, dan menjadi indikator penting dalam menilai peforma model. Grafik *f1 curve* membantu dalam menentukan nilai ambang (*threshold*) *confidence* yang optimal untuk digunakan saat proses gangguan.

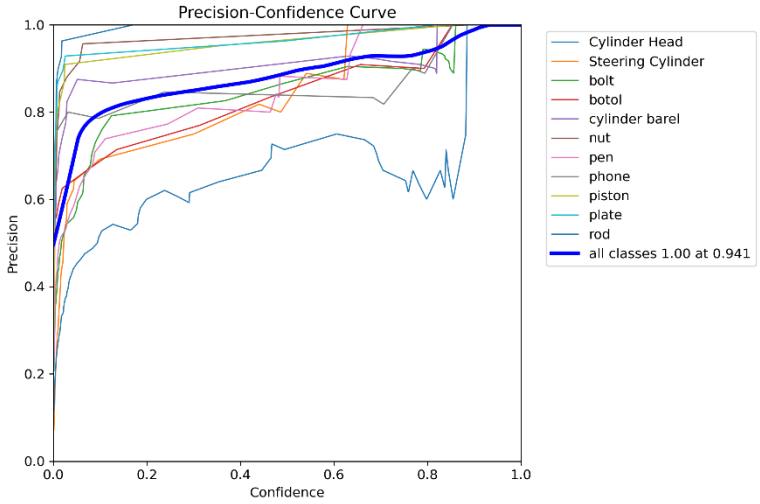


Gambar 4. 6 F1 *confidence curve* (Penulis, 2025)

Berdasarkan gambar 4.6, nilai *f1 score* meningkat seiring bertambahnya nilai *confidence* hingga titik tertentu, sebelum akhirnya menurun ketika ambang batas semakin tinggi. Hal ini menunjukkan bahwa pada nilai *confidence* yang terlalu rendah dan model menghasilkan banyak prediksi positif dengan tingkat kesalahan tinggi. Sedangkan pada *confidence* yang terlalu tinggi menghasilkan banyak deteksi yang diabaikan (*missed detections*) sehingga nilai *recall* menurun. Kurva tebal bewarna biru pada grafik merupakan tampilan gabungan dari seluruh kelas yang menunjukkan bahwa nilai *f1 score* tertinggi untuk semua kelas tercapai sebesar 0,90 pada *confidence* 0,555. Titik ini digunakan sebagai batas optimal untuk implementasi sistem dalam mendeteksi objek dengan keseimbangan antara *precision* dan *recall*.

C. *Precision Confidence Curve*

Precision confidence curve adalah grafik yang menggambarkan hubungan antara nilai *confidence* dengan nilai *precision* pada setiap kelas objek. *Precision* mengukur banyak prediksi positif yang benar dari seluruh prediksi positif yang dihasilkan oleh model. Semakin tinggi *precision*, maka akan sedikit kesalahan *false positives* yang dilakukan model.

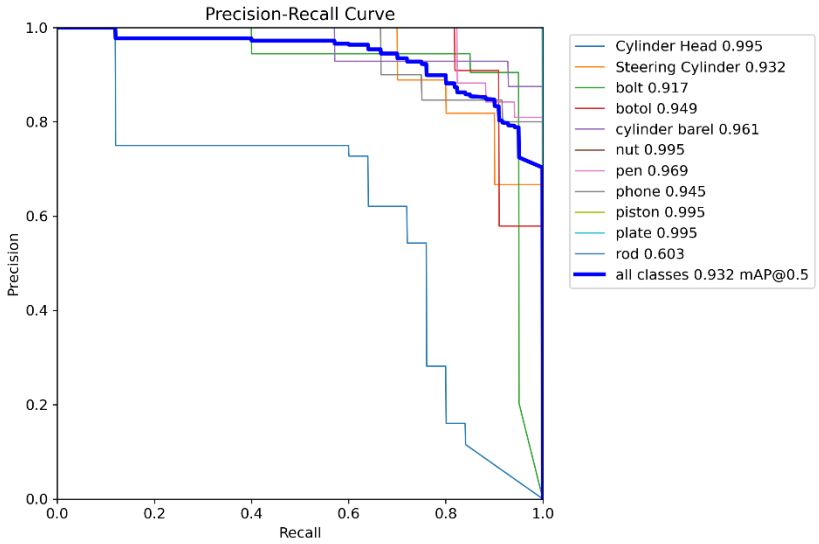


Gambar 4. 7 *precision confidence curve* (Penulis, 2025)

Berdasarkan gambar 4.7, terlihat bahwa *precision* untuk seluruh kelas mencapai nilai maksimum (1.00) pada nilai *confidence* 0.941, yang ditunjukkan oleh garis tebal biru. Jika ambang *confidence* ditetapkan pada nilai tersebut, maka seluruh prediksi yang dibuat oleh model memiliki tingkat ketepatan yang sangat tinggi. Setiap garis warna pada grafik menampilkan setiap kelas objek dan dari kurva tersebut terlihat bahwa sebagian besar kelas memiliki nilai *precision* yang tinggi sebesar 0.8, pada rentang *confidence* menengah hingga tinggi. Namun demikian, terdapat variasi performa antara kelas lainnya seperti, *cylinder head* dan *steering cylinder* yang menunjukkan fluktuasi *precision* yang lebih besar dibandingkan dengan kelas *plate* atau *rod* dengan nilai kurva lebih stabil.

D. *Precision Recall Curve*

Precision recall curve adalah grafik yang digunakan untuk mengevaluasi performa model deteksi objek, ketika kinerja model pada dataset dengan kelas tidak sepadan. Grafik ini menunjukkan hubungan antara dua metrik, yaitu *precision* dan *recall*. PR *curve* menggambarkan bagaimana *precision* berubah terhadap *recall* saat batas Keputusan (*confidence threshold*) bervariasi. Idealnya, model yang baik memiliki kurva yang tetap tinggi di kedua metrik, artinya model mampu mendeteksi objek secara lengkap tanpa banyak menghasilkan kesalaha deteksi.

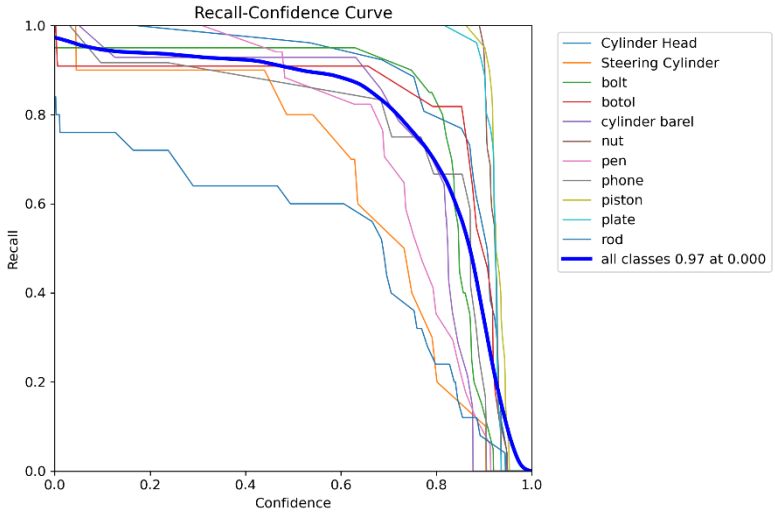


Gambar 4. 8 *precision recall curve* (Penulis, 2025)

Berdasarkan gambar 4.8, menunjukkan performa masing-masing kelas objek. Pada garis biru tebal menampilkan peforma rata-rata dari seluruh kelas dengan skor mAP0.5, sebesar 0.932. Hal ini menunjukkan bahwa secara keseluruhan model mampu mendeteksi objek dengan baik pada ambang IoU 0.5. Kelas *cylinder head*, *nut*, *piston*, dan *plate* memiliki *precision* serta *recall* yang hampir sempurna (0.995), menunjukkan bahwa model sangat akurat untuk mendeteksi objek ini. Namun, kelas *rod* hanya memiliki nilai 0.603, menunjukkan performa deteksi yang lebih rendah, karena data pelatihan yang terbatas, tampilan visual sama dengan objek lain, dan bentuk objek tidak konsisten.

E. *Recal Confidence Curve*

Recall confidence curve menggambarkan hubungan antara nilai *recall* dan *confidence threshold*. Grafik ini menunjukkan sensitif model dalam mendeteksi objek pada banyak tingkat *confidence*. Nilai *recall* tinggi dengan *threshold* rendah untuk menunjukkan bahwa model mampu mendeteksi Sebagian besar objek.



Gambar 4. 9 *recall confidence curve* (Penulis, 2025)

Berdasarkan gambar 4.9, terlihat kurva biru tebal menunjukkan performa dari semua kelas dengan nilai *recall* maksimum (0.97) pada *confidence* (0.0). hal ini dapat diartikan jika *threshold* dibuat sangat rendah, model berhasil menangkap hampir semua objek yang berpotensi menghasilkan lebih banyak *false positive*. Semakin meningkatnya *confidence*, *recall* Sebagian besar kelas menurun di atas *threshold* (0.6-0.8) yang terlihat pada penurunan tajam kurva. Kelas seperti *plate*, *piston*, dan *nut* memiliki kurva yang mendekati puncak (stabil *recall* tinggi hingga *confidence* tinggi). Sedangkan *rod* dan *steering cylinder* menunjukkan penurunan *recall* yang lebih awal, menunjukkan kemungkinan kesulitan dalam deteksi objek berkala.

4.2 Implementasi Metode *YOLOv11*

Pada bagian ini dibahas proses implementasi metode *You Only Look Once* versi 11 (*YOLOv11*) ke dalam sistem deteksi dan identifikasi komponen alat berat yang telah dirancang sebelumnya. *YOLOv11* merupakan salah satu algoritma deteksi objek terkini yang dikenal memiliki keunggulan dalam hal kecepatan pemrosesan serta tingkat akurasi yang tinggi, sehingga sangat sesuai untuk kebutuhan deteksi objek secara waktu nyata (*real-time*). Implementasi metode ini mencakup proses integrasi model ke dalam sistem perangkat keras dan perangkat lunak, serta dilanjutkan dengan pengujian sistem untuk mengevaluasi kinerja model

dalam mendeteksi objek, baik secara kualitatif maupun kuantitatif. Pembahasan dalam subbab ini dibagi menjadi tiga bagian utama.

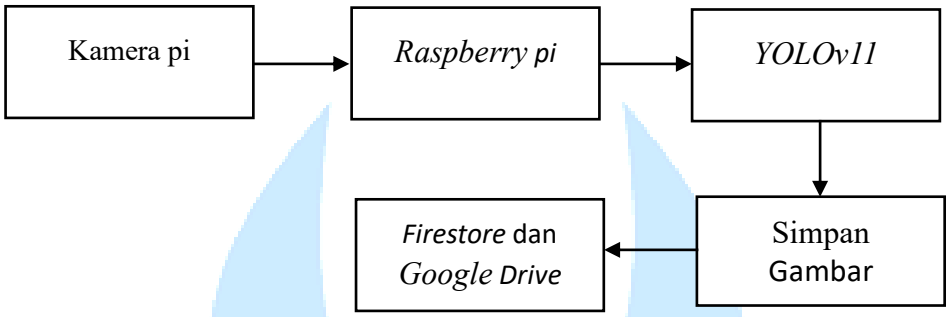
a. Pertama, dijelaskan mengenai proses integrasi *YOLOv11* ke dalam sistem, yang meliputi alur kerja sistem, pemanfaatan perangkat keras seperti kamera Pi dan *Raspberry Pi*, penggunaan perangkat lunak *Python* dan pustaka *OpenCV*, serta mekanisme pemrosesan data mulai dari pengambilan citra hingga penyimpanan dan visualisasi hasil deteksi.
b. Kedua, dilakukan pengujian sistem deteksi tanpa label, yaitu pengamatan terhadap hasil deteksi objek yang ditampilkan oleh sistem secara visual tanpa perbandingan terhadap data anotasi. Pengujian ini bertujuan untuk mengevaluasi kemampuan sistem dalam mengenali dan memberikan label pada objek secara langsung berdasarkan input citra yang diterima.
c. Ketiga, dilakukan pengujian sistem deteksi dengan label, yang bertujuan untuk menilai performa model secara kuantitatif. Evaluasi dilakukan menggunakan sejumlah metrik pengukuran seperti *precision*, *recall*, *F1-score*, dan *mean average precision* (*mAP*), serta dilengkapi dengan analisis grafik performa dan *confusion matrix* guna memperoleh gambaran menyeluruh terhadap kemampuan deteksi sistem.

Melalui proses implementasi dan pengujian ini, diharapkan sistem yang telah dikembangkan mampu memberikan hasil deteksi dan identifikasi komponen alat berat secara akurat.

4.2.1 Integrasi *YOLOv11* dengan Sistem

Pada bagian ini dibahas mengenai proses integrasi model deteksi objek *You Only Look Once* versi 11 (*YOLOv11*) ke dalam sistem yang telah dirancang untuk mendeteksi dan mengidentifikasi komponen alat berat secara waktu nyata (*real-time*). Untuk memberikan gambaran yang jelas mengenai alur integrasi model *You Only Look Once* versi 11 (*YOLOv11*) ke dalam sistem deteksi objek, diperlukan pemetaan visual terhadap komponen-komponen sistem yang saling terhubung. Proses integrasi ini mencakup perangkat keras, perangkat lunak, dan layanan penyimpanan berbasis cloud yang bekerja secara terkoordinasi dalam menjalankan fungsi deteksi objek secara otomatis. Gambar berikut menyajikan alur integrasi

sistem, mulai dari tahap pengambilan citra hingga proses penyimpanan dan visualisasi hasil deteksi.



Gambar 4. 10 Diagram integrasi *YOLOv11* dengan Sistem (Penulis, 2025)

Berdasarkan gambar 4.10, Sistem dimulai dari kamera Pi yang terhubung dengan *Raspberry Pi* sebagai unit pemrosesan utama. Kamera ini berfungsi untuk menangkap citra secara langsung, yang kemudian diproses oleh model *You Only Look Once* versi 11 (*YOLOv11*) yang telah ditanamkan di dalam perangkat *Raspberry Pi*. Proses inferensi ini menghasilkan keluaran berupa bounding box, label objek, serta *confidence score* yang divisualisasikan secara waktu nyata (real-time) melalui layar monitor lokal. Setelah proses deteksi selesai, gambar hasil deteksi disimpan dalam format JPEG dan secara otomatis diunggah ke *Google Drive*. Secara bersamaan, metadata hasil deteksi—yang mencakup nama objek, tingkat akurasi, waktu deteksi, dan lokasi perangkat—dikirimkan dan disimpan ke dalam basis data *Firebase Firestore*. Seluruh data tersebut kemudian dapat diakses dan ditampilkan melalui dashboard berbasis web yang telah terintegrasi dengan *Firestore*, sehingga memungkinkan pengguna untuk memantau dan mengevaluasi hasil deteksi dari jarak jauh. Adapun pengerjaan yang dilakukan pada integrasi *YOLOv11* sebagai berikut.

A. Alur Kerja Sistem

Alur kerja sistem deteksi objek berbasis *YOLOv11* ini terdiri dari tiga tahap utama, yaitu pengambilan gambar (*input*), pemrosesan deteksi objek, dan penyimpanan hasil (*output*), yang seluruhnya diotomatisasi dalam satu rangkaian program berbasis *Python* dan dijalankan di perangkat *Raspberry Pi*. Proses diawali dengan pengambilan citra secara langsung melalui kamera yang terhubung ke *Raspberry Pi,* menggunakan perintah “libcamera-still” yang dieksekusi melalui pustaka subprocess. Citra diambil dalam bentuk buffer mentah (stdout). kemudian

dikonversi menjadi format gambar menggunakan ‘numpy’ dan cv2.imdecode() agar dapat diproses oleh model deteksi.

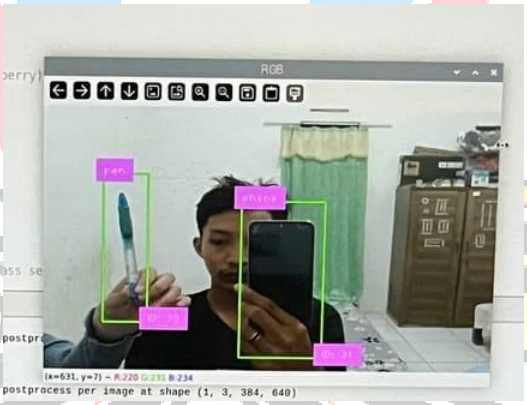
Tahap selanjutnya adalah pemrosesan gambar oleh model *YOLOv11* yang telah dimuat sebelumnya menggunakan pustaka ultralytics. Deteksi objek dilakukan melalui fungsi model.track(frame, persist=True, conf=0.3), yang menghasilkan koordinat *bounding box*, label kelas, dan nilai *confidence* dari setiap objek yang teridentifikasi. Label objek dibaca dari file cocoTA.txt, yang berisi daftar nama-nama objek yang menjadi target dalam sistem ini. Visualisasi hasil deteksi dilakukan dengan menggunakan fungsi cv2.rectangle untuk menggambar kotak di sekitar objek dan cvzone.putTextRect untuk menampilkan label beserta nilai *confidence* di atas citra secara real-time.

Hasil deteksi kemudian disimpan secara lokal dalam format JPEG dengan nama berkas berdasarkan timestamp saat pengambilan gambar, menggunakan fungsi cv2.imwrite(). Selanjutnya, gambar tersebut diunggah ke *Google Drive* melalui integrasi API yang dilakukan dengan pustaka *googleapiclient*. Proses ini dilakukan dengan fungsi upload_to_drive(), yang mengatur metadata file dan mengunggah gambar ke folder yang telah ditentukan. Setelah berhasil diunggah, tautan akses gambar disimpan dan dikaitkan dengan metadata lainnya.

Sebagai tahap akhir, sistem mencatat metadata hasil deteksi ke dalam database *Firebase Firestore* menggunakan pustaka firebase_admin. Metadata yang dikirim meliputi waktu, nama objek, tingkat akurasi, lokasi pengambilan, serta tautan gambar yang tersimpan di *Google Drive*. Proses ini dilakukan dengan memanggil fungsi simpan_metadata_ke_firestore(), yang memastikan semua data penting tersimpan dan dapat diakses melalui website. Dengan alur kerja ini, sistem mampu menjalankan proses deteksi objek secara otomatis sehingga memudahkan pemantauan hasil deteksi.

B. Pengguna Antarmuka

Sistem deteksi objek yang dibangun tidak hanya berfokus pada identifikasi objek, namun juga menyediakan antarmuka yang memungkinkan pengguna untuk memantau dan mengakses hasil deteksi secara langsung maupun jarak jauh. Antarmuka ini dibagi ke dalam tiga lapisan, yaitu antarmuka lokal melalui monitor Raspberry Pi, penyimpanan metadata ke *Firebase Firestore*, serta dashboard berbasis web untuk menampilkan hasil secara komprehensif. Antarmuka lokal pertama dihadirkan melalui layar monitor yang terhubung langsung ke perangkat Raspberry Pi. Visualisasi hasil deteksi ditampilkan secara real-time menggunakan fungsi cv2.imshow("Tracking", frame), yang merupakan bagian dari pustaka OpenCV.



Gambar 4. 11 Visualisasi Hasil Deteksi dengan Kamera Pi (Penulis, 2025)

Berdasarkan gambar 4.11, pengguna dapat melihat secara langsung objek yang terdeteksi pada kamera, lengkap dengan bounding box, label objek, dan nilai confidence yang telah divisualisasikan. Tampilan ini sangat berguna untuk kebutuhan pemantauan langsung di lapangan atau saat sistem sedang diuji. Selain tampilan lokal, sistem juga mendukung penyimpanan hasil deteksi dalam bentuk metadata ke layanan Firebase Firestore. Integrasi Firebase dilakukan melalui pustaka firebase_admin, dan fungsi simpan_metadata_ke_firestore() digunakan untuk mengirim data ke koleksi hasil_deteksi.

Informasi yang disimpan meliputi nama objek yang terdeteksi, tingkat akurasi, waktu deteksi, serta tautan gambar yang telah diunggah ke Google Drive. Dengan pendekatan ini, sistem menyediakan repositori terstruktur yang dapat diakses kembali untuk analisis lebih lanjut atau audit proses deteksi. Lapisan terakhir dari antarmuka pengguna adalah dashboard web yang telah dikembangkan

secara terpisah. Dashboard ini mengakses data dari Firebase Firestore untuk menampilkan informasi hasil deteksi dalam bentuk tabel. Setiap entri pada dashboard disertai dengan link gambar dari Google Drive yang diperoleh dari proses upload_to_drive() dalam program. Dengan demikian, pengguna dapat memverifikasi hasil deteksi dengan membuka gambar pendukung yang telah tersimpan di cloud.

C. Format Model dan Pemangilan

Model deteksi objek yang digunakan dalam sistem ini dikembangkan menggunakan arsitektur *YOLOv11* yang telah dilatih dan disimpan dalam format file best.pt (PyTorch). Format ini membuat model untuk digunakan secara fleksibel di berbagai perangkat berbasis *Python*, termasuk *Raspberry Pi*. Pemanggilan dan eksekusi model dilakukan menggunakan pustaka *Ultralytics*, yang menyediakan antarmuka pemrograman sederhana dan efisien untuk proses deteksi objek, pelacakan, maupun pelatihan ulang. Dalam implementasi sistem, model dipanggil satu kali di awal program menggunakan perintah *YOLO*("yolov11.pt"), dan hasil objek yang terdeteksi diperoleh melalui fungsi model.track. Fungsi ini digunakan untuk melakukan pelacakan objek pada setiap frame gambar yang ditangkap kamera, dengan mempertahankan identitas objek antar frame. Berikut tabel dibawah menjelaskan rincian pemanggilan model dan penggunaannya dalam program.

Tabel 4. 3 Format Model dan Pemanggilan Program

Komponen	Keterangan
Format Model	best.pt (PyTorch model)
Pustaka yang Digunakan	ultralytics
Pemanggilan Model	model = YOLO("yolov11.pt")
Fungsi yang Digunakan	model.track(frame, persist=True, conf=0.3) – digunakan untuk pelacakan objek secara real-time
Output dari Fungsi	Bounding box, label objek, confidence, dan ID tracking untuk setiap objek

Komponen	Keterangan
Posisi dalam Program	Pemanggilan model dilakukan di bagian awal script utama sebelum loop pendeteksian

Sumber: Penulis,2025

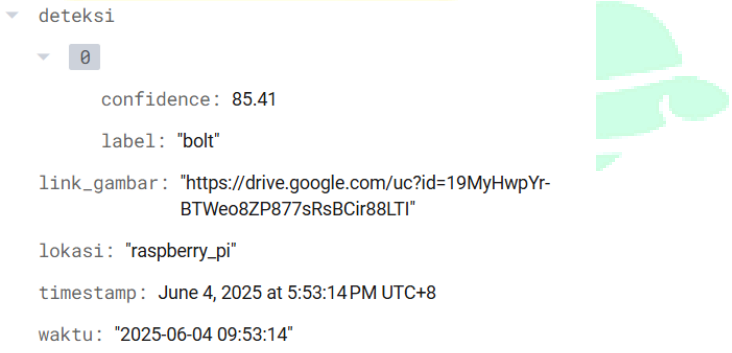
```
# === Inisialisasi Model YOLOv11 ===
model = YOLO("yolov11.pt")
class_names = open("cocoTA.txt").read().strip().split("\n")  # Label
```

Gambar 4. 12 Kode pemanggilan model *YOLOv11* (Penulis, 2025)

Berdasarkan gambar 4.12, dan table 4.2, model *You Only Look Once* versi 11 (*YOLOv11*) dipanggil menggunakan pustaka ultralytics pada bahasa pemrograman Python. Pemanggilan dilakukan satu kali di awal program menggunakan perintah model = YOLO("yolov11.pt"). Model yang telah dimuat ke dalam memori akan digunakan secara berulang untuk melakukan inferensi terhadap setiap frame yang diterima dari kamera, tanpa perlu dimuat ulang. Berdasarkan tabel tersebut, model hanya dimuat sekali di awal dan digunakan terus-menerus selama program berjalan. Pendekatan ini memungkinkan efisiensi penggunaan sumber daya dan mendukung performa deteksi secara waktu nyata (real-time).

D. Integrasi *Firebase*

Pada integrasi *firebase* dilakukan untuk mendukung pencatatan, pemantauan hasil deteksi objek secara terstruktur dan berbasis cloud. Sistem ini diintegrasikan dengan *Firebase Firestore* sebagai basis data. Integrasi ini dilakukan menggunakan pustaka firebase_admin yang disediakan secara resmi oleh *Google* untuk *Python*. Inisialisasi Firebase dilakukan di awal program untuk mengautentikasi aplikasi dengan kredensial proyek dan menetapkan koneksi ke *Firestore*.



Gambar 4. 13 Hasil Pengiriman Metadata ke *Firestore* (Penulis, 2025)

Berdasarkan gambar 4.13, setiap sistem berhasil mendeteksi objek dan menyimpan gambar ke *Google Drive*, sistem akan mencatat metadata hasil deteksi ke dalam *Firestore*. Metadata ini meliputi waktu deteksi, nama objek terdeteksi, tingkat akurasi (*confidence*), link gambar dari *Google Drive*, informasi lokasi perangkat "raspberry_pi" untuk mendukung pengelompokan, dan identifikasi sumber data. Berikut tabel program proses integrasi *firebase*.

Tabel 4. 4 Program Proses Integrasi *Firebase*

Komponen	**Keterangan**
Pustaka yang Digunakan	firebase_admin
Inisialisasi Firebase	Cred = credentials.Certificate("serviceAccountKey.json") firebase_admin.initialize_app(cred)
Database yang Digunakan	firestore.client() → Mengakses Firestore
Nama Koleksi	"hasil_deteksi"
Fungsi Pengiriman Metadata	simpan_metadata_ke_firestore()
Field yang Disimpan	- timestamp (waktu deteksi) - label (nama objek) - confidence - link_gambar - lokasi
Contoh Data yang Dikirim	{ "label": "bucket", "confidence": 0.92, "timestamp": "2025-05-30 14:20", "link_gambar": "...", "lokasi": "raspberry_pi" }

Sumber: Penulis,2025

Berdasarkan tabel 4.3, Proses pencatatan metadata ke *Firebase Firestore* dilakukan secara otomatis setiap kali sistem menyelesaikan satu siklus deteksi objek dan unggah gambar ke *Google Drive*. Dengan metode ini, sistem tidak hanya menyimpan data secara lokal, tetapi juga menyediakan *repository* berbasis *cloud* yang dapat diakses kapan saja dan dari mana saja. Hal ini sangat berguna dalam konteks pemantauan jarak jauh, analisis data historis, serta dokumentasi hasil kerja sistem secara sistematis. Integrasi dengan *Firestore* juga memungkinkan skalabilitas dan fleksibilitas sistem untuk dikembangkan lebih lanjut, seperti menambahkan fitur klasifikasi berdasarkan waktu, jenis objek, atau lokasi deteksi.

E. Integrasi *Google Drive*

Integrasi *Google Drive* dalam sistem ini berfungsi sebagai tempat penyimpanan gambar hasil deteksi objek. Setiap gambar yang berhasil diproses oleh model *YOLO* akan disimpan ke dalam folder tertentu di akun *Google Drive* yang telah dikonfigurasi. Proses ini menggunakan pustaka *googleapiclient* dengan pendekatan MediaFileUpload untuk mengunggah file secara langsung dari *Raspberry Pi*.

Tabel 4. 5 Program Integrasi *Google Drive*

Langkah	**Potongan Program**
Autentikasi & Setup API	service = build('drive', 'v3', credentials=creds)
Menyiapkan metadata file	file_metadata = {'name': filename, 'parents': [FOLDER_ID]}
Mengunggah gambar ke Drive	media = MediaFileUpload(local_path, mimetype='image/jpeg')file = service.files().create(...).execute()
Mengatur izin berbagi & mengambil link	service.permissions().create(...).execute()link = "https://drive.google.com/uc?id=" + file.get("id")

Sumber: Penulis, 2025

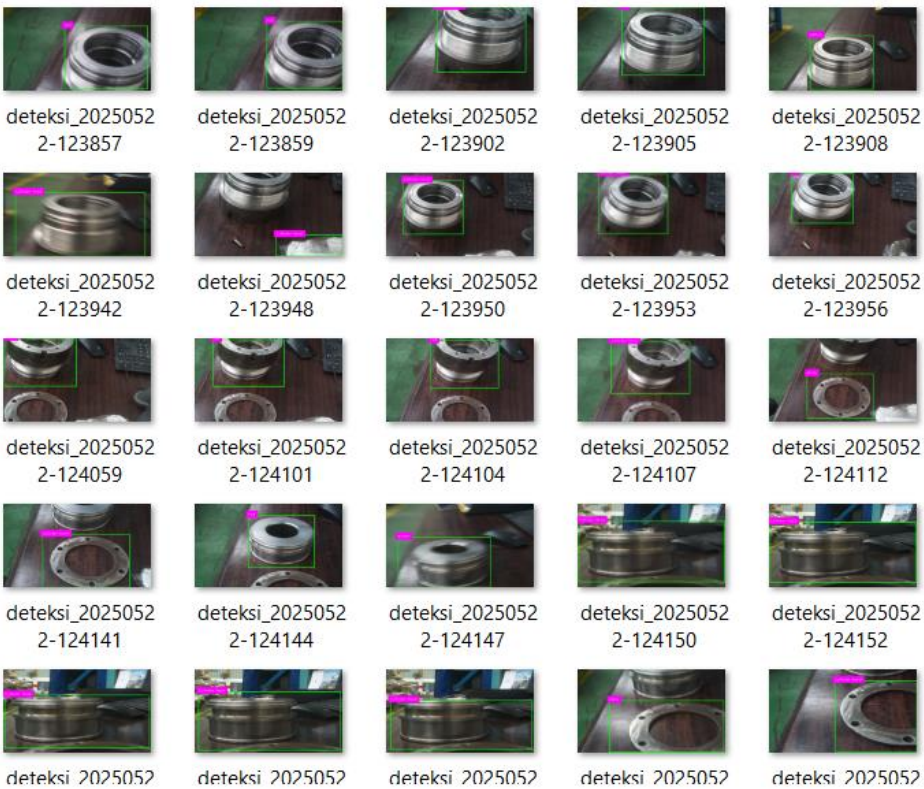
Berdasarkan tabel 4.4, setelah gambar hasil deteksi disimpan secara lokal, sistem memulai proses upload ke Google Drive menggunakan fungsi MediaFileUpload. Metadata file disiapkan untuk menetapkan nama file dan direktori tujuan di dalam akun Drive. Proses unggah ini dilakukan ke folder yang telah ditentukan melalui ID folder (FOLDER_ID), sehingga pengelolaan file tetap terorganisir. Selanjutnya, sistem menetapkan izin akses publik untuk file yang diunggah agar dapat diakses oleh dashboard web. Link berbagi yang diperoleh dari file tersebut kemudian disimpan sebagai bagian dari metadata yang dikirim ke Firebase Firestore.

4.2.2 Pengujian Sistem Deteksi Tanpa Nilai Akurasi

Pengujian ini bertujuan untuk mengevaluasi kinerja sistem dalam mendeteksi dan mengidentifikasi objek berdasarkan label yang ditampilkan pada hasil visualisasi deteksi, tanpa menggunakan perhitungan metrik akurasi secara langsung dari model. Pendekatan yang digunakan berfokus pada tingkat kesesuaian antara label objek yang terdeteksi oleh sistem dengan objek yang sebenarnya

terdapat dalam citra uji. Proses pengujian dilakukan dengan memberikan sejumlah citra yang memuat komponen alat berat kepada sistem. Selanjutnya, hasil deteksi diamati melalui visualisasi yang mencakup *bounding box* serta label objek pada masing-masing citra.

Evaluasi dilakukan dengan mencatat jumlah deteksi yang sesuai dan tidak sesuai berdasarkan objek sebenarnya yang terdapat dalam citra uji. Berdasarkan hasil tersebut, nilai akurasi akan dihitung secara manual dengan membandingkan jumlah deteksi yang benar terhadap total jumlah citra uji. Selain itu, dilakukan analisis terhadap kemungkinan penyebab kesalahan deteksi, seperti kondisi pencahayaan yang kurang optimal, sudut pengambilan gambar, serta kemiripan bentuk antar objek yang dapat memengaruhi kinerja model. Berikut gambar hasil deteksi objek komponen alat berat tanpa nilai akurasi.



Gambar 4. 14 Hasil Deteksi Tanpa Nilai Akurasi (Penulis, 2025)

Berdasarkan gambar 4.10, Pengujian sistem dilakukan menggunakan citra objek yang tersedia di lokasi pengambilan data, yang terdiri dari empat jenis komponen alat berat, yaitu *bolt*, *cylinder head*, *piston*, dan *plate*. Pemilihan keempat objek ini didasarkan pada ketersediaannya secara fisik di lokasi serta

dinilai cukup representatif untuk menguji kemampuan dasar dari model deteksi yang digunakan. Hasil visualisasi deteksi ditampilkan dalam bentuk gambar yang menunjukkan objek-objek yang teridentifikasi oleh sistem. Setiap objek ditandai dengan *bounding box*, label nama objek, serta nilai tingkat kepercayaan (*confidence score*), yang dihasilkan melalui proses pemrosesan oleh model *YOLOv11*.

Namun, hasil deteksi menunjukkan bahwa tidak seluruh label yang ditampilkan oleh sistem sesuai dengan objek sebenarnya yang terdapat pada citra uji. Selain label yang sesuai, seperti *piston*, *cylinder head*, *plate*, dan *bolt*, sistem juga menghasilkan label yang tidak relevan, antara lain *cylinder barrel*, *rod*, *steering cylinder*, dan *phone*. Kemunculan label-label yang tidak sesuai ini mengindikasikan adanya kesalahan klasifikasi (*misclassification*) atau pendeteksian berlebih (*over-detection*) oleh model, meskipun objek-objek tersebut sejatinya tidak ada di dalam citra. Berikut merupakan tabel hasil deteksi terhadap komponen yang terdeteksi.

Tabel 4. 6 Label Deteksi Objek

No	Label yang Terdeteksi	Jumlah Terdeteksi	Termasuk Objek Uji	Keterangan
1.	Cylinder Head	65	Ya	Deteksi sesuai
2.	Piston	51	Ya	Deteksi sesuai
3.	Plate	49	Ya	Deteksi sesuai
4.	Bolt	8	Ya	Deteksi sesuai
5.	Cylinder Barrel	48	Tidak	Salah deteksi / label mirip
6.	Rod	50	Tidak	Salah deteksi / label mirip
7.	Steering Cylinder	2	Tidak	Salah deteksi / noise
8.	Phone	14	Tidak	Salah deteksi / noise ekstrim
Total Gambar			**340**	

Sumber: Penulis, 2025

Berdasarkan tabel 4.5, diketahui bahwa dari delapan label objek yang terdeteksi oleh sistem, hanya empat label yang sesuai dengan objek uji yang sebenarnya, yaitu *Cylinder Head*, *Piston*, *Plate*, dan *Bolt*. Keempat label tersebut menunjukkan hasil deteksi yang akurat, karena objek-objek tersebut memang terdapat secara fisik dalam citra uji dan sesuai dengan hasil visualisasi yang ditampilkan oleh sistem. Kemudian terdapat empat label lain yang tidak termasuk dalam objek uji, yaitu *Cylinder Barrel*, *Rod*, *Steering Cylinder*, dan *Phone*. Deteksi terhadap keempat label ini dikategorikan sebagai kesalahan deteksi (*false positive*), karena objek-objek tersebut tidak ada dalam gambar uji.

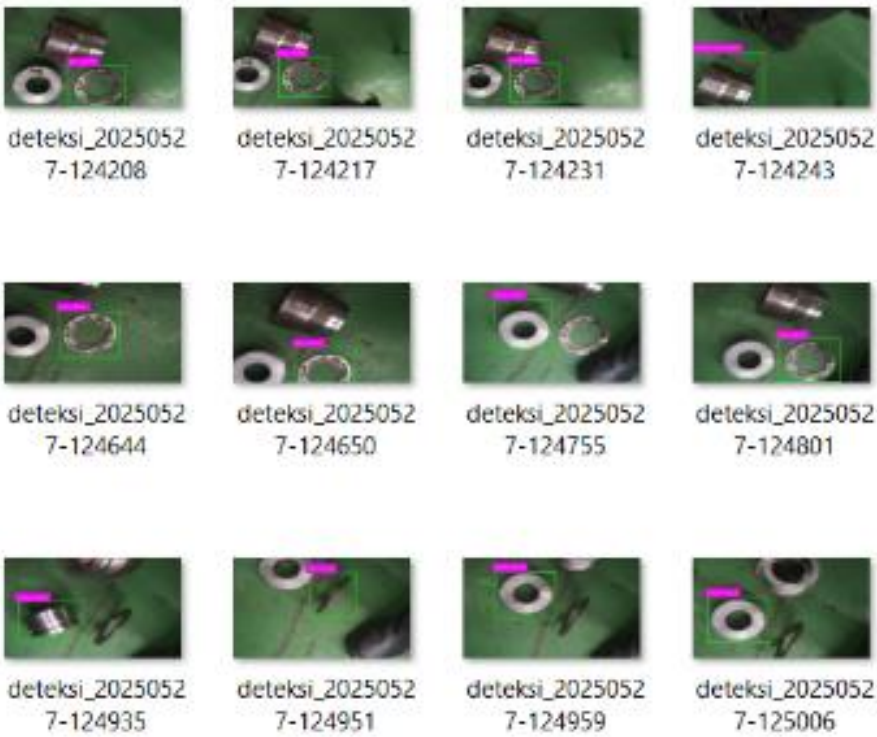
Gambar 4. 15 Hasil Kesalahan Deteksi objek (Penulis, 2025)

Berdasarkan gambar 4.11, kesalahan ini dapat disebabkan oleh beberapa faktor, antara lain kemiripan visual antar objek, sudut pengambilan gambar yang kurang tepat, kondisi pencahayaan yang tidak merata, serta keterbatasan model dalam membedakan ciri-ciri visual spesifik dari masing-masing objek. Keberadaan deteksi yang tidak sesuai ini menunjukkan bahwa meskipun model *YOLOv11* memiliki kemampuan yang baik dalam mengidentifikasi objek, masih terdapat kemungkinan terjadinya klasifikasi yang tidak akurat. Oleh karena itu, evaluasi secara manual tetap diperlukan untuk memverifikasi keakuratan hasil deteksi, khususnya dalam penerapan nyata di lingkungan lapangan.

4.2.3 Pengujian Sistem Deteksi Dengan Label dan Akurasi

Pengujian ini dilakukan untuk mengevaluasi kinerja sistem dalam mendeteksi dan mengklasifikasikan objek berdasarkan label yang dihasilkan oleh model deteksi. Kemudian menggunakan pendekatan kuantitatif melalui perhitungan metrik evaluasi. Berbeda dengan pengujian sebelumnya yang hanya mengandalkan kesesuaian label secara visual. pada pengujian ini digunakan data *ground truth* sebagai acuan untuk membandingkan hasil deteksi sistem secara objektif dan terukur.

Proses pengujian dilakukan dengan memberikan sejumlah citra uji yang memuat komponen alat berat kepada sistem, lalu hasil deteksi dibandingkan dengan label *ground truth* yang telah ditentukan secara manual. Setiap hasil deteksi dievaluasi berdasarkan parameter pengujian seperti *True Positive* (TP), *False Positive* (FP), dan *False Negative* (FN) yang kemudian digunakan untuk menghitung metrik akurasi, *precision*, *recall*, dan *F1-score*. Metrik-metrik ini memberikan gambaran yang lebih menyeluruh mengenai seberapa baik sistem mampu mengenali objek secara konsisten dan tepat.



Gambar 4. 16 Hasil Deteksi dengan Label dan Akurasi (Penulis, 2025)

Berdasarkan gambar 4.12, sistem berhasil mendeteksi sejumlah objek dari citra uji menggunakan model *YOLOv11*. Visualisasi deteksi menampilkan label objek, nilai tingkat kepercayaan (*confidence score*), serta bounding box yang mengelilingi objek terdeteksi. Objek-objek yang digunakan dalam pengujian ini terbatas pada empat komponen utama, yaitu *bolt*, *cylinder head*, *piston*, dan *plate*. Keempat objek ini merupakan komponen alat berat yang tersedia secara fisik di lokasi pengambilan data dan telah ditentukan sebagai *ground truth* untuk keperluan evaluasi. Namun, hasil deteksi memperlihatkan bahwa model juga mengidentifikasi beberapa label lain di luar objek uji yang sebenarnya, seperti *phone*, *cylinder barrel*,

rod, dan *steering cylinder*. Kemunculan label-label yang tidak sesuai ini menunjukkan adanya potensi kesalahan deteksi atau *false positive*, yang dapat disebabkan oleh kemiripan bentuk antar objek, noise dalam citra, atau keterbatasan model dalam mengenali fitur visual secara spesifik.

Tabel 4. 7 Hasil Deteksi Sistem dengan Label dan Akurasi

No	Label yang Terdeteksi	Rata-rata Confidence	Jumlah Deteksi	Termasuk Objek Uji	Keterangan
1.	Cylinder Head	0.7536	22	Ya	Deteksi sesuai
2.	Piston	0.6517	35	Ya	Deteksi sesuai
3.	Plate	0.7260	36	Ya	Deteksi sesuai
4.	Bolt	0.4172	5	Ya	Deteksi sesuai
5.	Phone	0.5651	13	Tidak	Salah deteksi / noise ekstrim
6.	Cylinder Barrel	0.5395	5	Tidak	Salah deteksi / label mirip
7.	Rod	0.3419	7	Tidak	Salah deteksi / label mirip
8.	Steering Cylinder	0.4602	2	Tidak	Salah deteksi / noise minor

Sumber: Penulis, 2025

Berdasarkan tabel 4.6, dapat dilihat bahwa dari delapan label objek yang terdeteksi oleh sistem, hanya empat label yang benar-benar termasuk dalam objek uji yang digunakan, yaitu *Cylinder Head*, *Piston*, *Plate,* dan *Bolt*. Keempat label ini menunjukkan hasil deteksi yang sesuai, karena objek tersebut memang tersedia secara fisik dalam citra uji. Namun, rata-rata nilai confidence (tingkat keyakinan model terhadap hasil deteksi) dari keempat label tersebut belum mencapai target yang diharapkan, yaitu lebih dari 0.9. Nilai rata-rata *confidence* tertinggi terdapat pada label *Cylinder Head* dengan 0.7536, *Plate* sebesar 0.7260*, Piston* sebesar 0.6517, dan *Bolt* hanya sebesar 0.4172. Hal ini menunjukkan bahwa meskipun deteksi dilakukan terhadap objek yang benar, keyakinan model terhadap prediksi tersebut masih tergolong sedang hingga rendah.

Selain itu, terdapat empat label lain yang tidak termasuk dalam objek uji, yaitu *Phone*, *Cylinder Barrel*, *Rod*, dan *Steering Cylinder*. Label-label ini merupakan hasil dari kesalahan deteksi (*false positive*), di mana sistem mendeteksi keberadaan objek yang sebenarnya tidak ada dalam gambar. Kemunculan label-label ini kemungkinan besar disebabkan oleh kemiripan visual antar objek, kualitas

pencahayaan, sudut pengambilan gambar, atau keterbatasan model dalam membedakan fitur visual yang spesifik antar komponen. Nilai *confidence* dari label-label ini juga cukup bervariasi, mulai dari 0.3419 (*Rod*) hingga 0.5651 (*Phone*), yang menunjukkan bahwa meskipun nilai *confidence* tidak terlalu tinggi, sistem tetap menganggap deteksi tersebut valid.

Secara keseluruhan, rata-rata nilai *confidence* per kelas objek yang terdeteksi belum memenuhi tujuan yang ditetapkan dalam pengujian ini, yaitu (>0.9). Hasil rata-rata akurasi per kelas yang tidak mencapai target ≥0,9 menunjukkan bahwa kinerja model deteksi objek masih belum optimal. Hal ini mengindikasikan perlunya peningkatan dari berbagai aspek, baik dari sisi data pelatihan, parameter model, maupun kondisi lingkungan saat proses akuisisi data. Salah satu penyebab utama rendahnya akurasi adalah kualitas dan kuantitas dataset yang digunakan. Kemungkinan besar dataset belum merepresentasikan variasi yang cukup dari setiap objek, baik dalam hal sudut pandang, pencahayaan, ukuran, maupun latar belakang. Selain itu, ketidakseimbangan jumlah sampel antar kelas juga dapat menyebabkan model memiliki kecenderungan lebih akurat terhadap kelas tertentu dibanding kelas lainnya.

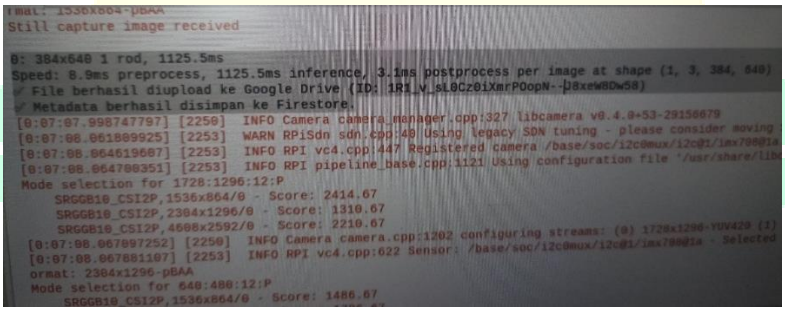
Pada kemiripan visual antar objek seperti antara *cylinder barrel*, *rod*, dan *piston* juga berkontribusi terhadap kesalahan klasifikasi. Fitur visual yang menyerupai membuat model kesulitan membedakan karakteristik unik dari masing-masing kelas, terutama dalam citra dengan pencahayaan atau sudut yang tidak optimal. Hal ini diperparah jika proses pelabelan tidak dilakukan secara konsisten atau akurat. Selain itu, pengaruh parameter dalam pelatihan model, seperti *threshold confidence*, *anchor boxes*, dan jumlah *epoch*, juga sangat menentukan kualitas deteksi yang dihasilkan. Pengaturan threshold yang terlalu rendah, misalnya, dapat memunculkan deteksi dengan tingkat keyakinan rendah yang seharusnya dapat disaring.

Kondisi pengambilan data juga menjadi faktor penting. Gambar yang diambil di lingkungan nyata sering kali memiliki pencahayaan yang tidak merata, bayangan, atau latar belakang yang kompleks, yang dapat mengurangi kemampuan model dalam mengenali objek secara tepat. Oleh karena itu, perlu adanya standardisasi dalam proses pengambilan citra uji atau penggunaan metode

preprocessing, seperti peningkatan kontras dan normalisasi pencahayaan, sebelum dilakukan deteksi. Secara keseluruhan, hasil ini menunjukkan bahwa evaluasi dan validasi manual tetap penting untuk menghindari kesalahan klasifikasi, terutama jika sistem ini akan diterapkan di lapangan yang menuntut presisi tinggi. Untuk meningkatkan performa sistem deteksi, beberapa langkah dapat dilakukan, seperti melatih ulang model dengan dataset yang lebih besar dan representatif, menyesuaikan parameter pelatihan, serta menerapkan preprocessing citra untuk meningkatkan kualitas input. Selain itu, teknik fine-tuning dengan transfer learning dari model yang telah dilatih pada domain serupa juga dapat meningkatkan akurasi.

4.2.4 Analisis Kecepatan Sistem *Frame Rate*

Selain mengukur akurasi model dalam mendeteksi objek, pengujian sistem ini juga mencakup evaluasi terhadap kecepatan proses deteksi yang diukur menggunakan parameter *Frame Per Second* (FPS). FPS merupakan satuan yang menunjukkan jumlah frame atau citra yang dapat diproses oleh sistem dalam waktu satu detik, dan menjadi indikator utama dalam menilai performa sistem deteksi objek secara waktu nyata (real-time). Pengukuran FPS menjadi penting mengingat sistem ini diimplementasikan pada perangkat dengan spesifikasi terbatas, yaitu *Raspberry Pi*, yang digunakan untuk melakukan inferensi model *YOLOv11* secara langsung. Oleh karena itu, analisis terhadap FPS dilakukan untuk mengetahui sejauh mana sistem mampu memproses input citra secara efisien dan cepat, serta untuk menilai apakah sistem layak diterapkan dalam kondisi lapangan yang membutuhkan respons waktu nyata.



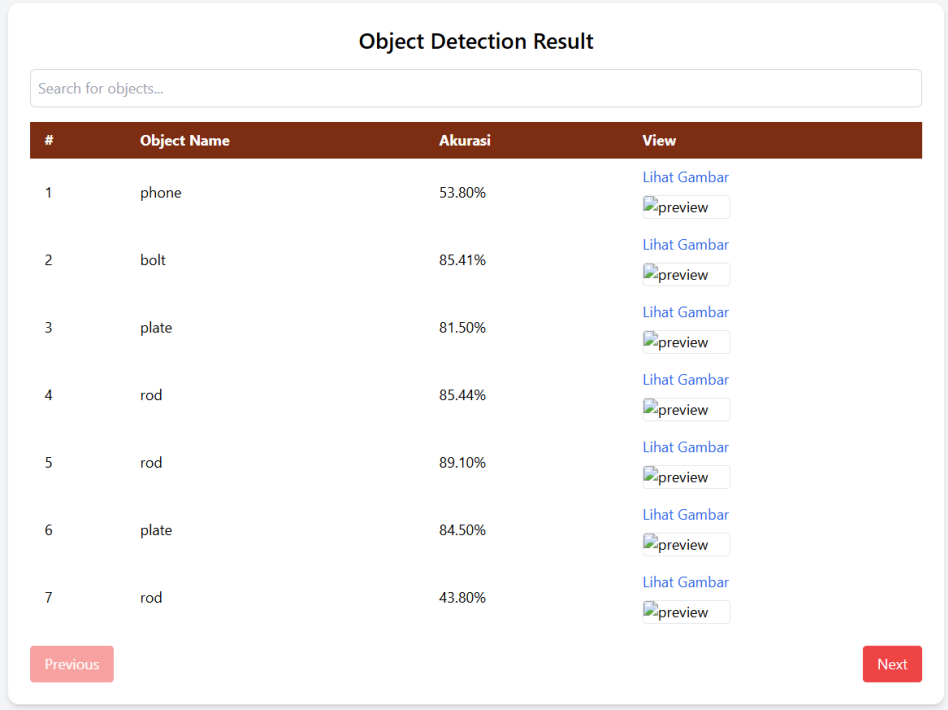
Gambar 4. 17 *Output* Proses Deteksi (Penulis, 2025)

Berdasarkan gambar 4.13, hasil output yang ditampilkan pada terminal, proses pendeteksian objek terdiri dari tiga tahap utama, yaitu preprocessing selama

8,9 milidetik, inference selama 1125,5 milidetik, dan *postprocessing* selama 3,1 milidetik. Jika seluruh waktu pemrosesan ini dijumlahkan, maka total waktu yang dibutuhkan untuk memproses satu gambar adalah sekitar 1137,5 milidetik atau 1,1375 detik. Dengan demikian, kecepatan pemrosesan atau *frame per second* (FPS) dapat dihitung menggunakan rumus FPS = 1 / waktu per frame (dalam detik), yang menghasilkan nilai FPS sebesar kurang lebih 0,88. Ini menunjukkan bahwa sistem hanya mampu memproses kurang dari satu gambar setiap detik, yang berarti performa masih tergolong lambat. Untuk meningkatkan FPS, beberapa langkah optimasi dapat dilakukan, seperti menggunakan model deteksi yang lebih ringan, menurunkan resolusi input gambar, atau meningkatkan performa perangkat keras yang digunakan.

4.3 Visualisasi Hasil pada *Dashboard Website*

Visualisasi hasil deteksi objek dilakukan melalui sebuah dashboard berbasis web yang dirancang untuk menampilkan informasi hasil deteksi secara sistematis dan dapat diakses secara daring. Dashboard ini terintegrasi dengan layanan *Firebase Firestore* sebagai basis data metadata dan *Google Drive* sebagai penyimpanan citra hasil deteksi. Tujuan utama dari pengembangan dashboard ini adalah untuk memudahkan pengguna dalam memantau hasil deteksi secara real-time maupun historis, serta mendukung proses evaluasi dan dokumentasi hasil sistem. Dashboard yang dikembangkan dalam proyek ini berfungsi sebagai antarmuka visual untuk menampilkan hasil deteksi objek yang telah diproses oleh model *YOLOv11*. Tampilan utama dashboard ditampilkan dalam bentuk tabel yang memuat data hasil deteksi, seperti nama objek, nilai akurasi (*confidence*), dan tautan *google drive*. Berikut merupakan tampilan gambar dashboard website untuk visualisasi hasil.

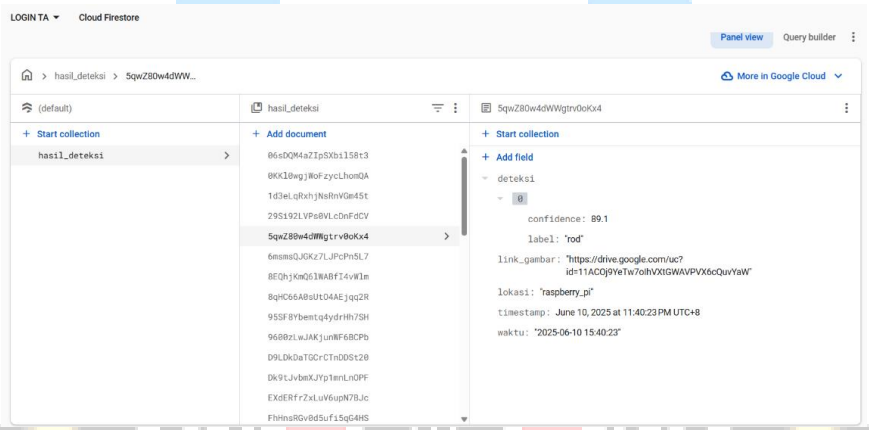


Gambar 4. 18 *Dashboard* Visualisasi Hasil Data (Penulis, 2025)

Berdasarkan gambar 4.14, menampilkan antarmuka dashboard web yang dikembangkan untuk menampilkan hasil deteksi objek secara interaktif. Dashboard ini menyajikan data dalam bentuk tabel yang terdiri atas beberapa kolom, yaitu nomor urut (#), nama objek (*Object Name*), tingkat akurasi (Akurasi), dan kolom *View*. Pada kolom *View*, tersedia dua fitur utama: tautan "Lihat Gambar" yang mengarah langsung ke file gambar hasil deteksi yang disimpan di *Google Drive*, serta pratinjau gambar berupa thumbnail yang memungkinkan pengguna melihat isi gambar secara langsung tanpa perlu membuka tab baru. Tampilan ini dilengkapi dengan fitur pencarian (search bar) di bagian atas tabel untuk memudahkan pencarian berdasarkan nama objek, serta navigasi halaman (*pagination*) di bagian bawah tabel berupa tombol "*Previous*" dan "*Next*" agar pengguna dapat menjelajahi seluruh data yang tersedia. Semua data yang ditampilkan diperoleh secara dinamis dari *Firebase Firestore*, sedangkan gambar-gambar deteksi ditautkan melalui link *Google Drive*. Dengan memanfaatkan kombinasi *HTML, Tailwind CSS*, dan *JavaScript,* dashboard ini dirancang agar responsif, modern, dan mendukung kebutuhan dokumentasi serta pemantauan hasil deteksi secara efisien.

Setelah hasil deteksi objek berhasil divisualisasikan dalam bentuk tabel pada *dashboard web*, data tersebut disimpan secara terstruktur menggunakan

Firebase Firestore sebagai media basis datanya. Penyimpanan ini dilakukan untuk memastikan bahwa setiap informasi hasil deteksi, seperti nama objek, tingkat akurasi, waktu deteksi, dan tautan gambar dapat diakses dan ditampilkan kembali secara efisien. Integrasi ini juga memungkinkan sistem untuk melakukan pembaruan data secara real-time serta mendukung fitur pencarian dan filter data pada antarmuka pengguna. Berikut merupakan tampilan struktur data hasil deteksi yang tersimpan pada Firestore.

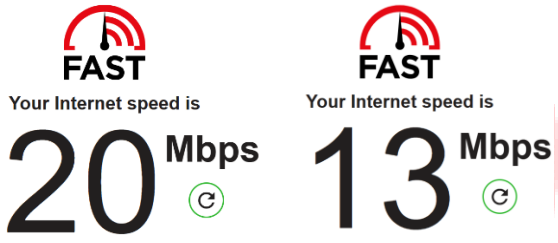


Gambar 4. 19 Metadata Hasil Deteksi pada *Firestore* (Penulis, 2025)

Berdasarkan gambar 4.15, menunjukkan tampilan data yang tersimpan dalam koleksi hasil_deteksi pada *Firebase Firestore*. Setiap data deteksi disimpan dalam bentuk dokumen dengan ID unik, yang berisi array data hasil deteksi objek. Informasi yang dicatat dalam setiap entri mencakup nilai *confidence* yang menunjukkan tingkat keyakinan model terhadap hasil deteksi, dalam hal ini sebesar 89,1%. Label objek yang terdeteksi ditampilkan sebagai "rod", sedangkan link_gambar berisi *URL* yang mengarah ke file gambar hasil deteksi yang telah diunggah ke *Google Drive*. Selain itu, data juga mencakup informasi lokasi sebagai penanda sumber deteksi, yaitu dari perangkat "raspberry_pi", serta waktu deteksi dalam format UTC (*timestamp*) dan waktu lokal (waktu). Penyimpanan data dalam format ini memudahkan proses pelacakan dan pemantauan hasil deteksi, serta memungkinkan integrasi dengan dashboard untuk visualisasi data secara real-time. Penggunaan *Google Drive* sebagai penyimpanan gambar juga memberikan efisiensi dalam pengelolaan file dan mengurangi beban penyimpanan langsung di *Firestore*.

4.3.1 Analisis Kecepatan Jaringan

Untuk memastikan kelancaran proses pengiriman data hasil deteksi ke *Firebase*, dilakukan pula pengujian terhadap kualitas jaringan internet yang digunakan selama pengambilan data. Pengujian ini bertujuan untuk mengetahui seberapa besar pengaruh kecepatan internet terhadap jumlah data yang berhasil dikumpulkan dan diunggah ke server. Adapun hasil pengujian kecepatan jaringan ditunjukkan pada gambar berikut.



Gambar 4. 20 Kecepatan Jaringan (Penulis, 2025)

Berdasarkan gambar 4.16, Kecepatan jaringan internet memiliki pengaruh langsung terhadap kelancaran proses pengambilan dan pengunggahan data dalam sistem yang dirancang. Pada saat pengambilan data pertama, kecepatan internet tercatat sebesar 20 Mbps dan mampu menghasilkan serta mengunggah sebanyak 340 gambar hasil deteksi ke *Firebase*. Sementara itu, pada pengambilan data kedua, kecepatan internet menurun menjadi 13 Mbps, dan jumlah gambar yang berhasil dikumpulkan hanya sebanyak 122 gambar. Perbedaan ini mengindikasikan bahwa semakin tinggi kecepatan internet, maka proses pengunggahan data dapat dilakukan secara lebih cepat dan stabil, sehingga memungkinkan sistem untuk mengolah lebih banyak data dalam waktu yang sama. Sebaliknya, kecepatan internet yang lebih rendah berpotensi menyebabkan keterlambatan unggah data, bahkan kemungkinan terjadinya jeda atau kegagalan koneksi yang berdampak pada berkurangnya jumlah data yang berhasil disimpan secara real-time. Oleh karena itu, kestabilan dan kecepatan jaringan merupakan aspek penting yang perlu diperhatikan dalam implementasi sistem deteksi berbasis internet seperti pada penelitian ini.

Halaman ini sengaja dikosongkan

BAB V

KESIMPULAN

5.1 Kesimpulan

Dari hasil penelitian dan implementasi sistem yang telah dilakukan, penulis menyimpulkan beberapa hal sebagai berikut:

1. Dataset yang digunakan terdiri dari 950 gambar berbagai komponen alat berat yang telah melalui proses anotasi dan augmentasi melalui *Roboflow*, serta pelatihan model di platform *Google Colab*. Hasil pelatihan menunjukkan performa yang cukup baik dengan nilai *mean Average Precision (mAP)* sebesar 93,2% pada IoU 0,5, dan *mAP50-95* sebesar 64%. Hal ini menunjukkan bahwa model mampu mengenali objek dengan baik berdasarkan data pelatihan.
2. Hasil pengujian sistem deteksi dilakukan dalam dua pendekatan, yaitu pengujian tanpa label dan pengujian dengan label serta perhitungan akurasi. Pada pengujian tanpa label, sistem mampu mendeteksi objek yang benar, namun masih ditemukan label yang tidak relevan (*false positive*). Sementara itu, pada pengujian dengan label dan akurasi, rata-rata *confidence* setiap kelas tidak memenuhi target ≥0.9, dengan nilai tertinggi sebesar 0.75 dan terendah 0.41.
3. Faktor-faktor yang menyebabkan ketidaksesuaian hasil deteksi antara lain adalah keterbatasan variasi dataset, kemiripan visual antar objek, kondisi pencahayaan, sudut pengambilan gambar, serta parameter model yang belum optimal.
4. Sistem deteksi objek yang dijalankan pada *Raspberry Pi* hanya mampu menghasilkan kecepatan sekitar 0,88 FPS, yang berarti belum memenuhi kriteria untuk aplikasi waktu nyata *(real-time)*. Oleh karena itu, diperlukan upaya optimasi baik dari sisi model, resolusi input, maupun spesifikasi perangkat keras agar sistem dapat berjalan lebih efisien dan responsif di lapangan.

5.2 Saran

Berdasarkan hasil implementasi dan evaluasi sistem yang telah dilakukan, penulis memberikan beberapa saran sebagai upaya pengembangan lebih lanjut agar sistem dapat bekerja secara lebih optimal dan andal, yaitu:

1. Peningkatan Kualitas dan Variasi Dataset

   Untuk meningkatkan akurasi dan keandalan model, disarankan agar dataset dilengkapi dengan variasi sudut pandang, intensitas pencahayaan, ukuran objek, serta latar belakang yang beragam. Jumlah sampel juga perlu ditingkatkan dan disesuaikan secara proporsional untuk setiap kelas objek guna menghindari bias selama proses pelatihan.

2. Optimasi Parameter Model Deteksi

   Pengaturan parameter model seperti *confidence threshold*, ukuran *anchor box*, jumlah epoch pelatihan, dan skema augmentasi perlu diuji secara sistematis guna mendapatkan konfigurasi terbaik yang sesuai dengan karakteristik objek yang diamati. Hal ini penting agar model dapat memberikan hasil deteksi dengan confidence yang lebih tinggi dan lebih stabil.

3. Stabilisasi Kinerja Kamera dan Servo

   Selama implementasi, sistem menunjukkan tampilan kamera yang kurang lancar (*lag*) serta gerakan servo yang tidak stabil. Oleh karena itu, diperlukan optimasi pada sisi perangkat keras dan perangkat lunak, misalnya dengan mengurangi resolusi citra agar lebih ringan diproses, mengatur interval pengambilan gambar, serta menerapkan kontrol gerak yang lebih halus untuk servo guna menghindari gerakan berlebihan atau getaran yang tidak diinginkan.

DAFTAR PUSTAKA

Abby, M., Syah, R., Studi, P., & Elektro, T. (2023). *Analisis Performa Yolo, Cnn, Dan Efficientnet Pada Masked-Face Recognition*.

Adi Surya Nugraha. (2023). *Tugas Akhir Perancangan Sistem Kendali Motor Servo Finger Tracking Dengan Metode Cnn Disusun Dalam Memenuhi Syarat Guna Memperoleh Gelar Sarjana Teknik (S1) F I N A L*.

Afni, M. S. N., Septiani, M., Afni, N., & Andharsaputri, R. L. (2019). *Jusim (Jurnal Sistem Informasi Musirawas) Perancangan Sistem Informasi Penyewaan Alat Berat*.

Andry Andaru. (2018). *Osf Preprints _ Andry Andaru 155100006*.

Anggarkusuma, R., Alwiah Musdar, I., Studi Teknik Informatika, P., Kharisma Makassar, S., Studi Sistem Informasi, P., & Alauddin Makassar, U. (2021). *Klasifikasi Citra Komponen Sepeda Motor Menggunakan Algoritma Cnn Dengan Arsitektur Mobilenet*. Https://Jurnal.Kharisma.Ac.Id/Kharismatech

Arif Hudaya, M., Santoso, I., Yosua, D., & Soetrisno, A. A. (2020). Perancangan Sistem Pelacakan (Tracking) Dan Perhitungan Kendaraan Pada Citra Bergerak Menggunakan Metode Convolutional Neural Network. In *Transient* (Vol. 9, Issue 1). Https://Ejournal3.Undip.Ac.Id/Index.Php/Transient

Azis, A. (2021). *Identifikasi Jenis Ikan Menggunakan Model Hybrid Deep Learning Dan Algoritma Klasifikasi*. Http://Groups.Inf.Ed.Ac.Uk/F4k/Groundtruth/

Basrah Pulungan, A., Nafis, Z., Anwar, M., Elvanny Myori, D., & Padang, N. (2021). Object Detection With A Webcam Using The Python Programming Language. In *Journal Of Applied Engineering And Technological Science* (Vol. 2, Issue 2).

Bastian, L., Matas, J., & Sebe, N. (2016). *Bastian Leibe Jiri Matas Nicu Sebe Max Welling (Eds.) Computer Vision-Eccv 2016*. Http://Www.Springer.Com/Series/7412

Cheng, S., Han, Y., Wang, Z., Liu, S., Yang, B., & Li, J. (2025). An Underwater Object Recognition System Based On Improved Yolov11. *Electronics (Switzerland)*, *14*(1). Https://Doi.Org/10.3390/Electronics14010201

Dwi Wijaya, Y., & Wardah Astuti, M. (2019). *Sistem Informasi Penjualan Tiket Wisata Berbasis Web Menggunakan Metode Waterfall*. Http://Www.Php.Net.

Huang, J., Wang, K., Hou, Y., & Wang, J. (2025). Lw-Yolo11: A Lightweight Arbitrary-Oriented Ship Detection Method Based On Improved Yolo11. *Sensors*, *25*(1). Https://Doi.Org/10.3390/S25010065

Husaini, M., Raharja, A. R., Putra, V. H. C., & Lukmana, H. H. (2025). *Enhanced Plant Disease Detection Using Computer Vision YOLOv11: Pre- Trained Neural Network*

Model Application. Journal of Computer Networks, Architecture and High Performance Computing, *7*(1), 66–81. https://doi.org/10.47709/cnahpc.v7i1.5113

Lase, E. V., Syaifuddin, M., & Ginting, E. F. (2020). Sistem Pakar Mendeteksi Kerusakan Komponen Alat Berat Excavator Dengan Menggunakan Metode Teorema Bayes. *Jurnal Cybertech*, *3*(6), 1093–1105. Https://Ojs.Trigunadharma.Ac.Id/

Leibe, B., Matas, J., Sebe, N., & Welling, M. (Eds.). (2016). *Computer Vision – Eccv 2016* (Vol. 9905). Springer International Publishing. Https://Doi.Org/10.1007/978-3-319-46448-0

Li, Y., Yan, H., Li, D., & Wang, H. (2024). Robust Miner Detection In Challenging Underground Environments: An Improved Yolov11 Approach. *Applied Sciences (Switzerland)*, *14*(24). Https://Doi.Org/10.3390/App142411700

M Tumengkol, A. M., Kunci, K., -Alat Berat, A., & Konstruksi, Peralatan. (2021). *Jurnal Jenis Dan Fungsi Alat Berat Gambar Penggunaan Alat Dalam Proyek Konstruksi Bendungan Dan Tambang*.

Mailoa, R. M., & Santoso, L. W. (2022). *Deteksi Rompi Dan Helm Keselamatan Menggunakan Metode Yolo Dan Cnn*.

Mubin, R. A. (2021). *Implementasi Deep Learning (Convolutional Neural Network) Untuk Mendeteksi Pemakaian Masker Secara Real-Time Video Stream*.

Nazilly, M. L., Rahmat, B., & Puspaningrum, E. Y. (2020). Implementasi Algoritma Yolo (You Only Look Once) Untuk Deteksi Api 1. In *Jurnal Informatika Dan Sistem Informasi (Jifosi)* (Vol. 1, Issue 1).

Prabowo, D. A., Abdullah, D., & Manik, A. (2018). Deteksi Dan Perhitungan Objek Berdasarkan Warna Menggunakan Color Object Tracking. In *Jurnal Pseudocode* (Issue 2). Www.Ejournal.Unib.Ac.Id/Index.Php/Pseudocode

Putra, C. M., Triayudi, A., & Ningsih, S. (2023). Face Mask Recognition Menggunakan Model Cnn (Convolutional Neural Network) Berbasis Python Dan Opencv. *Journal Of Computer System And Informatics (Josyc)*, *4*(3), 722–730. Https://Doi.Org/10.47065/Josyc.V4i3.3532

Rawat, W., & Wang, Z. (2017). Deep Convolutional Neural Networks For Image Classification: A Comprehensive Review. In *Neural Computation* (Vol. 29, Issue 9, Pp. 2352–2449). Mit Press Journals. Https://Doi.Org/10.1162/Neco_A_00990

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. *Proceedings Of The Ieee Computer Society Conference On Computer Vision And Pattern Recognition*, *2016-December*, 779–788. Https://Doi.Org/10.1109/Cvpr.2016.91

Ren, S., He, K., Girshick, R., & Sun, J. (2015). *Faster R-Cnn: Towards Real-Time Object Detection With Region Proposal Networks*. Http://Arxiv.Org/Abs/1506.01497

Silalahi, M., & Wahyudi, D. (2018). Computer Based Information System Journal Perbandingan Performansi Database Mongodb Dan Mysql Dalam Aplikasi File Multimedia Berbasis Web. *Cbis Journal*, *06*(01). Http://Ejournal.Upbatam.Ac.Id/Index.Php/Cbis\Http://Ejournal.Upbatam.Ac.Id/Index.Php/Cbis

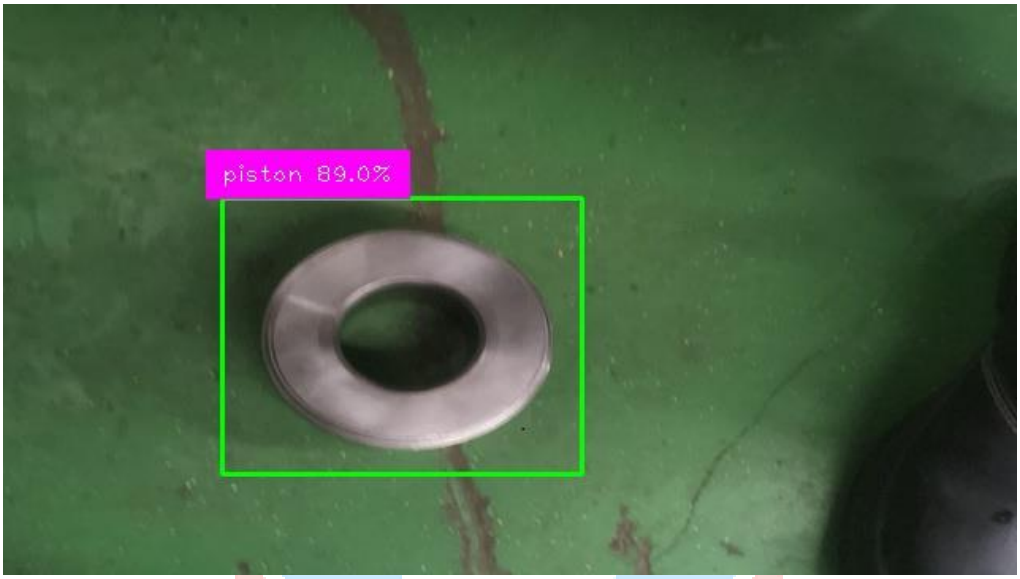
Sonita, A. , & F. R. F. (2018). 5886-Article Text-8498-11552-10-20181123. *Cbis Journal*. Https://Doi.Org/10.33369/Pseudocode.5.2.38-45

Wardoyo, S., Wiryadinata, R., & Sagita, R. (2014). *Sistem Presensi Berbasis Agoritma Eigenfce Dengan Metode Principal Component Analysis*. *3*(1).

Halaman ini sengaja dikosongkan

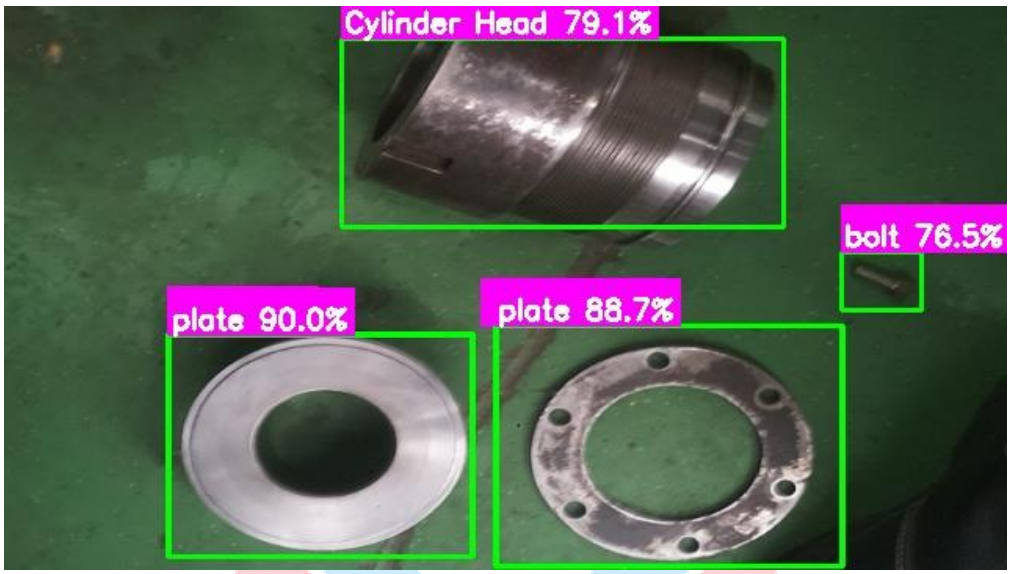
LAMPIRAN

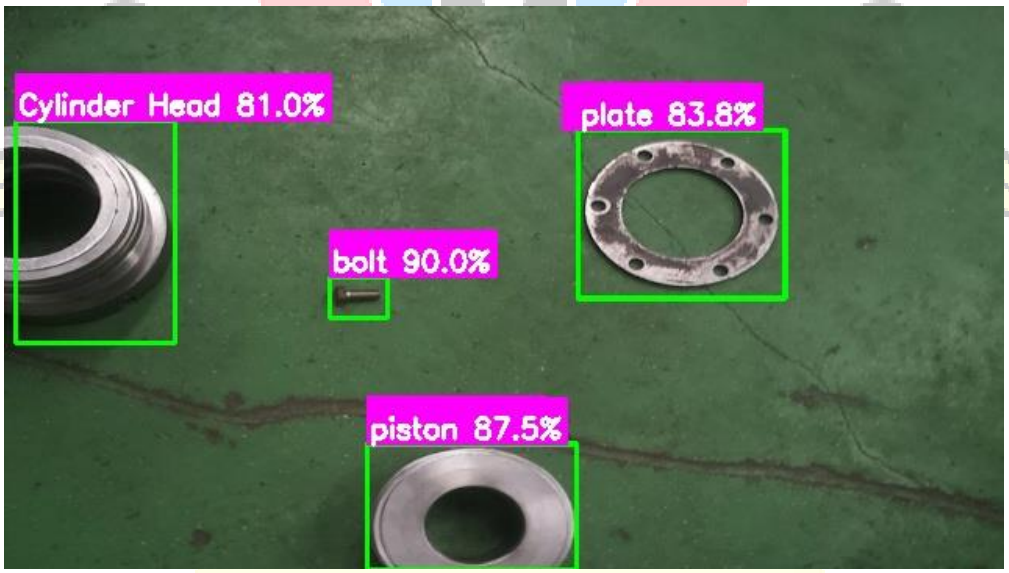
Gambar objek per kelas yang terdeteksi:


piston 89.0%

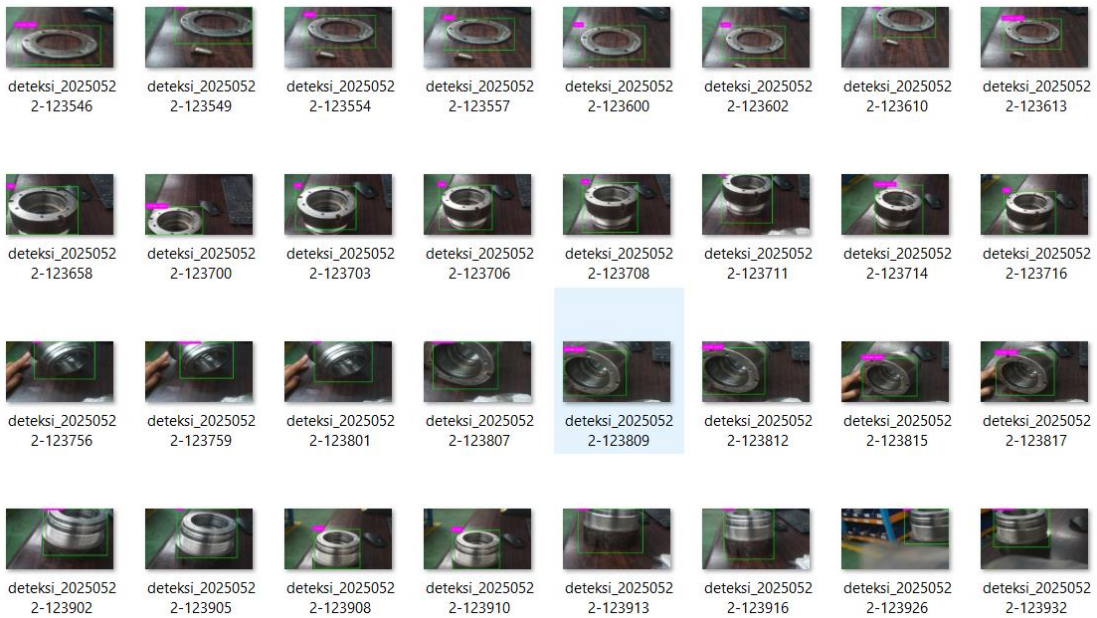

bolt 38.2%

Gambar Multi objek yang terdeteksi :

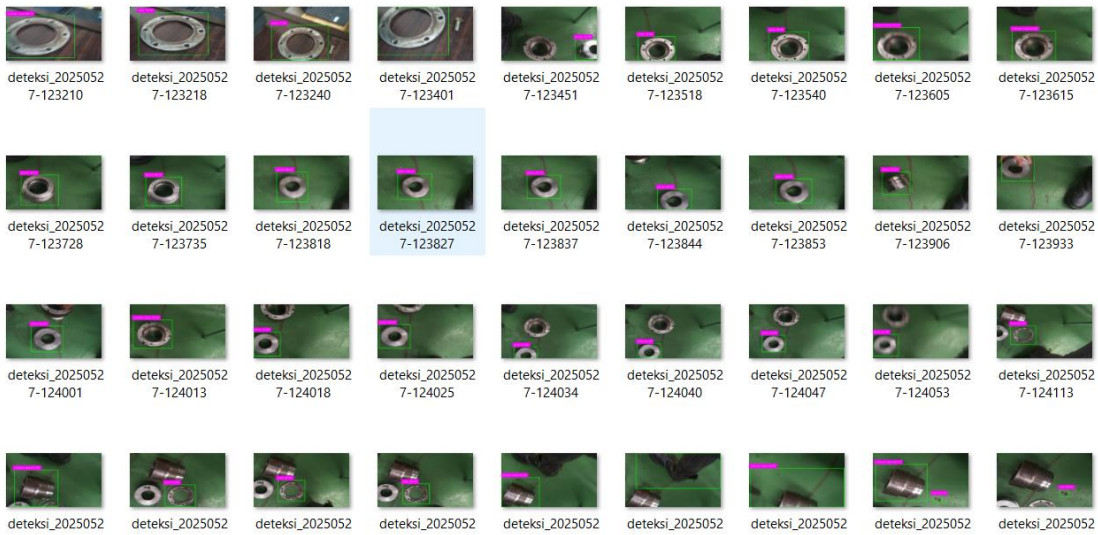




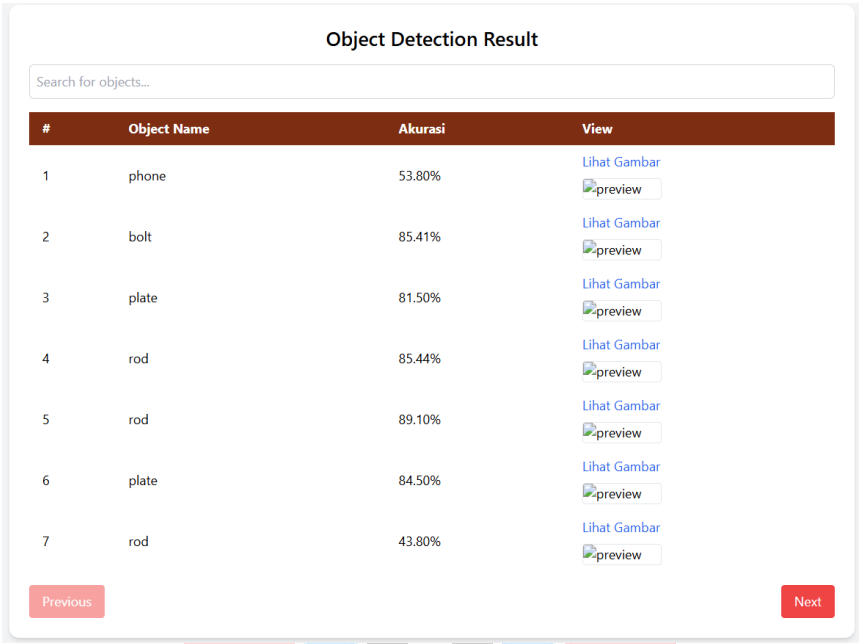
Gambar seluruh objek terdeteksi tanpa nilai akurasi:



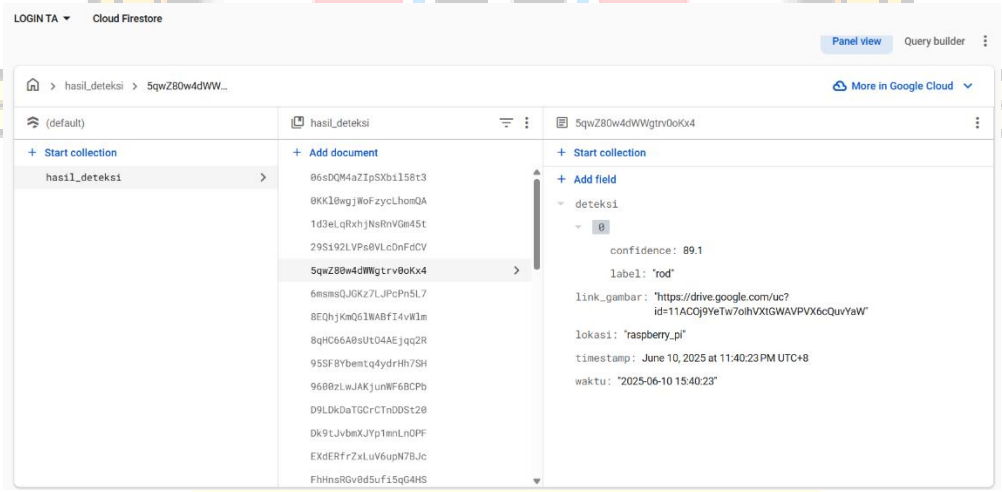
Gambar seluruh objek terdeteksi dengan label dan akurasi



Visualisasi Website Hasil Deteksi:



Penyimpanan Metadata pada *Firestore*:



Halaman ini sengaja dikosongkan

BIOGRAFI PENULIS

Penulis tugas akhir berjudul "PERANCANGAN SISTEM OBJECT DETECTION PADA INDUSTRI ALAT BERAT MENGGUNAKAN DEEP LEARNING" bernama Dikky Kurniawan yang akrab dipanggil masdik. Penulis merupakan anak sulung dari Bapak Edy Pramono dan Ibu Yulianti. Penulis lahir di Balikpapan, 12 juli 2003. Penulis memulai Pendidikan di SDN 024 Balikpapan Utara pada tahun 2009 - 2015. Kemudian, penulis melanjutkan Pendidikan di SMPN 22 Balikpapan Tengah pada tahun 2016 - 2018. Penulis melanjutkan Pendidikan di SMKN 1 Balikpapan Selatan dan lulus pada tahun 2021.

Penulis melanjutkan studi sebagai mahasiswa di Institut Teknologi Kalimantan, Program Studi Teknik Elektro pada tahun 2021. Selama masa studi di Institut Teknologi Kalimantan, penulis aktif mengikuti berbagai kegiatan pengembangan diri dan organisasi. Pada semester 3, penulis mengikuti program Pertukaran Mahasiswa Merdeka di Universitas Singaperbangsa Karawang, Jawa Barat. Selain itu, penulis terlibat dalam berbagai kepanitiaan, antara lain sebagai staf Departemen Advokasi dan Kesejahteraan Mahasiswa (Adkesma) di Himpunan Mahasiswa Teknik Elektro pada semester 4, serta sebagai staf Perlengkapan dan Keamanan (Perkamjin) dalam kegiatan SPIN Etam 2022. Pada semester 5, penulis dipercaya untuk menjabat sebagai Ketua Dewan Perwakilan Mahasiswa Teknik Elektro. Kemudian penulis pernah menjadi asisten kelas matematika Teknik I pada semester 4 dan matematika diskrit pada semester 5. Selain aktif di lingkungan perkuliahan, penulis juga aktif dalam perlombaan yang ada diluar antara lain mengikuti perlombaan Essay yang diadakan oleh Universitas Halu Oleo sebagai finalis dan Teknologi Tepat Guna yang diadakan oleh Universitas Riau.

Halaman ini sengaja dikosongkan

Biodata Penulis

Dikky Kurniawan

Balikpapan, 12 Juli 2003

Jalan Padat Karya RT. 05 No. 65 Balikpapan

087812144514

04211020@student.itk.ac.id

dikkykurniawan95@gmail.com