

PERANCANGAN PURWARUPA ALAT DETEKSI KERUSAKAN JALAN BERBASIS *MACHINE LEARNING*

I Wayan Okta Arianta¹, Steven Tulus Parulian Elluya Sitompul², Gede Candra
Maha Dharmawan³, Ngurah Indra ER⁴, Lie Jasa⁵, Widyadi Setiawan⁶, I Putu Elba
Duta Nugraha⁷

¹²³⁴⁵⁶⁷ Program Studi Teknik Elektro, Fakultas Teknik, Universitas Udayana
Jl. Raya Kampus Unud Jimbaran, Kec. Kuta Selatan, Kabupaten Badung, Bali 80361
iwayanoktaarianta@gmail.com, steventulus03@gmail.com, chandra.id90@gmail.com,
indra@unud.ac.id, liejasa@unud.ac.id, widyadi@unud.ac.id, elba.nugraha@unud.ac.id

ABSTRAK

Lamanya proses deteksi kerusakan jalan menjadi penyebab lamanya proses perbaikan jalan di Indonesia. Proses deteksi yang dilakukan secara manual membutuhkan waktu yang lama dan sangat rentan terhadap kesalahan. Beberapa teknologi sudah dikembangkan untuk mengatasi permasalahan ini, namun harga yang mahal dan tingkat penggunaan yang kompleks seringkali menjadi kendala. Penelitian ini bertujuan untuk merancang purwarupa alat deteksi kerusakan jalan berbiaya rendah (*low cost*) yang mampu membantu mempercepat proses deteksi dan inspeksi kerusakan jalan di Indonesia. Sistem menggunakan prinsip penginderaan jauh dengan teknologi *Object Detection* YOLOv8s untuk mendeteksi empat kelas kerusakan jalan. Purwarupa alat dirancang menggunakan komponen elektronika alternatif yang ditenagai oleh Raspberry Pi 5 dan akselerator Hilo-8L. Penggunaan perangkat alternatif ini mampu menjalankan model *Object Detection* serta tugas komputasi lainnya secara stabil dengan rata-rata ≥ 27 FPS. Penggunaan GPS Module Ublox Neo-7M memiliki deviasi sebesar $\pm 3,5$ meter. Model object detection yang dirancang memiliki *overall accuracy* sebesar 92,96%. Hasil pengujian purwarupa pada kondisi diam menunjukkan sudut optimal 20–30 derajat dengan jarak deteksi 1, 3, dan 5 meter. Pada kondisi bergerak, performa sistem optimal berada pada kecepatan 20–30 km/jam dengan intensitas cahaya efektif pada pagi dan siang hari.

Kata kunci: *Object Detection*, YOLOv8s, Raspberry Pi 5, Hilo-8L

ABSTRACT

The lengthy process of road damage detection is one of the cause of delays of road repair processes in Indonesia. Manual detection method takes a lot of time and is highly prone to errors. Although several technologies have been developed to solve this problem, the high cost and complexity of use often become significant barriers. This research aims to design a prototype of a low-cost road damage detection tool that can help accelerate the process of road damage detection and inspection in Indonesia. The system uses remote sensing principles with YOLOv8s Object Detection technology to detect four classes of road damage. The prototype was designed using alternative electronic components powered by Raspberry Pi 5 and Hilo-8L accelerator. The use of this alternative device is capable of running Object Detection model as well as other computational tasks stably with an average of ≥ 27 FPS. The Ublox GPS Module Neo-7M GPS Module used in the system has a deviation of ± 3.5 meters. The designed object detection model has an overall accuracy of 92.96%. Testing results of the prototype in stationary conditions showed an optimal angle of 20-30 degrees with detection distances of 1, 3, and 5 meters. In moving conditions, optimal system performance is at a speed of 20-30 km/h with effective light intensity in the morning and afternoon.

Key words: *Object Detection*, YOLOv8s, Raspberry Pi 5, Hilo-8L

1. PENDAHULUAN

Meningkatnya volume kendaraan dan faktor lingkungan berdampak signifikan terhadap meningkatnya jumlah kerusakan jalan di Indonesia. Pada tahun 2023, tingkat kerusakan tipe berat dan ringan jalan raya nasional di Indonesia, mencapai 2769,74 kilometer atau sekitar 5,82% dari 47603,491 kilometer total panjang jalan nasional [1]. Kerusakan jalan seperti lubang (*pothole*) dan retakan (*crack*) dapat membahayakan pengguna jalan karena dapat menimbulkan kecelakaan, kerugian materi, dan kematian. Berdasarkan hal tersebut diperlukan adanya pemeliharaan kondisi jalan (preservasi jalan) secara berkala agar kerusakan dan dampaknya dapat dicegah. Oleh karena itu, pengetahuan tentang tingkat kerusakannya sangat penting untuk pemeliharaan yang efisien dan hemat biaya dengan tujuan untuk mempertahankan kondisi jalan yang baik dan aman. Langkah awal dalam kegiatan preservasi jalan adalah melakukan identifikasi atau inspeksi pada seluruh jaringan jalan. Kegiatan ini bertujuan untuk mengumpulkan data kerusakan pada perkerasan jalan yang ada pada setiap jaringan jalan.

Kegiatan ini umumnya dilakukan secara manual oleh petugas profesional yang melakukan inspeksi visual pada permukaan perkerasan jalan dengan berjalan kaki atau berkendara di sepanjang permukaan perkerasan jalan menggunakan kendaraan yang bergerak lambat [2]. Penggunaan metode manual memerlukan intervensi manusia yang signifikan yang terbukti memakan banyak waktu, mengingat panjangnya jaringan jalan yang harus diinspeksi. Metode ini juga sangat rentan terhadap kesalahan, pengamatan yang dilakukan oleh manusia seringkali subjektif dan tidak konsisten [3]. Selain itu, inspektur sering kali harus hadir secara fisik di jalur perjalanan yang ramai, yang berpotensi membahayakan. Lamanya proses inspeksi dapat berdampak pada keterlambatan kegiatan perbaikan, yang pada akhirnya dapat menyebabkan meningkatnya volume kerusakan serta munculnya kerusakan baru.

Penggunaan metode inspeksi otomatis dapat menjadi salah satu cara untuk mengatasi permasalahan ini. Namun, metode otomatis sering kali menggunakan kendaraan khusus yang dilengkapi dengan sensor yang canggih dan mahal. Kendaraan yang digunakan umumnya dilengkapi dengan beberapa sensor seperti sensor lidar, *road profiler*, dan kamera berkualitas tinggi dengan tujuan untuk menangkap aset jalan, termasuk gambar perkerasan, dan untuk memperoleh profil memanjang dan melintang jalan [[4], [5]]. Kendaraan seperti ini umumnya akan memiliki harga yang mahal dan cara penggunaan yang kompleks.

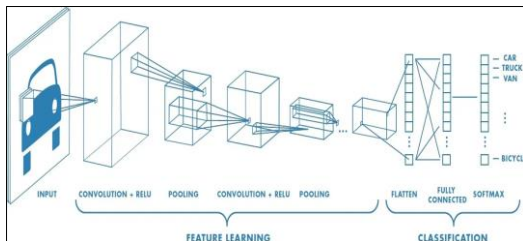
Baru-baru ini, penelitian tentang teknologi *machine learning* sebagai teknologi sensor untuk mendeteksi dan mengklasifikasikan kondisi jalan telah secara aktif dilakukan dengan kinerja deteksi yang tinggi. Teknologi ini menggunakan algoritma untuk mendeteksi area kerusakan jalan dengan menerapkan *Convolutional Neural Networks* (CNN) ke jaringan saraf yang terdiri dari beberapa lapisan. Sehingga dapat digunakan untuk deteksi kerusakan jalan pada permukaan jalan dengan akurat, karena dapat mendeteksi objek pada tingkat piksel [6]. Namun teknologi semacam ini biasanya membutuhkan perangkat dengan performa tinggi yang umumnya memiliki harga mahal dan ukuran besar. Seiring perkembangan teknologi, perangkat komputasi mini seperti Raspberry Pi 5 dapat menjadi salah satu alternatif pengganti perangkat berukuran besar, dengan catatan model *machine learning* dioptimalkan dan didukung dengan perangkat GPU eksternal.

Pada penelitian ini, solusi yang diusulkan adalah sebuah purwarupa alat deteksi kerusakan jalan berbasis teknologi *machine learning* dengan menggunakan metode CNN sebagai metode deteksi dan YOLOv8 sebagai algoritma pemodelan. Penelitian ini bertujuan untuk merancang dan membangun purwarupa alat deteksi kerusakan jalan berperforma tinggi dan mudah digunakan dengan menggunakan komponen elektronika alternatif.

2. STUDI PUSTAKA

2.1 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan jenis neural network yang digunakan memproses data dengan pola kisi, seperti gambar, video, dan media lainnya, yang terinspirasi oleh organisasi korteks visual hewan dan dirancang untuk secara otomatis dan adaptif mempelajari seluruh fitur spesial, dari pola pada citra tingkat rendah hingga tinggi [7]. CNN tersusun atas beberapa lapisan yang saling terhubung, setiap lapisan melakukan tugas tertentu untuk mengekstrak fitur-fitur penting dari gambar. Dapat dilihat pada Gambar 1 merupakan arsitektur CNN.



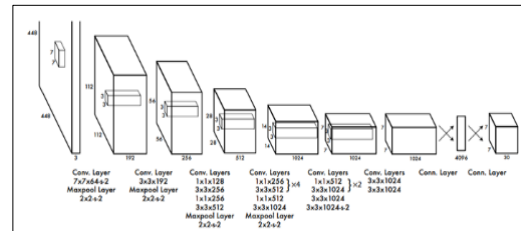
Gambar 1. Arsitektur CNN

Arsitektur CNN terdiri dari *feature learning* dan *classification layer*. Untuk mengklasifikasikan citra gambar CNN akan memproses gambar yang diinputkan, kemudian mengklasifikasikan pada kategori tertentu. Gambar input akan diubah menjadi bentuk *array* yang berisi nilai dari setiap piksel dengan resolusi tinggi x panjang x dimensi yang disebut dengan *channel* [8]. *Channel* terdiri dari 3 lapisan yang menandakan bahwa citra merupakan gambar RGB. Untuk gambar dengan warna *grayscale* akan memiliki satu lapisan. Proses pembelajaran pada CNN melibatkan penyampaian sejumlah besar data dari citra ke dalam jaringan, dan jaringan akan secara otomatis belajar untuk mengenali pola-pola yang ada di dalam gambar tersebut.

2.2 You Only Look Once (YOLO)

Algoritma *You Only Look Once* (YOLO) adalah algoritma deteksi objek berbasis CNN yang dirancang untuk memproses gambar dalam satu evaluasi tunggal. Algoritma YOLO berbeda dengan algoritma konvensional yang memindai area

gambar secara terpisah. YOLO membagi input citra menjadi *grid* $S \times S$, dimana setiap sel *grid* bertanggung jawab untuk mendeteksi objek apabila titik pusat objek jatuh kedalam wilayah sel tersebut. Deteksi kemudian dilakukan dengan memprediksi *bounding box*, *confidence score*, serta probabilitas dari objek yang terdeteksi [9]. Dapat dilihat pada Gambar 2 merupakan arsitektur YOLO.

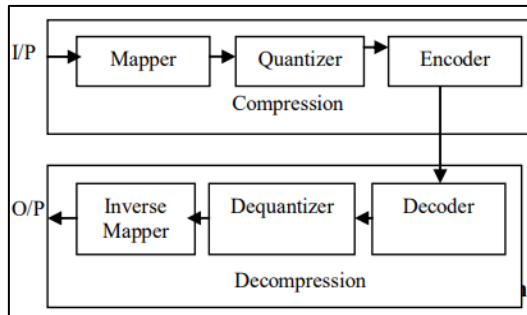


Gambar 2. Arsitektur YOLO

YOLO terdiri dari 24 *convolutional layer* dan dilanjutkan dengan 2 lapisan *fully connected layer*. *Convolutional layer* 1×1 pada YOLO digunakan untuk mengurangi dimensi fitur sebelum dilakukannya klasifikasi dan regresi. Setiap prediksi *bounding box* dari objek yang terdeteksi mencakup lima parameter: koordinat pusat (x, y), lebar (w), tinggi (h), dan nilai *confidence score*. *Confidence score* berfungsi untuk menunjukkan tingkat keyakinan model dalam mendeteksi objek yang ada di dalam *bounding box* tersebut. Semakin tinggi nilai *confidence score* (mendekati 1.0), maka semakin besar keyakinan model bahwa objek yang terdeteksi benar. Jika tidak terdapat objek dalam *bounding box* maka nilai *confidence score* akan menjadi nol.

2.3 Image Compression

Image compression merupakan teknik kompresi data yang digunakan untuk mengecilkan size dari sebuah gambar. Kompresi data bertujuan untuk mengurangi redundansi data gambar agar dapat menyimpan atau mengirimkan data dengan ruang atau *bandwidth* minimal, sekaligus menjaga resolusi dan kualitas gambar mirip dengan gambar aslinya [10]. Dalam kompresi gambar terdapat dua proses yaitu, *compression* dan *decompression* yang dapat dilihat pada Gambar 3.



Gambar 3. Blok Diagram *Image Compression*

Compression merupakan proses dimana *input* gambar akan melewati beberapa tahapan untuk dekompresi menggunakan algoritma tertentu untuk menghasilkan gambar yang telah terkompresi. Sedangkan *decompression* atau dekompresi adalah proses mengembalikan data yang telah dikompresi ke bentuk aslinya atau mendekati citra aslinya [11]. Seperti yang ditunjukkan pada Gambar 3, pada tahap pertama, *mapper* akan mengubah citra masukan menjadi format yang dirancang untuk mengurangi redundansi antar piksel. Tahap kedua, blok *quantizer* mengurangi akurasi dari keluaran *mapper* sesuai dengan kriteria yang telah ditetapkan. Pada tahap ketiga dan terakhir, *dekoder* akan membuat kode untuk keluaran *quantizer* dan memetakan keluaran sesuai dengan kode. Blok-blok ini melakukan, dalam urutan terbalik, operasi kebalikan dari blok koder simbol dan *mapper encoder*.

2.4 Koreksi Perspektif

Koreksi perspektif adalah suatu metode yang digunakan untuk memperbaiki distorsi geometris akibat sudut pandang kamera dalam suatu sistem pengolahan citra. Jika posisi kamera tidak tegak lurus terhadap permukaan horizontal, maka citra yang dihasilkan akan mengalami distorsi perspektif, di mana bentuk dan ukuran objek dapat terlihat tidak proporsional sesuai dengan kenyataannya. Distorsi pada citra menyebabkan objek yang berada di bagian bawah citra untuk tampak lebih besar dibandingkan objek yang berada lebih jauh di depan citra. Untuk memperoleh estimasi dari Panjang objek yang lebih akurat, maka

perlu dilakukan koreksi perspektif dengan menggunakan transformasi geometris. Metode ini akan mengkonversi nilai panjang objek dari pixel menjadi panjang sebenarnya dalam satuan cm dan kemudian mempertimbangkan sudut kamera terhadap bidang vertikal. persamaan yang digunakan adalah sebagai berikut [12].

$$p' = \left(\frac{p}{k}\right) \frac{1}{\cos \theta} \quad (1)$$

Di mana:

- p' adalah panjang sebenarnya (cm),
- p adalah panjang dalam piksel,
- k adalah skala konversi piksel ke cm,
- θ adalah Sudut kamera secara vertikal (90°) - Derajat kemiringan sudut lensa kamera

Dengan menggunakan rumus ini, dimungkinkan untuk melakukan koreksi akurat terhadap objek yang terdampak sudut pandang kamera. Studi [12] menerapkan pendekatan serupa dalam pengukuran area luka berbasis citra digital. Hasil penelitian menunjukkan bahwa koreksi terhadap sudut kemiringan (tilt) kamera secara signifikan meningkatkan akurasi dan presisi estimasi ukuran kerusakan pada permukaan jalan.

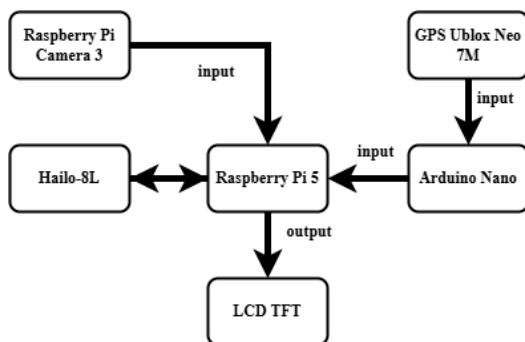
3. METODE PENELITIAN

Ruang lingkup penelitian ini meliputi perkerasan lentur (*flexible pavement*) dengan lokasi pengujian di Program Studi Teknik Elektro, Kampus Jimbaran dan ruas jalan sepanjang 1,4 km di Daerah Kampus Jimbaran, yang dibagi menjadi beberapa interval pengujian. Penelitian ini memfokuskan pada perancangan sistem berbiaya rendah (*low cost*) melalui pemanfaatan komponen elektronika alternatif untuk membangun perangkat *On-Board Unit* (OBU). Penelitian ini mempertimbangkan sudut pandang pemantauan kondisi jalan dengan kategori kerusakan, terutama dalam bentuk retakan dan lubang, yang sering terjadi di sebagian besar negara, salah satunya di Indonesia. Dataset dalam penelitian ini diperoleh secara *online* dari *Global Road Damage*

Detection Challenge (GRDDC) yang diselenggarakan oleh IEEE DataPort, mencakup 10.250 data kerusakan jalan dari berbagai negara. Secara *offline*, data dikumpulkan melalui dokumentasi langsung di Provinsi Bali sebanyak 2.869 data. Pelaksanaan penelitian terbagi menjadi beberapa bagian yang akan dijabarkan pada sub berikut.

3.1 Perancangan On-Board Unit (OBU)

Perangkat *On-Board Unit* (OBU) dirancang menggunakan arsitektur *Master-Slave*, di mana Raspberry Pi 5 berperan sebagai pusat pemrosesan utama (*master*), sedangkan Arduino Nano berfungsi sebagai *slave* yang bertugas melakukan pembacaan data dari sensor. OBU ini terdiri dari berbagai komponen elektronika alternatif, antara lain Raspberry Pi 5, Accelerator Hailo-8L, Raspberry Pi Camera Module V3, Arduino Nano, GPS Module Ublox Neo-7M, serta antena GPS dengan gain 3 dBi. Sistem ini juga menggunakan tiga buah baterai lithium 18650 sebagai sumber daya, disertai komponen pendukung lainnya untuk menunjang operasional perangkat. Seluruh perangkat diintegrasikan dengan menggunakan PCB dan terhubung dengan Raspberry Pi sebagai master kontrol. Gambar 4 menunjukkan diagram blok dari perangkat *On-Board Unit* (OBU).

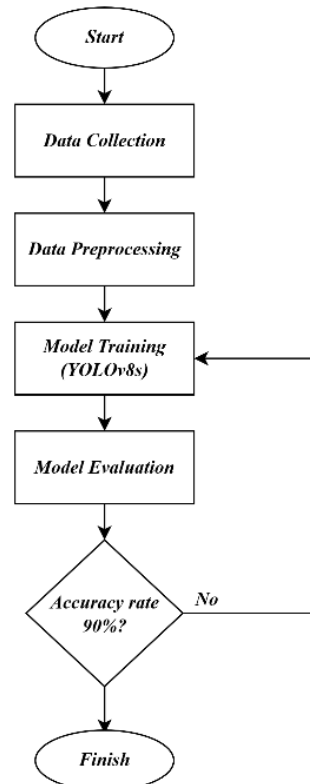


Gambar 4. Diagram Blok OBU

3.2 Perancangan Model Object Detection

Model *object detection* dirancang dengan menggunakan algoritma YOLOv8 tipe *small* (s) yang dikembangkan oleh ultralytics. Model YOLOv8 dipilih karena kemampuan model untuk dapat melakukan

deteksi objek secara *real-time* dengan akurasi yang tinggi. Perancangan model terdiri dari beberapa tahapan, yang dapat dilihat pada Gambar 5.



Gambar 5. Flowchart Perancangan Model

Dataset yang digunakan untuk pelatihan model terdiri dari 9.430 data, dengan 70% dataset (6.600) diperoleh dari sumber online, sedangkan 30% dataset (2.829) berasal dari pengumpulan secara *offline*. Seluruh dataset kemudian diberikan label menggunakan *software* Roboflow. Label dataset terdiri dari empat kelas, yaitu *pothole* (jalan berlubang), *longitudinal crack* (retak memanjang), *alligator crack* (retak kulit buaya), dan *transverse crack* (retak melintang). Pelabelan data akan menyimpan informasi tentang lokasi *bounding box* dan kelas objek. Informasi tersebut akan digunakan untuk membentuk kelas target objek. Dataset yang digunakan memiliki rincian untuk training sebanyak 70%, validasi 30%, dan pengujian sebanyak 10%.

Model training dilakukan dengan menggunakan *pre-trained* model *yolov8s.pt*

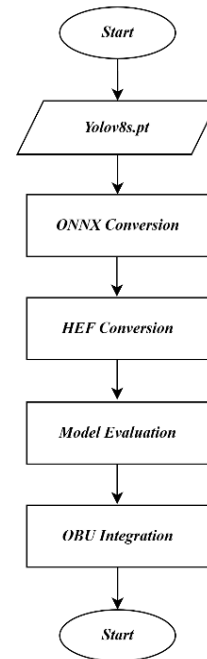
dengan dengan parameter *training*, 300 *epochs*, 16 *batch*, dan image size 640 x 640 px. Dalam proses training Jumlah *epoch* akan mempengaruhi tingkat konvergensi model atau mencapai tingkat kinerja yang optimal [13], namun Di sisi lain, terlalu banyak epoch dapat meningkatkan risiko *overfitting* [14]. Untuk menghindari risiko *overfitting* dalam proses training di integrasikan fitur *Early Stopping*, dimana ketika dalam beberapa *epoch* yang telah berjalan tidak ada peningkatan maka proses *training* akan diberhentikan. Proses *training* dilakukan menggunakan Laptop ASUS TUF Gaming F15 dengan sistem operasi Ubuntu 22.04 dan GPU NVIDIA RTX 3050 (CUDA *toolkit*). Proses pelatihan juga menggunakan beberapa *library* dan *framework*, seperti YOLOv8, dan PyTorch. Hasil akhir dari tahapan ini adalah file model berformat .pt (PyTorch).

Sebelum dapat digunakan model harus melalui tahapan evaluasi menggunakan *confusion matrix*. Tahapan evaluasi dilakukan untuk mengetahui akurasi dari model yang dibangun. Tingkat akurasi klasifikasi diukur dengan menggunakan parameter *accuracy*, *precision*, *recall*, dan *F-1 Score*. Evaluasi dilakukan dengan menguji model terhadap dataset validasi yang terdiri dari 1.886 gambar. Apabila model mencapai tingkat akurasi $\geq 90\%$, maka model dianggap layak untuk digunakan. Menurut studi [15], jika algoritma mencapai akurasi lebih dari 90%, maka akan memberikan akurasi deteksi yang lebih baik serta kecepatan pemrosesan secara *real-time* dalam sistem deteksi. Selain itu, penyediaan skalabilitas pada sistem deteksi dapat meningkatkan kinerja secara keseluruhan.

3.3 Metode Integrasi Model Object Detection

Model hasil pelatihan tidak dapat langsung digunakan pada perangkat *On-Board-Unit* (OBU) karena perangkat Hailo-8L hanya mendukung format .hef (*Hailo Executable Format*). Oleh karena itu, diperlukan proses konversi model. Pada

Gambar 5 menunjukkan *flowchart* integrasi model pada perangkat OBU.



Gambar 6. Flowchart Integrasi Model

Konversi model *Object detection* dilakukan melalui dua tahapan. Pada tahapan pertama model dikonversi ke format ONNX (*Open Neural Network Exchange*). Perbedaan arsitektur antara format PyTorch dan Hailo menjadi salah satu alasan mengapa model hasil train harus dikonversi ke format onnx terlebih dahulu. Dalam konversi ke onnx perlu diperhatikan dua parameter yaitu Opset dan ukuran gambar. Opset harus disesuaikan dengan versi opset yang didukung oleh hailo dan untuk ukuran gambar disesuaikan dengan ukuran gambar dari data set. Model onnx kemudian akan dikonversi ke format .hef menggunakan tool Hailo Dataflow Compiler dan Hailo Model Zoo. Ketika proses konversi ke format HEF berjalan, terjadi dua tahapan penting, yaitu kuantisasi dan optimisasi.

Kuantisasi mengubah bobot model dari float 32-bit ke integer 8-bit untuk efisiensi inferensi pada perangkat Hailo-8L. Optimisasi melakukan penghapusan layer yang tidak dibutuhkan dan juga mengatur alokasi memori pada model, serta menyusun ulang *neural network* agar sesuai dengan struktur internal dari Hailo-8L.

Setelah proses kuantisasi dan optimisasi selesai, dihasilkan model dalam format .hef yang siap untuk digunakan pada perangkat OBU. Sebelum diintegrasikan model akan dievaluasi terlebih dahulu pada perangkat Raspberry Pi 5, evaluasi bertujuan untuk mengukur performa model untuk melakukan inferensi secara *real-time*. Evaluasi dilakukan dengan memonitoring model menggunakan perintah `hailortcli-monitor`. Dengan perintah ini maka akan terlihat peningkatan FPS dari perangkat ketika menjalankan model secara *real-time*.

3.4 Metode Analisis Pengukuran Dimensi Kerusakan

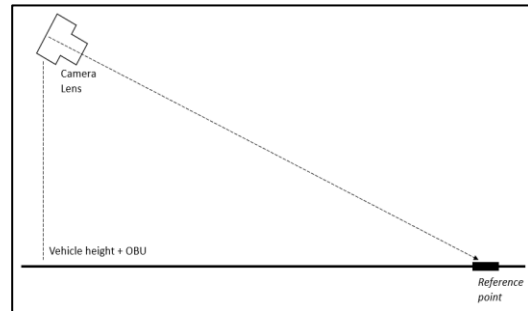
Pengukuran dimensi kerusakan jalan dilakukan mengkonversi nilai panjang dan lebar dalam nilai piksel dari *bounding box* ke dalam satuan sentimeter. Konversi nilai dilakukan dengan mempertimbangkan faktor tinggi kamera dari permukaan jalan, serta kemiringan kamera. Nilai konversi dari piksel ke cm dilakukan dengan mencari titik acu (*reference point*), dari titik pusat gambar ke kamera. *Reference point* kemudian digunakan untuk menentukan skala konversi berdasarkan perbandingan antara ukuran objek nyata dengan objek citra. Setelah skala diperoleh, maka panjang dan lebar *bounding box* dapat dikonversi ke satuan centimeter dengan mengalikan nilai piksel dengan skala yang telah ditentukan.

Langkah pertama dalam pengukuran dimensi kerusakan adalah dengan melakukan menghitung jarak dari kamera ke titik pusat pandang dihitung menggunakan persamaan sebagai berikut.

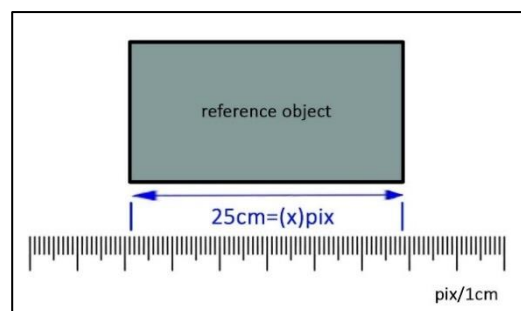
$$d = \frac{t}{\cos \theta} \quad (2)$$

Di mana d adalah jarak dalam cm, t adalah tinggi kamera dalam cm, dan θ adalah sudut kemiringan kamera. Informasi utama yang didapat dari *bounding box* berupa lebar dan tinggi *bounding box* dalam satuan piksel. Dari satuan piksel kemudian dikonversi menjadi satuan centimeter menggunakan skala konversi yang sudah didapatkan dari perhitungan kalibrasi terhadap objek referensi berupa papan berukuran 25 cm yang diletakkan di tengah titik pusat kamera,

seperti yang ditunjukkan pada Gambar 7 dan 8.

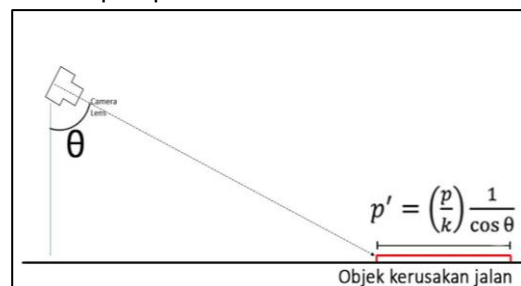


Gambar 7. Vehicle Height & Reference Point



Gambar 8. Skala Konversi Dari Objek Referensi Di Reference Point

Konversi secara langsung belum cukup untuk mengukur dimensi kerusakan jalan, karena distorsi perspektif yang dialami kamera akibat kemiringan kamera. Kamera yang dipasang dengan sudut kemiringan tertentu pada permukaan jalan akan menghasilkan citra yang lebih pendek daripada panjang aslinya. Hal ini terjadi karena proyeksi perspektif dari kamera yang menyebabkan adanya efek *foreshortening* (pemendekan) [12]. Oleh karena itu, dilakukan koreksi distorsi perspektif dengan menggunakan rumus transformasi seperti pada Persamaan (1). Gambar 9 menunjukkan ilustrasi dari proses koreksi distorsi perspektif.



Gambar 9. Koreksi Distorsi Perspektif

3.5 Metode Pengujian dan Analisis Hasil Pengujian.

Pengujian purwarupa dilakukan melalui dua skenario, yaitu kondisi diam (*steady state*) dan kondisi bergerak (*mobile state*). Pada kondisi diam, pengujian bertujuan untuk menentukan sudut kemiringan dan jarak deteksi yang optimal dalam proses identifikasi kerusakan. Pengujian dilakukan pada jarak 1, 3, 5, 7, dan 11 meter dengan variasi sudut kemiringan kamera sebesar 20°, 30°, dan 40° terhadap objek kerusakan. Sementara itu, pada kondisi bergerak, sistem diuji dalam keadaan kendaraan melaju dengan variasi kecepatan 20, 30, 40, dan 50 km/jam. Parameter yang diukur meliputi akurasi deteksi terhadap objek kerusakan, performa perangkat yang dinyatakan dalam satuan *frame per second* (FPS), serta akurasi koordinat GPS yang dievaluasi menggunakan metode *Root Mean Square Error* (RMSE) dengan membandingkan hasil koordinat terhadap nilai referensi. Kecepatan kendaraan dengan hasil terbaik akan digunakan sebagai acuan dalam pengujian lanjutan pada kondisi intensitas yang divariasikan.

4. HASIL DAN PEMBAHASAN

4.1 Perangkat *On-Board-Unit* (OBU)

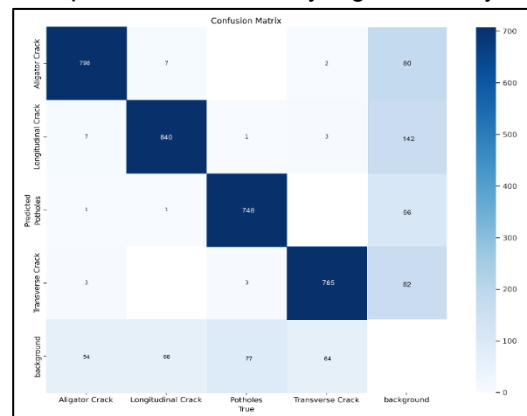
Perangkat *On-Board-Unit* (OBU) merupakan sebuah perangkat elektronik yang dipasang pada kendaraan bermotor, perangkat ini berfungsi untuk mendeteksi, merekam, dan menyimpan data kerusakan jalan. Perangkat OBU memiliki dimensi fisik dengan panjang 20 cm, lebar 15 cm, dan tinggi 30 cm, serta memiliki bobot keseluruhan sebesar 1060 gram. Perangkat OBU terbagi menjadi dua bagian utama, yaitu bagian perekaman citra dan bagian kontrol. Bagian perekaman citra berfungsi untuk menangkap citra kerusakan jalan. Sementara itu, bagian kontrol berfungsi sebagai pusat kontrol dan pemrosesan data. Dapat dilihat pada Gambar 10 merupakan hasil rancangan perangkat OBU.



Gambar 10. Perangkat *On-Board-Unit*

4.2 Analisis Hasil Pengujian Model *Machine Learning*

Pengujian model *machine learning* dilakukan dengan mengevaluasi kinerjanya menggunakan *confusion matrix* yang ditampilkan pada Gambar 11. *Confusion matrix* digunakan untuk menunjukkan performa model dalam mendeteksi berbagai jenis kerusakan jalan, yaitu *pothole*, *alligator crack*, *longitudinal crack*, dan *transverse crack*. Setiap baris pada *confusion matrix* merepresentasikan label yang diprediksi oleh model, sedangkan setiap kolom merepresentasikan label yang sebenarnya.



Gambar 11. *Confusion Matrix*

Dari *confusion matrix* pada Gambar 11, dapat dilihat bahwa model memiliki tingkat keberhasilan klasifikasi yang baik untuk setiap kelas. Sebagai contoh, pada kelas *Alligator Crack*, model berhasil mengklasifikasikan sebanyak 798 data dengan benar, dan hanya mengalami 89 kesalahan deteksi. Berikut ini merupakan hasil perhitungan matriks evaluasi yang diperoleh berdasarkan *confusion matrix* tersebut.

Tabel 1. *Matrix Evaluasi*

Metriks	Hasil (%)
<i>Accuracy</i>	92.96
<i>Precision</i>	91.60
<i>Recall</i>	89.10
<i>F1-Score</i>	90.40

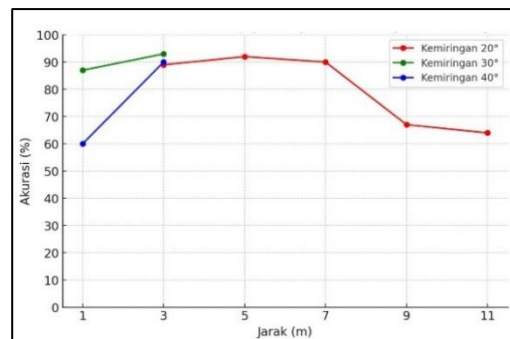
Berdasarkan matriks evaluasi pada Tabel 1, *accuracy* dari model yang dirancang menunjukkan kemampuan untuk mengklasifikasikan kerusakan dengan benar sebesar 92,96%. Nilai ini menunjukkan bahwa model *machine learning* yang dibangun memberikan akurasi deteksi yang lebih baik serta kecepatan pemrosesan secara *real-time* dalam sistem deteksi. Selain itu, tingginya nilai akurasi model *machine learning* menunjukkan bahwa model telah berhasil mempelajari data dengan baik dan mampu untuk mengklasifikasikan kelas kerusakan jalan kedalam kelas yang tepat, serta menunjukkan bahwa model dapat bekerja dengan baik untuk mengklasifikasikan dengan baik data yang belum pernah dilihat sebelumnya.

Kemudian nilai *precision* yang mencapai 91,60% mengindikasikan bahwa sebagian besar prediksi positif yang dilakukan oleh model adalah benar. Artinya, model memiliki tingkat ketepatan yang tinggi dalam mengidentifikasi kelas target. Sementara itu, nilai *recall* yang diperoleh sebesar 89,10% menunjukkan kemampuan model dalam mendeteksi sebagian besar data dari masing-masing kelas dengan baik, tanpa bias terhadap salah satu kelas tertentu.

Nilai *F1-score*, yang merupakan rata-rata harmonis dari *precision* dan *recall*, berada pada angka 90,40%. Nilai ini menunjukkan bahwa model memiliki keseimbangan yang baik antara ketepatan (*precision*) dan kelengkapan (*recall*) dalam mendeteksi dan mengklasifikasikan objek kerusakan jalan. Dengan hasil evaluasi tersebut, dapat disimpulkan bahwa model sudah layak untuk diterapkan dalam sistem deteksi kerusakan jalan secara otomatis.

4.3 Analisis Kondisi Steady State

Pengujian pada kondisi diam (*steady state*) dilakukan untuk mengetahui performa sistem dalam kondisi statis, pengujian ini berfokus untuk menentukan sudut dan jarak deteksi efektif untuk meletakkan algoritma deteksi untuk sistem mendeteksi kerusakan. Berdasarkan hasil pengujian yang ditunjukkan pada Gambar 12 menunjukkan nilai tertinggi pada setiap sudut cenderung terjadi pada jarak 3 meter, dengan rentang akurasi berada pada 89-93%. Namun, pada jarak lainnya terjadi penurunan akurasi pada beberapa sudut. Hal ini terlihat pada sudut 40°, di mana akurasi mengalami fluktuasi yang signifikan, terutama pada jarak 1 meter, dengan penurunan hingga 60%. Kemudian, pada sudut 20°, akurasi menurun menjadi 67% pada jarak 9 meter dan semakin turun hingga 64% pada jarak 11 meter. Kondisi ini disebabkan oleh perubahan perspektif kamera yang lebih tajam ke arah objek, yang menyebabkan distorsi perspektif, sehingga bentuk objek tampak lebih menyempit atau melebar, dan menurunkan akurasi deteksi.



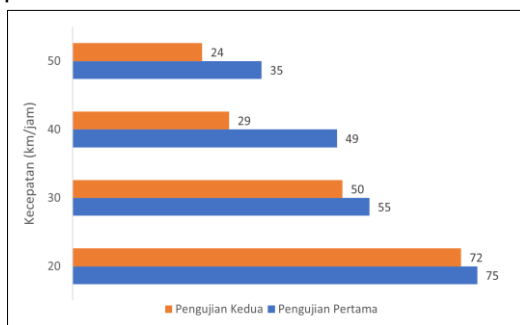
Gambar 12. Grafik Akurasi Terhadap Jarak pada Berbagai Sudut Kemiringan

Berdasarkan grafik di atas, menunjukkan bahwa sudut paling efektif dalam proses deteksi kerusakan jalan berada pada sudut 20° dan 30°. Pada kedua sudut ini, sistem menunjukkan rata-rata akurasi prediksi dimensi kerusakan yang lebih besar dari 89%. Sementara itu, untuk jarak deteksi terbaik, berdasarkan grafik pada Gambar 12, berada pada jarak 1, 3 dan 5 meter. Pada jarak ini, hasil deteksi sistem cenderung konsisten dengan tingkat akurasi tinggi, dan estimasi dimensi kerusakan jalan

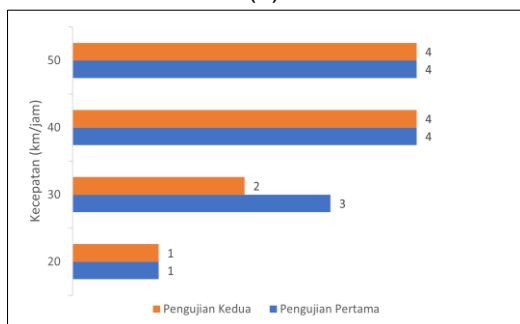
juga menunjukkan hasil yang akurat. Meskipun akurasi prediksi cukup tinggi, hasil prediksi ukuran kerusakan masih menunjukkan sedikit perbedaan dibandingkan dengan ukuran aslinya. Perbedaan ini cenderung berada pada kisaran ± 16 cm untuk sudut 20° dan ± 19 cm untuk sudut 30° . Oleh karena itu, dalam penelitian ini ditetapkan nilai toleransi dengan cara menghitung rata-rata selisih antara ukuran hasil prediksi dan ukuran sebenarnya. Berdasarkan perhitungan tersebut, diperoleh nilai toleransi sebesar $\pm 16,99$ cm untuk sudut 20° dan $\pm 19,99$ cm untuk sudut 30° .

4.4 Analisis Kondisi *Mobile State*

Pada kondisi bergerak (*mobile state*) dilakukan pengujian dengan variasi kecepatan, yaitu 20, 30, 40, dan 50 km/jam pada jalur sepanjang 1,4 km. Pada masing-masing kecepatan diuji sebanyak 2 kali menggunakan variabel yang sama. Hasil pengujian menunjukkan bahwa setiap kecepatan memiliki jumlah data terdeteksi dan tingkat kesalahan deteksi yang berbeda seperti yang ditunjukkan pada kedua grafik pada Gambar 13



(a)



(b)

Gambar 13. Grafik perbandingan jumlah data terdeteksi dan kesalahan deteksi

Berdasarkan Gambar 13a, terlihat bahwa jumlah kerusakan yang berhasil terdeteksi cenderung menurun seiring dengan peningkatan kecepatan kendaraan. Seperti pada kecepatan 20 km/jam, dari kedua pengujian yang dilakukan, sistem mampu mendeteksi lebih dari 70 kerusakan. Penurunan jumlah data terdeteksi mulai terlihat ketika kecepatan 30 km/jam di mana kerusakan yang berhasil terdeteksi berada pada rentang 50 kerusakan. Penurunan paling signifikan terjadi pada kecepatan 40 dan 50 km/jam, di mana terjadi perbedaan yang sangat jauh antara pengujian pertama dan kedua. Tren data tersebut menunjukkan bahwa seiring meningkatnya kecepatan akan menurunkan kemampuan sistem untuk mendeteksi kerusakan. Hal ini dikarenakan ketika sistem bergerak cepat, kamera pada sistem akan kesulitan untuk fokus terhadap objek kerusakan, sehingga banyak objek yang tidak terdeteksi

Selain dari jumlah kerusakan, meningkatnya kecepatan kendaraan juga berpengaruh terhadap kesalahan deteksi. Berdasarkan Gambar 13b menunjukkan bagaimana jumlah kesalahan deteksi meningkat seiring bertambahnya kecepatan kendaraan. Kesalahan terbanyak terjadi pada kecepatan 40 km/jam dan 50 km/jam di mana dari keseluruhan data yang berhasil terdeteksi terjadi empat kesalahan deteksi. Kesalahan terjadi dikarenakan berkurangnya waktu fokus kamera untuk menangkap citra kerusakan, yang menyebabkan terjadinya peningkatan *motion blur* yang mempersulit sistem untuk menangkap dan mengidentifikasi citra dengan akurasi yang optimal.

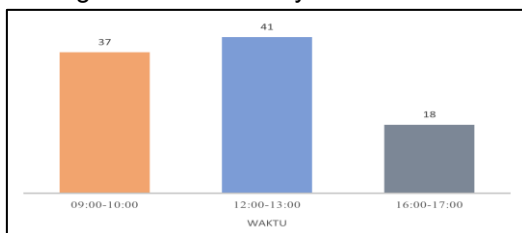


Gambar 14. Kualitas Citra pada Berbagai Kecepatan

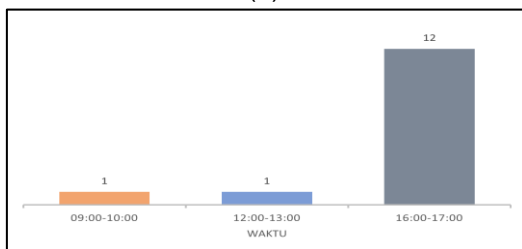
Pada Gambar 14 menunjukkan *motion blur* yang terjadi pada citra seiring meningkatnya kecepatan kendaraan. Terlihat pada kecepatan 20 km/jam dan 30 km/jam, kualitas gambar masih normal dengan kerusakan yang terlihat jelas. Namun, seiring meningkatnya kecepatan, terutama pada 40 km/jam dan 50 km/jam, terjadi *motion blur* yang menyebabkan kesalahan deteksi.

4.5 Analisa Performa Deteksi pada Intensitas Cahaya Berbeda

Pengujian bergerak dengan memvariasikan intensitas cahaya dilakukan pada waktu pagi hari (09:00-10:00), siang hari (12:00-13:00), dan sore hari (16:00-17:00). Pengujian dilakukan pada jalan yang berbeda dengan panjang lintasan 500 meter. Hasil pengujian menunjukkan bahwa Intensitas cahaya pada pagi dan siang hari menghasilkan jumlah data valid terbanyak Gambar 15a. Pada pengujian di pagi hari, sistem berhasil mendeteksi 38 kerusakan, dengan 37 di antaranya merupakan data valid. Sementara itu, pada siang hari terdeteksi 43 kerusakan, dengan 42 diantaranya valid. Namun, ketika sistem diuji pada sore hari, terjadi penurunan jumlah data valid. Dari 30 kerusakan yang terdeteksi, hanya 18 data yang valid, sedangkan 12 data lainnya tidak valid.



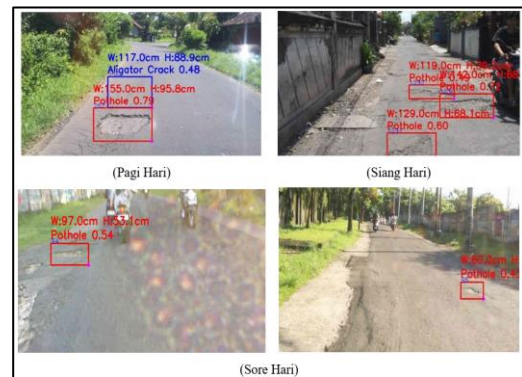
(a)



(b)

Gambar 15. (a) Jumlah Data Valid, (b) Jumlah Data Tidak Valid

Berdasarkan Gambar 15b dapat diketahui bahwa jumlah kesalahan deteksi meningkat secara signifikan pada pengujian di sore hari, di mana dari 30 kerusakan yang berhasil terdeteksi, sebanyak 12 data merupakan data yang salah terdeteksi. Peningkatan kesalahan mengindikasikan bahwa berkurangnya intensitas cahaya pada sore hari berdampak negatif terhadap menurunnya akurasi deteksi dari sistem. Dapat diperhatikan pada Gambar 16 merupakan perbandingan kualitas citra pada tiga intensitas cahaya yang berbeda.

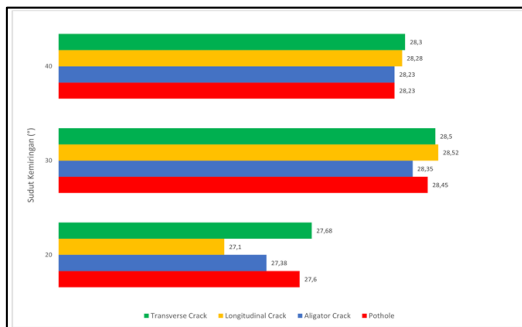


Gambar 16. Kualitas Citra pada Berbagai Intensitas Cahaya

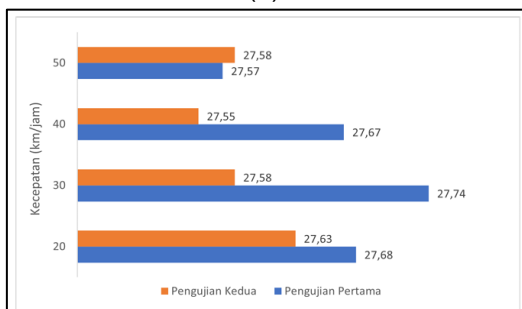
Pada sore hari cahaya matahari akan cenderung jauh lebih rendah di cakrawala, menghasilkan bayangan panjang yang dapat menutupi atau menyamarkan detail jalan. Pada kondisi ini sinar matahari akan cenderung berwarna orange dan cahaya yang lebih redup, kondisi ini mengurangi kontras antara lubang dan retakan dengan permukaan jalan, sehingga algoritma pada sistem kesulitan membedakan fitur yang relevan. Selain itu, cahaya yang rendah juga berdampak pada munculnya banyak bayangan, seperti bayangan pohon, kabel, tiang listrik, dan benda lainnya. Hal ini dapat mengakibatkan kesalahan deteksi, misalnya kabel yang terdeteksi sebagai *longitudinal* atau *transverse crack*. Selain itu jika diperhatikan, citra yang dihasilkan pada waktu sore hari memiliki lebih banyak *noise* yang disebabkan oleh pencahayaan matahari. Citra dengan kualitas seperti ini dapat menyulitkan proses analisis maupun pelaporan.

4.6 Analisis Perform Purwarupa Alat

Penggunaan perangkat elektronika alternatif mampu menghasilkan purwarupa alat yang mumpuni, hal ini terlihat dari peningkatan performa perangkat ketika menjalankan model *machine learning* baik ketika berada dalam kondisi bergerak maupun diam. Berdasarkan hasil pengujian integrasi Accelerator Hailo-8L pada Raspberry Pi 5 mampu meningkatkan performa perangkat ketika menjalankan model *machine learning* dan berbagai tugas komputasi. Sebelum diintegrasikan dengan perangkat Hailo-8L, Raspberry Pi 5 hanya mampu mencapai 0,6 FPS ketika menjalankan model YOLOv8s dengan 9,430 dataset. Sedangkan, ketika diintegrasikan dengan perangkat Hailo-8L perangkat mampu mencapai FPS rata-rata 27 FPS ketika seluruh tugas komputasi aktif. Dapat dilihat pada Gambar 17 menunjukkan FPS perangkat ketika beroperasi pada berbagai kondisi. Baik kondisi diam atau bergerak performa FPS perangkat tetap berada pada rata-rata ≥ 27 FPS, yang menunjukkan kemampuan perangkat dalam mengefisiensi sumber daya dalam pemrosesan citra dan menjalankan algoritma deteksi.



(a)



(b)

Gambar 17. Performa Perangkat (a) *Steady State* (b) *Mobile State*

Berdasarkan hasil pengujian dan evaluasi menggunakan Root Mean Square Error (RMSE), yang membandingkan koordinat hasil deteksi dengan koordinat sebenarnya, diperoleh nilai RMSE sebesar 0,00003167 derajat untuk GPS Module Ublox Neo-7M. Jika dikonversi ke satuan meter, nilai tersebut menunjukkan perbedaan antara titik hasil model dan data uji lapangan sebesar $\pm 3,5$ meter. Nilai ini menunjukkan performa perangkat yang baik, di mana perbedaan $\pm 3,5$ meter masih sesuai, mengingat jarak antara kamera dengan objek diatur pada jarak 3 meter. Selain itu, perbedaan sejauh ini masih dapat diamati secara visual.

5. KESIMPULAN

Berdasarkan analisis hasil pengujian yang telah dilakukan, dapat disimpulkan bahwa purwarupa alat deteksi kerusakan jalan berhasil dirancang menggunakan Raspberry Pi 5 dan akselerator Hailo-8L sebagai komponen utama. Perangkat mampu menjalankan tugas komputasi dengan rata-rata kecepatan lebih dari atau sama dengan 27 FPS serta akurasi GPS memiliki deviasi $\pm 3,5$ meter. Model *Object Detection* berhasil dirancang menggunakan YOLOv8s yang dioptimalkan pada perangkat *On-Board-Unit* (OBU). Model ini menunjukkan performa deteksi yang tinggi, dengan nilai akurasi sebesar 92,96%, precision sebesar 91,6%, recall sebesar 89,1%, dan F1-Score sebesar 90,4%.

Hasil pengujian menunjukkan bahwa sudut kemiringan kamera yang optimal berada pada rentang 20° – 30° , dengan jarak deteksi terbaik berada pada jarak 1, 3, dan 5 meter. Kemiringan sudut dan jarak deteksi sangat memengaruhi akurasi sistem deteksi. Hal ini disebabkan oleh distorsi perspektif yang terjadi, sehingga citra objek yang terdeteksi tampak lebih pendek dari ukuran sebenarnya. Pada kondisi bergerak, kecepatan kendaraan yang optimal untuk pengoperasian sistem berada pada rentang 20–30 km/jam. Di atas kecepatan tersebut, akan terjadi efek *motion blur* yang berdampak negatif terhadap akurasi deteksi.

Intensitas cahaya pada sore hari akan meningkatkan jumlah kesalahan deteksi. Kondisi pencahayaan yang tidak optimal dapat mempengaruhi kualitas citra yang ditangkap, sehingga berdampak pada penurunan akurasi.

6. DAFTAR PUSTAKA

- [1] Badan Pusat Statistik, Panjang Jalan Menurut Provinsi dan Tingkat Kewenangan Pemerintahan (km), 2023. Jakarta: BPS, 2023.
- [2] D. Arya, H. Maeda, S. K. Ghosh, D. Toshniwal, A. Mraz, T. Kashiya, and Y. Sekimoto, "Deep learning-based road damage detection and classification for multiple countries," *Automation in Construction*, vol. 132, p. 103935, 2021.
- [3] R. H. Pramestya, Deteksi dan klasifikasi kerusakan jalan aspal menggunakan metode YOLO berbasis citra digital. Tesis, Institut Teknologi Sepuluh Nopember, Surabaya, 2018.
- [4] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang, "CrackTree: Automatic crack detection from pavement images," *Pattern Recognition Letters*, vol. 33, no. 3, pp. 227–238, 2012.
- [5] X. Weng, Y. Huang, and W. Wang, "Segment-based pavement crack quantification," *Automation in Construction*, vol. 105, p. 102819, 2019.
- [6] S. Shim, J. Kim, S. W. Lee, and G. C. Cho, "Road damage detection using super-resolution and semi-supervised learning with generative adversarial network," *Automation in Construction*, vol. 135, p. 104139, 2022.
- [7] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: An overview and application in radiology," *Insights into Imaging*, vol. 9, no. 4, pp. 611–629, 2018.
- [8] M. F. Herlambang, A. N. Hermana, and K. R. Putra, "Pengenalan karakter huruf Braille dengan metode convolutional neural network," *Systemic: Information System and Informatics Journal*, vol. 6, no. 2, pp. 20–26, 2021.
- [9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 27–30, 2016, pp. 779–788.
- [10] H. S. Samra, "Image Compression Techniques," *International Journal of Computers & Technology*, vol. 2, no. 2, 2012.
- [11] H. Hidayat, T. Pamungkas, and W. Zarman, "Implementasi algoritma kompresi LZW pada database server," *Komputa: Jurnal Ilmiah Komputer dan Informatika*, vol. 2, no. 1, 2013.
- [12] P. Foltynski, "Ways to increase precision and accuracy of wound area measurement using smart devices: Advanced app Planimator," *PLoS One*, vol. 13, no. 3, e0192485, 2018.
- [13] M. Alamgeer et al., "Optimal fuzzy wavelet neural network based road damage detection," *IEEE Access*, vol. 11, pp. 61986–61994, 2023.
- [14] T. J. W. Adi, P. Suprobo, and Y. E. P. R. Waliulu, "iRodd (intelligent-road damage detection) for real-time infrastructure preservation in detection, classification, calculation, and visualization," *Journal of Infrastructure, Policy and Development*, vol. 8, no. 11, p. 6162, 2024.
- [15] H. Zhang et al., "A new road damage detection baseline with attention learning," *Applied Sciences*, vol. 12, no. 15, p. 7594, 2022.