

Sistem Deteksi Objek Menggunakan *Depth Camera* Untuk *Unmanned Ground Vehicle* Di Lingkungan *Outdoor*

Laduni Dinata
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia

ladunidinata@student.telkomuniversity.
ac.id

Angga Rusdinar
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia

anggarusdinar@telkomuniversity.ac.id

Azam Zamhuri Fuadi
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia

azamzamhurifuadi@telkomuniversity.
ac.id

Abstrak — Penggunaan *Unmanned Ground Vehicle* (UGV) di area luar ruangan membutuhkan sistem deteksi objek yang mampu bekerja secara optimal untuk membantu sistem navigasi untuk menghindari berbagai rintangan secara otomatis. Salah satu tantangan dalam navigasi UGV adalah mendeteksi dan mengukur jarak objek seperti pohon secara real time di medan yang kompleks. Penelitian ini bertujuan untuk merancang dan membangun sistem deteksi objek menggunakan algoritma *You Only Live Once* (YOLO) v11 yang dipadukan dengan *depth camera* Intel RealSense D435I, berfokus pada pohon sebagai objek tunggal. Sistem ini dirancang guna menunjang pergerakan mandiri UGV di lingkungan outdoor. Metodologi yang digunakan mencakup pengumpulan dataset secara mandiri sebanyak 3.502 gambar pohon dari area sekitar Gedung P Universitas Telkom, proses pelabelan menggunakan platform Roboflow, pelatihan model YOLOv11 dengan teknik augmentasi data, serta integrasi dengan algoritma robust center untuk mengestimasi jarak. Pelatihan dilakukan di Google Colab dengan GPU NVIDIA A100. Evaluasi performa dilakukan menggunakan metrik precision, recall, F1-score, dan mean Average Precision (mAP), serta perbandingan terhadap model YOLOv10.

Hasil pengujian menunjukkan bahwa model YOLOv11 memiliki kinerja deteksi dengan precision sebesar 88,1%, recall 85,6%, dan mAP 0.5 mencapai 90,5%. Estimasi jarak paling akurat diperoleh dengan nilai center_ratio sebesar 0,3 pada metode robust center. Sistem yang dikembangkan mampu mendukung sistem deteksi objek UGV di lingkungan outdoor dan memiliki potensi aplikasi di berbagai bidang seperti pertanian, militer, serta penanggulangan bencana.

Kata Kunci: Kendaraan Tanpa Awak, Deteksi Objek, Depth Camera, YOLOV11, Robust Center, Deep Learning.

PENDAHULUAN

Kendaraan tanpa awak (UGV) banyak digunakan di lingkungan luar ruangan seperti hutan dan lahan pertanian. Agar dapat bergerak secara otonom, UGV perlu mendeteksi objek dan mengukur jarak secara real time untuk menghindari rintangan seperti pohon [1]. Salah satu solusi efektif adalah menggabungkan *depth camera* Intel RealSense dengan algoritma deteksi objek YOLO. Kamera ini mampu menangkap gambar dan kedalaman secara bersamaan, sehingga dapat mengenali objek dan mengukur jaraknya

hanya dengan satu perangkat [2]. Teknologi ini terbukti andal digunakan di lingkungan berpohon dengan pencahayaan tidak merata dan mampu mengukur jarak hingga 10 meter. Dengan integrasi YOLOv11, sistem dapat mendeteksi pohon dan memperkirakan jaraknya secara akurat [3]. Penelitian ini bertujuan merancang sistem visual untuk mendeteksi dan mengukur jarak pohon menggunakan YOLOv11 dan *depth camera* Intel RealSense D435i, guna mendukung navigasi otonom UGV secara efisien di medan outdoor.

I. KAJIAN TEORI

Berikut adalah teori-teori yang mendukung penulisan jurnal penelitian yang dilakukan oleh penulis:

A. UGV

Unmanned Ground Vehicle (UGV) merupakan kendaraan darat yang dapat beroperasi tanpa pengemudi, baik secara mandiri maupun melalui kendali jarak jauh, tergantung pada tingkat otonomi yang dimilikinya [4]. UGV umumnya dilengkapi dengan berbagai sensor seperti kamera, lidar, dan radar yang memungkinkan pengumpulan data lingkungan secara *real time*. Informasi ini kemudian diproses oleh sistem kontrol untuk navigasi, deteksi hambatan, serta pelaksanaan misi secara otonom. Kemampuan ini memungkinkan UGV beroperasi di area yang sulit dijangkau atau berisiko tinggi, seperti medan berbatu, kawasan terdampak bencana, atau lingkungan industri ekstrem [4]. Seiring perkembangan teknologi kecerdasan buatan (AI), UGV kini mampu beradaptasi dengan kondisi lapangan dan menjalankan tugas kompleks secara lebih fleksibel dan efisien [4].

B. Deteksi Objek

Deteksi objek merupakan cabang penting dalam visi komputer yang bertujuan mengenali dan menentukan posisi objek dalam gambar atau video melalui proses klasifikasi dan lokalisasi menggunakan bounding box [5]. Metode tradisional seperti HOG atau Haar-like dengan algoritma SVM memiliki keterbatasan dalam menghadapi variasi skala, sudut, dan pencahayaan [5]. Perkembangan *deep learning* menghadirkan solusi lebih akurat dan efisien melalui jaringan saraf seperti CNN. Algoritma seperti Faster R-CNN menghasilkan proposal wilayah objek, sedangkan YOLO (*You Only Look Once*) mampu melakukan deteksi secara cepat dan menyeluruh dalam satu tahap, ideal untuk aplikasi *real time* [5]. Teknologi ini telah diterapkan luas di berbagai

bidang seperti pengawasan, kendaraan otonom, kesehatan, pertanian, hingga ritel [5].

C. Algoritma

Algoritma adalah serangkaian langkah-langkah logis yang dirancang untuk menyelesaikan masalah atau mencapai tujuan tertentu. Langkah-langkah ini tidak hanya digunakan di dunia komputer, tetapi juga dapat diterapkan dalam berbagai aktivitas sehari-hari, seperti memasak, berolahraga, atau mengatur sebuah proyek. Dalam ilmu komputer, algoritma biasanya adalah prosedur yang memproses data untuk menghasilkan hasil tertentu [6].

D. Deep Learning

Deep learning merupakan cabang dari kecerdasan buatan (AI) yang meniru cara kerja otak manusia dalam memproses data secara otomatis dan akurat. Meskipun telah dikembangkan sejak 1950-an, penerapannya baru efektif setelah dekade 1990-an dengan prinsip yang masih digunakan hingga kini [7]. Teknologi ini dirancang untuk menyederhanakan algoritma dan meningkatkan fleksibilitas dalam mengenali data tidak terstruktur seperti gambar, video, dan teks [7]. Berbeda dari *machine learning* konvensional yang bergantung pada data terstruktur, *deep learning* memproses informasi melalui lapisan-lapisan jaringan saraf (*neural network*) yang saling terhubung. Lapisan awal mengenali fitur dasar seperti garis dan warna, sementara lapisan berikutnya membentuk representasi kompleks dari data. Kemampuan ini menjadikan *deep learning* sangat efektif untuk analisis data visual secara mendalam [7].

E. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) adalah arsitektur *deep learning* yang dirancang khusus untuk mengolah data citra dua dimensi. CNN mampu mengekstraksi fitur spasial dari gambar menggunakan lapisan konvolusi, sehingga lebih efektif dibandingkan metode konvensional dalam tugas klasifikasi objek [8]. Lapisan konvolusi bekerja dengan menerapkan filter untuk mendeteksi pola penting dalam gambar, yang selanjutnya digunakan untuk proses klasifikasi secara akurat [9].

F. YOLO (You Only Look Once)

YOLO, singkatan dari You Only Look Once, adalah algoritma deteksi objek yang dirancang untuk mengenali berbagai jenis objek dalam gambar secara real-time. Algoritma ini memperlakukan deteksi objek sebagai masalah regresi, di mana tujuan utamanya adalah memberikan probabilitas kelas untuk setiap objek yang terdeteksi serta lokasi kotak pembatas (*bouding box*). Yolo menggunakan jaringan saraf konvolusi (Convolutional Neural Network atau CNN) sebagai fondasi utama dalam proses deteksi [10].

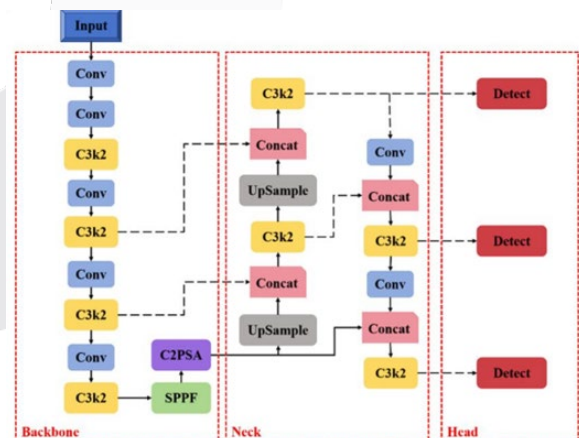
G. YOLOv10

YOLOv10 dikembangkan sebagai pengembangan lanjutan dari algoritma deteksi objek yang bertujuan untuk menciptakan model yang lebih ringan, efisien, dan terintegrasi secara menyeluruh. Model ini pertama kali diperkenalkan oleh tim Meta Research pada tahun 2024. Salah satu pendekatan utama dari arsitektur ini adalah penghapusan komponen bertingkat, seperti penggunaan

anchor box dan proses Non Maximum Suppression (NMS), yang biasanya ditemukan dalam pipeline deteksi objek tradisional. Pada bagian head, YOLOv10 menghadirkan sistem deteksi yang terintegrasi penuh, di mana proses prediksi dilakukan secara langsung di setiap titik fitur. Sistem ini tidak lagi menggunakan metode anchor based maupun anchor free klasik, melainkan langsung memprediksi kelas dan lokasi *bouding box* dalam satu langkah. Dengan pendekatan ini, proses inferensi menjadi lebih sederhana dan efisien [11].

H. YOLOv11

YOLOv11 merupakan versi lanjutan dari keluarga algoritma YOLO yang dikembangkan untuk meningkatkan akurasi deteksi serta kemampuan adaptasi model terhadap berbagai jenis objek dan kondisi visual. Berbeda dari pendekatan YOLOv10 yang berfokus pada penyederhanaan proses deteksi dan penghapusan modul tambahan, YOLOv11 hadir dengan arsitektur modular baru yang lebih kompleks namun tetap efisien. Model ini menggabungkan beberapa komponen baru, seperti modul C3k/C3k2 yang memperkuat jalur ekstraksi fitur, serta modul SPPF (Spatial Pyramid Pooling Fast) dan C2PSA (Convolutional Parallel Spatial Attention) yang membantu model lebih peka terhadap variasi ukuran dan posisi objek dalam citra. Struktur dasar YOLOv11 tetap terdiri dari tiga bagian utama, yaitu Backbone, Neck, dan Head, namun masing-masing bagian mengalami pengembangan yang signifikan. Bagian backbone menggunakan pola multi-cabang untuk menangkap informasi dari berbagai skala secara lebih efektif. Neck dirancang agar mampu menyatukan dan menyelaraskan informasi spasial dan semantik dari beragam level fitur, sehingga memperkaya konteks visual. Sedangkan bagian Head tetap menggunakan metode anchor free, namun kini dilengkapi dengan teknik alokasi target dan loss function adaptif yang menjadikan proses pelatihan lebih stabil dan hasil prediksinya lebih akurat [11].



GAMBAR 1
Arsitektur Jaringan YOLOv11

I. Penentuan Jarak Objek Terdeteksi

Algoritma *robust center* adalah pendekatan dua tahap untuk estimasi jarak berbasis *depth camera*, yang menggabungkan pemfilteran spasial dan eliminasi pencilan statistik guna meningkatkan akurasi pengukuran kedalaman dalam *bouding box* objek. Pada tahap pertama, algoritma

hanya menghitung kedalaman dari 30% area tengah bounding box (ditentukan melalui *center_ratio*), untuk menghindari piksel latar belakang yang sering mengganggu estimasi [ROI-based spatial filtering]. Tahap kedua menerapkan filter berbasis *z-score* untuk menyaring nilai kedalaman ekstrem, seperti nol atau lebih dari 10.000 mm, guna menghilangkan derau dari permukaan reflektif atau error sensor. Kombinasi kedua teknik ini membuat estimasi kedalaman lebih stabil dan akurat, serta lebih andal dibandingkan metode rata-rata konvensional, terutama untuk aplikasi deteksi dan pelacakan objek berbasis kamera kedalaman seperti Intel RealSense.

J. Evaluasi Hasil

1. Confusion Matrix, atau dikenal juga sebagai error matrix, adalah tabel yang merangkum hasil prediksi pada masalah klasifikasi. Tabel ini mencatat jumlah prediksi yang benar maupun salah, kemudian mengelompokkan data tersebut berdasarkan kelas. Dengan demikian, Confusion Matrix tidak hanya memberikan informasi tentang tingkat kesalahan yang dibuat oleh model klasifikasi, tetapi juga jenis kesalahan yang terjadi [9].

Berikut adalah beberapa ketentuan yang digunakan dalam Confusion Matrix :

- 1) Positive (P): Kondisi aktual memiliki nilai positif.
Negative (N): Kondisi aktual memiliki nilai negatif.
- 2) True Positive (TP): Kondisi aktual positif dan diprediksi positif oleh model.
- 3) True Negative (TN): Kondisi aktual negatif dan diprediksi negatif oleh model
- 4) False Positive (FP): Kondisi aktual negatif, tetapi diprediksi positif oleh model
- 5) False Negative (FN): Kondisi aktual positif, tetapi diprediksi negatif oleh model.

Confusion Matrix adalah alat yang sangat berguna untuk mengevaluasi kinerja model klasifikasi, karena memberikan gambaran yang jelas tentang bagaimana model menangani prediksi untuk setiap kelas.

2. Presisi dihitung dengan membagi jumlah sampel positif yang berhasil diklasifikasikan dengan benar oleh model dengan total sampel yang diprediksi sebagai positif yang diprediksi seperti pada Persamaan 2.1. Presisi yang tinggi menunjukkan bahwa sebagian besar prediksi positif yang dibuat oleh model benar-benar positif, dengan sedikit terjadi false positive [9].

$$\text{Presisi} = \frac{TP}{TP+FP} \quad (2.1)$$

High Recall, Low Precision: Kondisi ini terjadi ketika sebagian besar sampel positif berhasil dikenali oleh model (FN rendah), namun banyak juga False Positive yang terdeteksi.

Low Recall, High Precision: Pada kondisi ini, model gagal mengenali sebagian besar sampel positif (FN tinggi), tetapi prediksi positif yang dibuat oleh model sangat akurat, dengan sedikit False Positive rendah.

3. Recall adalah ukuran yang menunjukkan seberapa baik model yang dapat mendeteksi semua sampel positif. Recall dihitung dengan membagi jumlah sampel positif yang berhasil di klasifikasikan dengan benar oleh model dengan total sampel positif yang ada. Nilai recall yang tinggi berarti model mampu mengenali sebagian besar sampel positif dengan benar, menunjukkan bahwa kesalahan False Negative sangat sedikit. Recall dapat dihitung dengan Persamaan 2.2[9]

$$\text{Recall} = \frac{TP}{TP + FP} \quad (2.2)$$

4. Mean Average Precision (mAP) adalah rata-rata nilai dari Average Precision (AP) yang digunakan sebagai metrik evaluasi untuk menilai kinerja model deteksi objek. mAP menggabungkan hasil presisi dan recall dari semua kelas objek yang terdeteksi, sehingga memberikan Gambaran menyeluruh tentang seberapa baik model bekerja.

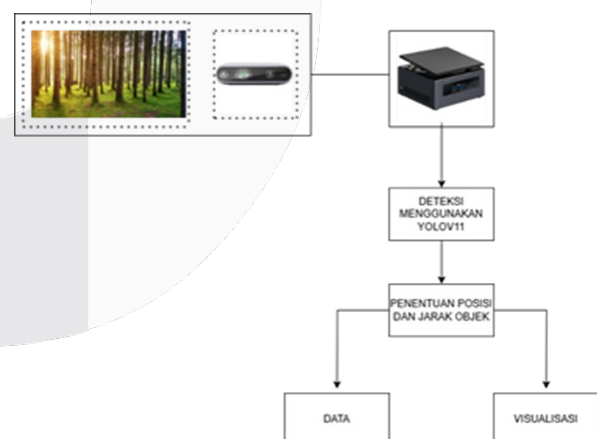
Nilai Average Precision (AP) dihitung menggunakan precision (dari persamaan 2.1) dan recall (dari persamaan 2.2). Setelah itu nilai AP dihitung berdasarkan integrasi presisi terhadap recall, seperti yang dijelaskan pada Persamaan 2.4. Kemudian nilai rata-rata AP untuk semua kelas dihitung menggunakan persamaan 2.5 untuk mendapatkan nilai mAP.

Metode ini sangat berguna dalam menilai model deteksi objek, terutama untuk memastikan model mampu mendeteksi berbagai kelas objek dengan akurasi dan konsistensi yang baik [9].

$$AP = \int_{recall_n}^{recall_{n+1}} Precision(recall_{n+1}) - recall_n \quad (2.4)$$

II. METODE

A. Desain Sistem



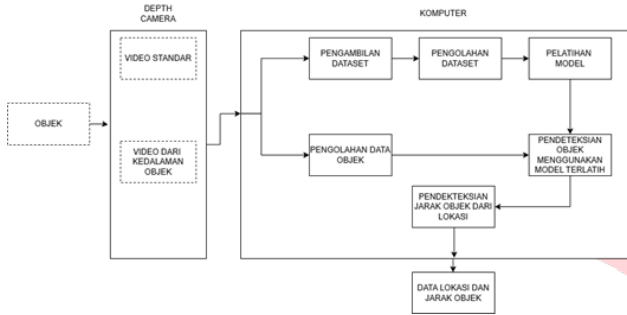
GAMBAR 2
Desain Sistem

Sistem ini memanfaatkan depth camera untuk menangkap citra serta komputer sebagai perangkat pemrosesan data. Sistem tersebut dirancang agar Unmanned Ground Vehicle (UGV) mampu mendeteksi objek yang menjadi rintangan, Sehingga UGV dapat bergerak tanpa menabrak hambatan. Berkat penggunaan depth camera, sistem tidak hanya dapat menemukan posisi objek, tetapi juga menghitung jarak antara

UGV dengan objek rintangan, sehingga UGV lebih mudah menghindari rintangan.

Pada gambar 3.1 ditunjukkan bagaimana sistem bekerja secara ringkas. Depth camera mengambil citra dari lingkungan sekitar dan mengirimkannya ke komputer untuk diproses. Di dalam komputer data yang telah dilatih digunakan untuk mendeteksi posisi objek. Selain itu komputer juga menentukan jarak antara UGV dan objek. Hasil akhirnya berupa data yang memuat informasi lokasi atau posisi serta jarak masing-masing rintangan yang kemudian dapat dimanfaatkan lebih lanjut.

B. Diagram Blok

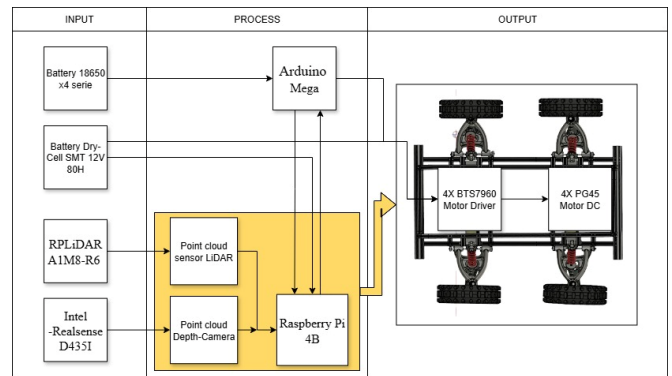


GAMBAR 3 (Diagram Blok Sistem)

Blok diagram diatas menampilkan alur kerja sistem pada Unmanned Ground Vehicle (UGV) untuk mendeteksi rintangan (obstacle) secara otomatis. Pertama, sistem menggunakan dua masukan utama yaitu video standar dan video kedalaman objek, yang dikumpulkan secara terintegrasi sebagai dasar pembentukan dataset. Proses awal dimulai dengan tahapan pengambilan dataset di mana beragam video dihimpun dan dievaluasi guna menjamin mutu serta variasi sampel. Setelah pengumpulan selesai dataset tersebut diolah melalui beberapa langkah termasuk pembersihan data, peningkatan kualitas citra, dan pengekstraksian fitur-fitur penting yang berkaitan dengan rintangan. Selanjutnya dataset yang telah diolah dipakai dalam pelatihan model, dimana algoritma deep learning biasanya berupa jaringan syaraf convolutional dilatih untuk mengenali pola dan karakteristik objek dalam citra.

Setelah diperoleh model yang terlatih, sistem melangkah ke tahap pendeteksian objek. Pada tahap ini model memproses masukan visual secara real-time untuk mengidentifikasi dan menandai letak rintangan pada setiap frame video. Bersamaan dengan itu data objek yang terdeteksi dikirim ke modul pengolahan data objek yang berfungsi memperkirakan jarak antara UGV dan rintangan melalui pemanfaatan data kedalaman. Proses ini sangat penting untuk menentukan tingkat resiko atau potensi tabrakan dengan rintangan di depannya. Pada akhirnya hasil keseluruhan dari proses tersebut berupa data lokasi dan jarak objek yang terintegrasi. Informasi ini kemudian dimanfaatkan oleh sistem UGV untuk melakukan manuver penghindaran rintangan dan menyusun rute aman selama beroperasi di lapangan. Dengan demikian blok diagram diatas menunjukkan aliran data serta tahapan analitik yang memungkinkan UGV bekerja secara mandiri dalam mengenali, mengkasifikasi dan menghindari rintangan di lingkungan dinamis.

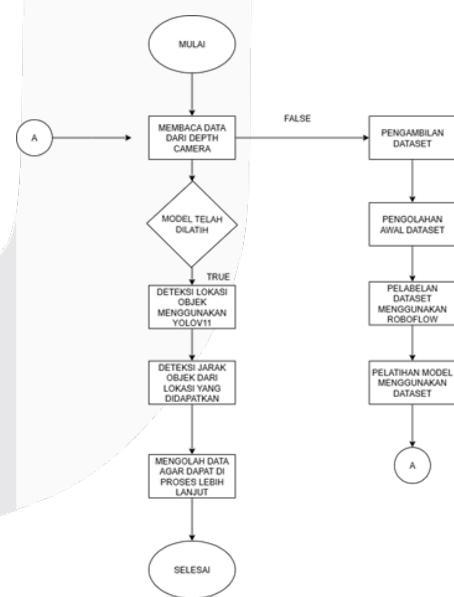
C. Desain Perangkat Keras



GAMBAR 4 (Desain Perangkat Keras Sistem)

Sistem navigasi yang ditunjukkan pada Gambar 3.3 berfokuskan pada bagian proses dari keseluruhan sistem UGV yang akan dibangun, secara singkat berdasarkan Gambar 3, sistem navigasi mendapatkan masukan berupa data hasil dari pengolahan dan pembacaan dari sensor LiDAR dan kamera kedalaman, dan data perhitungan odometri yang dikirimkan oleh Arduino Mega, kemudian diproses menjadi sistem navigasi yang keluarannya berupa perintah untuk menggerakkan AGV melalui motor driver yang terhubung ke setiap motor DC.

D. Desain Perangkat Lunak



GAMBAR 5 (Diagram Alir Sistem)

Diagram alir ini menunjukkan langkah-langkah kerja sistem deteksi objek berbasis kamera kedalaman yang dirancang untuk kendaraan tanpa awak (Unmanned Ground Vehicle) saat beroperasi di lingkungan luar. Proses diawali dengan membaca data yang dikirim oleh depth camera, yang memberikan informasi berupa citra dan kedalaman secara bersamaan. Informasi ini penting untuk memahami posisi dan jarak objek di sekitar kendaraan.

Setelah data berhasil diakses, sistem akan mengecek apakah model deteksi objek sudah dilatih sebelumnya. Jika model belum siap, maka proses dilanjutkan dengan menyiapkan dataset, yang mencakup tahap pengambilan gambar dari lingkungan nyata, kemudian dilakukan pengolahan awal untuk memastikan kualitas data. Selanjutnya, setiap gambar diberi label secara manual menggunakan platform Roboflow, dengan cara menandai objek target (pohon) menggunakan bounding box.

Dataset yang telah diberi label ini kemudian digunakan untuk melatih model deteksi objek. Proses pelatihan dilakukan menggunakan algoritma YOLOv11 dengan bantuan GPU pada layanan cloud seperti Google Colab. Untuk meningkatkan kinerja model, diterapkan teknik augmentasi data agar model dapat mengenali objek dalam berbagai kondisi pencahayaan dan posisi. Setelah proses pelatihan selesai, sistem akan kembali ke alur utama.

Tahap berikutnya adalah menggunakan model yang telah dilatih untuk mendeteksi posisi objek secara langsung dari data visual yang diterima. YOLOv11 akan memproses gambar dan menentukan lokasi objek melalui prediksi bounding box. Setelah lokasi objek berhasil dikenali, sistem akan menghitung jarak antara objek dan UGV menggunakan informasi kedalaman dari kamera. Untuk memastikan akurasi, sistem hanya mengambil bagian tengah dari bounding box sebagai dasar perhitungan, sehingga bisa mengurangi gangguan dari latar belakang.

Data lokasi dan jarak objek yang telah diperoleh kemudian diproses lebih lanjut agar bisa digunakan dalam aplikasi navigasi atau fungsi lainnya. Informasi ini dapat ditampilkan dalam bentuk visual, file data, atau digunakan langsung oleh sistem penggerak kendaraan.

Rangkaian proses ini ditutup dengan status akhir yaitu "SELESAI", yang menandakan bahwa semua tahapan, mulai dari pengumpulan data hingga pemrosesan akhir, telah berhasil dijalankan secara menyeluruh dan sistem siap untuk digunakan dalam pengujian lapangan.

E. Implementasi Sistem

Implementasi sistem bertujuan untuk mengintegrasikan seluruh komponen yang telah dirancang dan model yang telah dilatih untuk menciptakan sebuah sistem deteksi objek dan pengukuran jarak yang berfungsi secara real-time. Sistem ini dijalankan pada perangkat Mini PC Intel NUC yang terhubung dengan depth camera Intel RealSense. Lingkungan perangkat lunak utama yang digunakan adalah Python, dengan memanfaatkan library OpenCV untuk pemrosesan citra dan PyTorch untuk menjalankan inferensi model YOLOv11.

Proses kerja sistem secara real-time dimulai saat depth camera menangkap frame RGB dan frame kedalaman secara simultan. Frame RGB yang ditangkap kemudian diolah oleh model YOLOv11 (best.pt) yang telah dilatih untuk mendeteksi objek pohon. Apabila sebuah pohon berhasil diidentifikasi, model akan menghasilkan koordinat bounding box di sekitarnya. Selanjutnya, sistem memanfaatkan informasi spasial dari bounding box ini untuk mengakses data pada frame kedalaman yang bersesuaian. Jarak objek

dihitung dengan mengambil nilai median dari piksel-piksel kedalaman yang berada di dalam area pusat bounding box, yang didefinisikan oleh parameter `center_ratio` 0.3 sebagaimana telah ditetapkan pada sub-bab 4.3. Hasil akhir dari keseluruhan rangkaian proses ini divisualisasikan secara langsung di layar, menampilkan frame RGB asli yang telah ditambahi anotasi berupa bounding box, label kelas, dan informasi jarak terukur.

Penentuan jarak objek secara akurat merupakan fungsi krusial dari sistem ini. Setelah model YOLOv11 berhasil mendeteksi objek pohon dan menghasilkan bounding box, langkah selanjutnya adalah mengukur jarak UGV ke objek tersebut menggunakan data dari depth camera Intel RealSense. Untuk mendapatkan pengukuran yang stabil dan andal, jarak tidak dihitung dari satu piksel tunggal, yang bisa jadi merupakan noise atau titik tepi objek. Sebagai gantinya, jarak diestimasi dari area tengah (`robust center`) di dalam bounding box.

Metode ini menggunakan parameter `center_ratio` untuk menentukan seberapa besar area tengah dari bounding box yang akan dijadikan sampel untuk pengambilan data kedalaman. Nilai `center_ratio` 0.1 berarti hanya 10% x 10% area di pusat bounding box yang digunakan, sedangkan nilai 0.9 berarti 90% x 90% area yang digunakan. Untuk menemukan nilai `center_ratio` yang optimal, dilakukan pengujian dengan menempatkan objek pada beberapa jarak yang telah diketahui (1.45 m, 2.5 m, dan 6 m) dan mengukur jarak terdeteksi menggunakan nilai `center_ratio` yang berbeda. Hasil pengujian disajikan pada tabel di bawah ini.

TABEL 1
Tabel Hasil Pengujian Nilai Center Ratio

Jarak Sebenarnya (m)	Nilai Center_Ratio	Jarak Terdeteksi (m)
1.45	0.1	1.2
	0.3	1.41
	0.6	1.87
	0.9	2.3
2.5	0.1	2.4
	0.3	2.56
	0.6	2.7
	0.9	2.94
6	0.1	5.95
	0.3	5.99
	0.6	6.21
	0.9	6.46

Berdasarkan tabel di atas, dapat dianalisis bahwa nilai `center_ratio` yang lebih kecil (0.1 dan 0.3) cenderung menghasilkan pengukuran jarak yang lebih akurat. Hal ini disebabkan area sampling yang lebih kecil lebih mungkin hanya mencakup piksel dari objek itu sendiri dan menghindari piksel dari latar belakang atau objek lain di sekitarnya. Sebaliknya, `center_ratio` yang besar (0.6 dan 0.9) sering kali menghasilkan nilai jarak yang lebih besar dari jarak sebenarnya karena ikut menyertakan piksel latar belakang dalam perhitungannya.

Dengan mempertimbangkan akurasi di berbagai rentang jarak, nilai `center_ratio` 0.3 dipilih sebagai parameter optimal

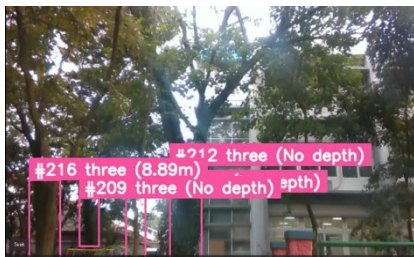
untuk implementasi sistem. Nilai ini memberikan keseimbangan terbaik antara akurasi pengukuran dan kekokohan terhadap bentuk objek pohon yang tidak beraturan.

III. HASIL DAN PEMBAHASAN

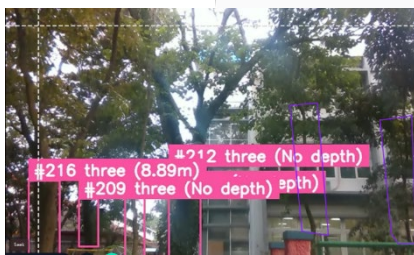
A. Evaluasi Kinerja Sistem Deteksi Objek Secara Real Time di Lingkungan Outdoor

Pengujian sistem deteksi objek secara real-time dilakukan untuk mengevaluasi performa sistem dalam kondisi nyata yang tidak terkendali, yaitu di lingkungan outdoor pada area sekitar Gedung P. Tujuan dari pengujian ini adalah untuk mengetahui seberapa baik sistem mampu mendeteksi pohon sebagai objek target ketika dijalankan langsung di lapangan, dengan pencahayaan alami yang bervariasi tergantung waktu pengujian.

Sistem diuji sebanyak lima puluh kali pada waktu yang acak, dengan pencatatan jumlah pohon aktual yang ada di lokasi dan jumlah pohon yang berhasil dideteksi oleh sistem. Berdasarkan hasil pengujian, dari total 214 pohon yang seharusnya terdeteksi, sistem berhasil mengenali sebanyak 161 pohon, menghasilkan tingkat keberhasilan deteksi sebesar 75,23%. Nilai ini menunjukkan bahwa sistem memiliki performa yang cukup baik untuk penggunaan langsung di luar ruangan, meskipun masih terdapat beberapa keterbatasan.



GAMBAR 6 Pengujian Sistem Realtime



GAMBAR 7 Pengujian Aktual

TABEL 2

Tabel hasil pengujian deteksi objek secara realtime

Uji Ke	Jumlah Pohon Aktual	Jumlah Pohon Terdeteksi	Akurasi Per Uji (%)
1	6	4	66,70%
2	4	3	75,00%
3	5	3	60,00%

4	7	5	71,40%
5	5	4	80,00%
...			
46	10	10	100%
47	6	5	83,33%
48	4	3	75%
49	4	4	100%
50	5	2	40%
Total	214	161	75,23%

Berdasarkan hasil yang ditampilkan pada tabel, sistem menunjukkan kemampuan yang cukup baik dalam mendeteksi sebagian besar pohon secara langsung di lingkungan nyata. Namun, masih ditemukan sejumlah kasus false negative, yaitu kondisi di mana objek pohon yang sebenarnya ada tidak terdeteksi oleh sistem, meskipun secara visual terlihat jelas. Salah satu penyebab utama dari permasalahan ini adalah pencahayaan lingkungan yang tidak konsisten. Kamera depth yang digunakan belum mendukung fitur auto-exposure secara maksimal, sehingga dalam beberapa situasi, citra yang dihasilkan menjadi terlalu terang (overexposed) atau sebaliknya terlalu gelap (underexposed), tergantung arah dan intensitas cahaya saat pengujian berlangsung.

Meskipun model deteksi YOLOv11n telah dilatih dengan teknik augmentasi yang bervariasi, model tersebut masih mengalami kesulitan dalam mengidentifikasi objek pada kondisi pencahayaan ekstrem. Hal ini mengakibatkan penurunan performa sistem, terutama pada area yang berada dalam bayangan atau pencahayaan yang tidak merata. Temuan ini menunjukkan bahwa sistem visi komputer yang dikembangkan masih perlu ditingkatkan, khususnya dalam hal kemampuannya beradaptasi dengan kondisi lingkungan luar ruangan yang tidak stabil.

Sebagai langkah validasi, dilakukan perbandingan antara kondisi nyata di lapangan dan hasil visual dari deteksi sistem. Tujuannya adalah untuk mengonfirmasi keberhasilan maupun kegagalan sistem dalam mengenali objek, serta memastikan bahwa seluruh data pengujian tercatat secara akurat dan dapat dipertanggungjawabkan.

Dengan capaian akurasi sebesar 75,23%, sistem ini sudah menunjukkan potensi yang menjanjikan dalam pendeteksian objek di lingkungan outdoor pada UGV. Meski demikian, peningkatan tetap diperlukan, khususnya dalam hal ketahanan terhadap perubahan pencahayaan, agar sistem dapat beroperasi dengan lebih andal di berbagai situasi lingkungan outdoor.

IV. KESIMPULAN

Penelitian ini dilaksanakan dengan tujuan untuk merancang dan mengimplementasikan sistem pendeteksi objek berupa pohon serta pengukuran jaraknya secara otomatis dan real-time, dengan memanfaatkan depth camera Intel RealSense D435i dan algoritma yang diterapkan. Sistem ini dirancang untuk mendukung mendeteksi rintangan berupa pohon pada lingkungan outdoor. Berdasarkan hasil perancangan dan pengujian, sistem dapat berfungsi sesuai yang diharapkan.

Model YOLOv11n yang dilatih menggunakan dataset hasil anotasi dari Roboflow menunjukkan performa yang baik dalam pengujian di outdoor, dengan nilai precision sebesar 88,1%, recall sebesar 85,6%, dan mAP 0.5 sebesar 90,5%. Selain itu, sistem mampu melakukan inferensi dengan kecepatan rata-rata 1,1 milidetik per gambar, sehingga telah memenuhi kriteria untuk pengolahan data secara real-time.

Estimasi jarak objek dilakukan dengan pendekatan robust center dari data kedalaman, dengan memusatkan pengambilan nilai pada bagian tengah bounding box. Penggunaan parameter center_ratio sebesar 0,3 terbukti memberikan hasil estimasi yang stabil dan lebih tahan terhadap gangguan dari lingkungan sekitar. Untuk menguji kinerja sistem secara nyata, pengujian dilakukan di area luar ruangan, tepatnya di sekitar Gedung P. Dari total 214 pohon aktual yang terdapat pada area uji, sistem berhasil mendeteksi 161 pohon, sehingga diperoleh tingkat keberhasilan deteksi sebesar 75,23%. Hasil ini menunjukkan bahwa sistem telah bekerja dengan cukup baik dalam kondisi lingkungan luar yang sebenarnya.

Meskipun demikian, sistem masih mengalami kendala pada kondisi pencahayaan ekstrem. Kamera yang digunakan belum sepenuhnya mampu menyesuaikan eksposur secara otomatis, sehingga dalam beberapa kasus, gambar yang dihasilkan menjadi terlalu terang (overexposed) atau terlalu gelap (underexposed). Hal ini menghambat kemampuan model dalam mengenali ciri visual dari objek, terutama saat kontras cahaya terlalu tinggi, dan menyebabkan terjadinya kesalahan deteksi atau false negative. Untuk memastikan keakuratan data, dilakukan validasi hasil dengan membandingkan gambar aktual di lapangan dengan keluaran sistem secara visual. Secara keseluruhan, sistem yang dirancang telah memenuhi tujuan utama penelitian dan menunjukkan potensi untuk terus dikembangkan dalam pengimplementasiannya untuk mengklasifikasi dan menentukan jarak dari rintangan yang ada.

REFERENSI

- [1] F. Islam, M. M. Nabi, and J. E. Ball, "Off-Road Detection Analysis for Autonomous Ground Vehicles: A Review," Nov. 01, 2022, *MDPI*. doi: 10.3390/s22218463.
- [2] C. Godard, O. Mac Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE International Conference on Computer Vision*, Institute of Electrical and Electronics Engineers Inc., Oct. 2019, pp. 3827–3837. doi: 10.1109/ICCV.2019.00393.
- [3] T. Pereira, T. Gameiro, J. Pedro, C. Viegas, and N. M. F. Ferreira, "Vision System for a Forestry Navigation Machine," *Sensors*, vol. 24, no. 5, Mar. 2024, doi: 10.3390/s24051475.
- [4] C. Ersü, E. Petlenkov, and K. Janson, "A Systematic Review of Cutting-Edge Radar Technologies: Applications for Unmanned Ground Vehicles (UGVs)," *Sensors*, vol. 24, no. 23, 2024, doi: 10.3390/s24237807.
- [5] Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, "Object Detection with Deep Learning: A Review," *IEEE Trans Neural Netw Learn Syst*, vol. 30, no. 11, pp. 3212–3232, 2019, doi: 10.1109/TNNLS.2018.2876865.
- [6] A. Y. V. Mulawarman Munsyir, S.E., S.SI., M.Kom, Harry Setya Hadi, S.Kom, M.Kom, Tata Sumitra, S.Kom., M.Kom, Ita Arfyanti, S. Kom, M. M, *Algoritma dan Pemrograman: Pendekatan Komprehensif*. [Online]. Available: <https://journal.yayasanpad.org/index.php/ypadbook/article/view/125>
- [7] J. Nurhakiki *et al.*, "Studi Kepustakaan: Pengenalan 4 Algoritma Pada Pembelajaran Deep Learning Beserta Implikasinya," *Jurnal Pendidikan Berkarakter*, no. 1, pp. 270–281, 2024, [Online]. Available: <https://doi.org/10.51903/pendekar.v2i1.598>
- [8] A. Y. W. dan R. S. I Wayan Suartika E. P, "JURNAL TEKNIK ITS Vol. 5, No. 1, (2016) ISSN: 2337-3539 (2301-9271 Print)," *Klasifikasi Citra Menggunakan Convolutional Neural Network (Cnn) pada Caltech 101*, vol. Vol. 5, No, no. 1, pp. 1–2, 2016.
- [9] F. Muhamad Zulkarnaen, "Sistem Deteksi Objek Manusia Menggunakan Algoritma Yolov8 Berbasis Kamera Depth Sensor (Studi Kasus: Cv. Ateri Global Teknologi)," 2024.
- [10] T. Nur, Huzaeni, and M. Khadafi, "Implementasi Metode Object Detection Dengan Algoritma Yolo (You Only Look Once) Untuk," *Jurnal Teknologi Rekayasa Informasi dan Komputer*, vol. 6, no. 2, pp. 28–33, 2023.
- [11] F. Chen, Y. Zhang, L. Fu, R. Hua, Q. Zhang, and S. Bi, "A Comparative Review of the Next-Generation YOLO Models: YOLOv10 and YOLO11".