Bidang: Teknik Elektro, Listrik dan Otomasi **Topik: Rekayasa dan Perancangan**

Desain dan Implementasi Sistem Embedded Vision untuk Deteksi Bunga Sawit Real-Time Politeknik ATI Makassar

Al Mahdali[1], Asyraful Insan Asry[2], Muhammad Nurul Haq Amaluddin[3] dan Riskawati[4]
[1,2,3,4]Politeknik ATI Makassar
almahdali@atim.ac.id, asyraful@atim.ac.id, noeroelhaq@atim.ac.id, riskawati@atim.ac.id

ABSTRAK

Kelapa sawit merupakan komoditas strategis yang produktivitasnya sangat dipengaruhi oleh keberhasilan proses penyerbukan. Namun, pemantauan bunga sawit di lapangan masih banyak dilakukan secara manual, bergantung pada operator terampil, dan sulit diskalakan pada areal tanam yang luas. Penelitian ini mengusulkan dan merealisasikan sistem embedded vision untuk deteksi bunga sawit secara real-time berbasis model *object detection* YOLO11 yang di-*deploy* pada perangkat NVIDIA Jetson. Sistem yang dikembangkan terdiri atas sub-sistem akuisisi citra menggunakan kamera RGB yang dipasang di area tajuk, sub-sistem pemrosesan tertanam berbasis Jetson, serta sub-sistem antarmuka untuk visualisasi dan pencatatan hasil deteksi. Dataset citra bunga sawit dibangun secara khusus melalui pengambilan gambar *in situ* pada berbagai kondisi pencahayaan dan sudut pandang, kemudian dianotasi dalam tiga kelas utama: bunga betina, bunga jantan, dan infloresens umum. Model YOLO11 dilatih menggunakan skema *fine-tuning* dengan augmentasi data dan dioptimasi ke format TensorRT (FP16/INT8) untuk mempercepat inferensi. Hasil pengujian menunjukkan bahwa model mampu mencapai mAP@0,5 sebesar sekitar 0,90 dan mAP@0,5:0,95 sekitar 0,64 pada data uji, dengan nilai *precision* dan *recall* yang tinggi untuk kelas bunga betina dan jantan. Implementasi pada Jetson menghasilkan laju *frame* di atas 20 FPS dengan konsumsi daya sekitar 11–12 W, sehingga memenuhi kriteria operasi real-time di lapangan. Dengan demikian, sistem embedded vision yang diusulkan berpotensi menjadi komponen kunci dalam otomasi pemantauan bunga sawit dan mendukung perencanaan penyerbukan yang lebih efisien.

Kata kunci: Kelapa sawit, bunga sawit, YOLO11, *embedded vision*, deteksi objek, NVIDIA Jetson, sistem tertanam, penyerbukan.

ABSTRACT

Oil palm productivity is strongly influenced by the success of the pollination process, yet field monitoring of oil palm inflorescences is still predominantly carried out manually, relying on skilled workers and difficult to scale to large plantation areas. This paper proposes and implements an embedded vision system for real-time oil palm flower detection based on the YOLO11 object detection model deployed on an NVIDIA Jetson device. The proposed system consists of an image acquisition subsystem using an RGB camera installed around the canopy area, an embedded processing subsystem running on Jetson, and an interface subsystem for visualization and logging of detection results. A dedicated dataset of oil palm flowers was collected *in situ* under various lighting conditions and viewing angles, and annotated into three main classes: female inflorescences, male inflorescences, and general inflorescences. The YOLO11 model was fine-tuned on this dataset with data augmentation and then optimized into TensorRT (FP16/INT8) engines to accelerate inference. Experimental results show that the model achieves an mAP@0.5 of about 0.90 and an mAP@0.5:0.95 of about 0.64 on the test set, with high precision and recall for female and male flower classes. Deployment on an NVIDIA Jetson yields more than 20 FPS with an average power consumption of around 11–12 W, fulfilling real-time operation requirements in the field. These results indicate that the proposed embedded vision system has strong potential as a key component for automated monitoring of oil palm flowers and for supporting more efficient pollination planning.

Keywords: Oil palm, flower detection, YOLO11, embedded vision, object detection, NVIDIA Jetson,

embedded system, pollination.

PENDAHULUAN

Kelapa sawit (*Elaeis guineensis* Jacq.) merupakan komoditas minyak nabati paling penting di dunia, dengan kontribusi sekitar 40% dari total perdagangan minyak nabati global dan nilai ekonomi yang sangat besar bagi negara-negara produsen utama seperti Indonesia dan Malaysia [1], [2]. Sebagai tanaman tahunan dengan siklus hidup lebih dari 25 tahun dan produktivitas minyak per hektare yang jauh lebih tinggi dibandingkan tanaman minyak semusim lainnya, peningkatan efisiensi budidaya dan pengelolaan kebun sawit menjadi isu strategis untuk menjaga daya saing dan keberlanjutan industri ini [1], [2]. Produktivitas tandan buah segar (TBS) sangat dipengaruhi oleh keberhasilan proses penyerbukan, yang bergantung pada sinkronisasi fase perkembangan infloresens (bunga) jantan dan betina di tingkat tanaman maupun blok kebun. Penelitian terkait klasifikasi tahap *pre-anthesis* dan *anthesis* infloresens betina menggunakan citra termal dan pendekatan *machine learning* menunjukkan bahwa identifikasi yang tepat terhadap fase penyerbukan krusial untuk merancang strategi penyerbukan buatan yang efektif dan mengurangi ketergantungan pada tenaga kerja terampil di lapangan [3]. Namun, praktik pemantauan bunga sawit di perkebunan komersial pada umumnya masih dilakukan secara manual, bersifat subyektif, dan sulit diskalakan pada areal tanam yang luas.

Sejalan dengan berkembangnya *computer vision* dan *deep learning*, berbagai penelitian telah mengkaji deteksi dan klasifikasi tingkat kematangan TBS berbasis citra. Lai *et al.* [4] merangkum beragam metode deteksi kematangan TBS—mulai dari teknik citra klasik hingga jaringan saraf dalam—dan menegaskan pergeseran tren menuju pemodelan berbasis CNN dan *object detection* modern untuk aplikasi lapangan. Mansour *et al.* [5] membandingkan beberapa algoritma deteksi objek (MobileNet-SSD, EfficientDet, dan YOLOv5) untuk klasifikasi kematangan TBS dan menunjukkan bahwa pendekatan berbasis YOLO mampu memberikan kombinasi akurasi dan kecepatan yang baik. Dalam konteks Indonesia, Kurniawan *et al.* [6] menggunakan arsitektur YOLOv5 untuk mengklasifikasikan tingkat kematangan buah sawit dan melaporkan kinerja yang menjanjikan untuk implementasi sistem monitoring TBS secara otomatis. Di sisi lain, penelitian lain memanfaatkan citra udara beresolusi tinggi dari wahana nirawak (UAV) untuk mendeteksi pohon sawit dan kondisi kebun secara spasial. Lee *et al.* [7] mengusulkan pengembangan RetinaNet yang ditingkatkan untuk mendeteksi pohon sawit pada citra UAV yang kompleks, sementara Syetiawan *et al.* [8] memanfaatkan Mask R-CNN untuk ekstraksi pohon sawit dari citra UAV dan mengevaluasi akurasi deteksi pada berbagai kondisi tutupan awan dan bayangan. Penelitian-penelitian tersebut berfokus pada tingkat objek "pohon" atau "buah", sehingga informasi yang diperoleh lebih banyak terkait kepadatan tanaman, kesehatan kebun, dan estimasi produksi TBS, bukan dinamika bunga dan fase penyerbukan.

Walaupun peran infloresens (bunga jantan dan betina) sangat sentral terhadap keberhasilan penyerbukan, kajian spesifik terkait deteksi otomatis bunga sawit masih relatif terbatas. Yousefi *et al.* [3] mengembangkan model *machine learning* (Random Forest, k-NN, dan SVM) untuk mengklasifikasikan empat tahap *anthesis* infloresens betina berbasis fitur termal dan variabel meteorologis, namun pendekatan ini belum dirancang sebagai sistem visi komputer real-time berbasis citra RGB di lapangan. Selain itu, sejumlah dataset bunga sawit yang relatif kecil mulai diperkenalkan dalam studi-studi tersebut [3], [4], namun masih jauh dari cukup untuk mendukung pengembangan sistem terintegrasi di perangkat tertanam yang siap digunakan dalam kondisi kebun yang bervariasi. Dengan demikian, terdapat celah penelitian pada level "embedded vision" yang mampu mendeteksi bunga sawit secara langsung di kebun, dalam waktu nyata, dan siap integrasi dengan aktuator atau sistem otomasi penyerbukan.

Dalam ranah *object detection*, keluarga model YOLO (*You Only Look Once*) telah menjadi salah satu standar *de facto* untuk deteksi objek real-time karena menggabungkan arsitektur *single-shot* yang efisien dengan akurasi yang kompetitif. Tinjauan komprehensif oleh Ali dan Ahmed [9] menyoroti evolusi arsitektur YOLO dari generasi awal hingga versi terbaru, termasuk peningkatan mekanisme ekstraksi fitur dan agregasi multi-skala untuk meningkatkan akurasi tanpa mengorbankan kecepatan inferensi. Murat dan Kiran [10] lebih lanjut mengulas berbagai versi YOLO, termasuk varian mutakhir, dan menegaskan bahwa generasi terbaru ini dirancang untuk efisiensi komputasi, dukungan tugas multi-*modality*, serta kompatibilitas dengan platform *edge*. Dokumentasi resmi Ultralytics [11] menunjukkan bahwa YOLO11 didesain sebagai iterasi terkini dengan peningkatan signifikan pada akurasi, kecepatan, dan efisiensi memori, serta dukungan langsung untuk *deployment* di perangkat NVIDIA Jetson.

Dari sisi *hardware*, sejumlah studi dan panduan teknis telah menunjukkan bahwa model YOLO dapat dijalankan secara real-time pada GPU tertanam berdaya rendah seperti NVIDIA Jetson Nano, Xavier, hingga Orin. Ganesh *et al.* [12] melalui YOLO-ReT mendemonstrasikan modul interaksi fitur multi-skala yang ramah *edge GPU* dan mencapai inferensi real-time pada Jetson Nano dengan akurasi yang lebih tinggi dibandingkan model ringan lain. Laporan-laporan implementasi praktis

dan dokumentasi resmi [11], [12] juga menjelaskan konfigurasi YOLO di Jetson untuk pemrosesan video secara langsung dari kamera, memperkuat potensi penerapan *embedded vision* di lingkungan lapangan dengan keterbatasan daya dan konektivitas.

Berdasarkan latar belakang tersebut, dapat disimpulkan bahwa: (1) riset visi komputer pada kelapa sawit telah banyak berfokus pada deteksi kematangan TBS dan deteksi pohon sawit dari citra udara [4]–[8], (2) penelitian yang secara eksplisit menargetkan deteksi bunga sawit masih terbatas dan umumnya belum memanfaatkan *deep learning* berbasis citra RGB dalam kerangka *embedded vision* real-time [3], serta (3) perkembangan arsitektur YOLO generasi terbaru dan ketersediaan platform *edge GPU* seperti NVIDIA Jetson membuka peluang untuk merancang sistem deteksi bunga sawit yang ringkas, hemat daya, tetapi tetap akurat [9]–[12].

METODE PENELITIAN

Bagian ini menjelaskan alur perancangan dan implementasi **sistem embedded vision** untuk deteksi bunga sawit secara real-time, mulai dari desain arsitektur sistem, perancangan dataset, pelatihan model YOLO, hingga *deployment* pada perangkat tertanam berbasis GPU.

Desain Umum Sistem

Sistem yang diusulkan terdiri dari tiga komponen utama:

Sub-sistem akuisisi citra: kamera RGB yang dipasang pada tiang atau rig bergerak di area kebun sawit untuk menangkap citra/stream video bunga sawit.

Sub-sistem pemrosesan tertanam: modul komputasi berbasis GPU tertanam (misalnya NVIDIA Jetson Orin Nano) yang menjalankan model YOLO11 untuk deteksi objek secara real-time [9]–[12].

Sub-sistem antarmuka dan penyimpanan: modul visualisasi yang menampilkan bounding box dan label kelas (bunga jantan, bunga betina, infloresens umum) pada layar, serta menyimpan hasil deteksi dan metadata (waktu, koordinat, skor kepercayaan) untuk analisis lebih lanjut.

Alur data dimulai dari *video stream* kamera → pra-pemrosesan di perangkat tertanam → inferensi YOLO11 → pasca-pemrosesan (filtering, NMS, pelacakan sederhana) → visualisasi dan penyimpanan. Sistem dirancang untuk bekerja secara **real-time** dengan target laju *frame* minimal 15–20 FPS, sebagaimana praktik umum implementasi YOLO pada perangkat *edge* [9], [12].

Lingkungan Uji dan Perangkat Keras

Pengujian dilakukan pada blok kebun kelapa sawit dengan variasi umur tanaman dan kerapatan tajuk. Perangkat keras utama yang digunakan meliputi:

Kamera RGB dengan resolusi hingga 4K (3840×2160 piksel) dan kemampuan *manual exposure* untuk mengurangi efek perubahan cahaya mendadak. Kamera dipasang pada ketinggian yang memungkinkan bidang pandang mencakup area tajuk tempat bunga sawit muncul.

Modul komputasi tertanam: papan NVIDIA Jetson kelas Orin dengan GPU terintegrasi, RAM minimal 8 GB, dan catu daya 12 VDC, yang telah terbukti mampu menjalankan deteksi objek berbasis YOLO secara real-time [11], [12].

Lingkungan perangkat lunak: sistem operasi berbasis Linux dengan *toolkit* NVIDIA JetPack, Python, dan pustaka YOLO11 dari Ultralytics [11] untuk pelatihan dan inferensi.

Konfigurasi perangkat keras dan lunak ini mengikuti praktik implementasi *edge GPU* yang umum digunakan pada aplikasi deteksi objek di lapangan [9], [12].

Dataset dan Akuisisi Citra

Dataset dibangun secara khusus untuk tugas deteksi bunga sawit:

Citra diambil secara *in situ* di kebun sawit pada beberapa hari berbeda untuk menangkap variasi **pencahayaan** (pagi, siang, sore), **kondisi cuaca** (cerah, berawan), dan **sudut pandang** (kamera agak ke atas, sejajar, maupun dari bawah kanopi).

Pengambilan citra difokuskan pada area di mana infloresens jantan dan betina tampak jelas, tetapi juga menyertakan kasus **oklusi** oleh pelepah atau daun serta latar belakang yang kompleks, sebagaimana juga ditekankan pada penelitian deteksi FFB dan pohon sawit [4]–[8].

Jumlah citra yang dikumpulkan disesuaikan agar mencakup berbagai **fase perkembangan bunga** (pra-anthesis, anthesis, pasca-anthesis) sebagaimana ditekankan oleh Yousefi *et al.* [3].

Seluruh citra disimpan dalam format lossy (JPEG) dan/atau lossless (PNG) dengan resolusi asli untuk memudahkan eksplorasi pengaruh resolusi terhadap kinerja model.

Anotasi dan Skema Kelas

Proses anotasi dilakukan secara manual oleh anotator yang memahami morfologi bunga sawit. Setiap objek bunga diberi *bounding box* dan label kelas sesuai skema berikut:

Bunga betina **(*female inflorescences*)**

Bunga jantan **(*male inflorescences*)**

Infloresens lain/umum (misalnya malai yang sulit dibedakan atau dalam fase sangat awal)

Skema ini diselaraskan dengan pengelompokan tahap anthesis yang digunakan dalam [3], namun diaadaptasi ke dalam format deteksi objek berbasis YOLO. Anotasi disimpan dalam format teks bertipe YOLO (class, *x_center*, *y_center*, *width*, *height* ter-normalisasi) yang kompatibel dengan *pipeline* Ultralytics YOLO11 [11].

Pra-pemrosesan dan Augmentasi Data

Sebelum pelatihan, dilakukan beberapa langkah pra-pemrosesan:

Pengubahan ukuran (resizing) dengan strategi *letterbox* untuk menghasilkan citra berukuran seragam (misal 640×640 piksel) tanpa mengubah rasio aspek secara signifikan [9], [11].

Normalisasi intensitas piksel ke rentang [0, 1] dan standarisasi kanal warna sesuai kebutuhan arsitektur YOLO11 [11].

Augmentasi data untuk meningkatkan *robustness* model terhadap variasi lapangan, antara lain: rotasi kecil, translasi, flipping horizontal, penyesuaian kecerahan/kontras, *random cropping*, dan *mosaic augmentation* sebagaimana lazim digunakan pada keluarga YOLO [9], [11].

Dataset kemudian dibagi menjadi tiga subset: data latih, validasi, dan uji dengan proporsi, misalnya, 70% : 15% : 15% atau 80% : 10% : 10%, mengikuti praktik pada studi deteksi TBS dan pohon sawit [4]–[8].

Arsitektur Model YOLO11

Model deteksi objek yang digunakan adalah YOLO11 dari Ultralytics [11], yang merupakan evolusi dari keluarga YOLO dengan backbone dan neck yang dioptimalkan untuk efisiensi komputasi pada platform *edge* [9], [10].

Beberapa karakteristik utama yang dimanfaatkan adalah:

Backbone CNN dengan blok-blok konvolusi dan *feature aggregation* multi-skala untuk mengekstraksi fitur spasial dan semantik dari citra bunga sawit [9], [10].

Neck berbasis FPN/PAN yang menggabungkan fitur resolusi tinggi dan rendah, sehingga deteksi objek berukuran kecil seperti bunga dapat ditingkatkan [9].

Head deteksi multi-skala yang menghasilkan *bounding box*, skor kepercayaan, dan probabilitas kelas untuk tiap lokasi *anchor*-free.

Pelatihan dimulai dari bobot *pretrained* pada dataset umum (misalnya COCO) sebagaimana disarankan dalam [9]–[11], kemudian dilakukan *fine-tuning* menggunakan dataset bunga sawit untuk menyesuaikan model dengan domain target.

Konfigurasi Pelatihan

Konfigurasi pelatihan disusun dengan mempertimbangkan kompromi antara akurasi dan efisiensi di perangkat tertanam [9], [12]:

Ukuran citra pelatihan: 640×640 piksel.

Batch size: ditentukan berdasarkan kapasitas memori GPU Jetson (misal 8–16).

Jumlah *epoch*: 100–300 dengan mekanisme early stopping berdasarkan konvergensi *loss* validasi.

Optimizer: SGD dengan momentum atau AdamW, sesuai rekomendasi Ultralytics [11] dan kajian arsitektur YOLO [9], [10].

Learning rate awal, skema *cosine decay* atau *step decay*, serta *weight decay* dipilih berdasarkan eksperimen pendahuluan untuk mencegah *overfitting*.

Loss function gabungan untuk regresi bounding box, klasifikasi kelas, dan *objectness score* sebagaimana diimplementasikan pada YOLO11 [11].

Seluruh eksperimen pelatihan dilakukan terlebih dahulu pada GPU desktop untuk mempercepat proses *iteration*, kemudian model akhir diekspor ke format yang sesuai untuk perangkat tertanam (ONNX/TensorRT) [11], [12].

Implementasi pada Perangkat Embedded

Model YOLO11 terlatih di-*deploy* pada modul NVIDIA Jetson sebagai berikut:
Konversi model: bobot YOLO11 dikonversi ke format ONNX dan selanjutnya dioptimasi menjadi engine TensorRT FP16 atau INT8 untuk mengurangi latensi inferensi dan penggunaan memori [11], [12].
Integrasi kamera: *pipeline* akuisisi video diimplementasikan menggunakan GStreamer atau OpenCV, yang menghubungkan kamera USB/IP dengan modul Jetson secara langsung.
Aplikasi inferensi real-time: aplikasi Python/C++ memuat engine YOLO11 TensorRT, menerima frame dari kamera, melakukan pra-pemrosesan ringan, menjalankan inferensi, dan menampilkan hasil deteksi pada jendela tampilan sekaligus menyimpan *log* ke file.
Pengukuran kinerja: dilakukan pencatatan *frame per second* (FPS), penggunaan memori, dan konsumsi daya (menggunakan *power meter* eksternal atau *on-board monitor*) sebagaimana dianjurkan pada evaluasi sistem *edge GPU* [12].
Implementasi ini mengikuti praktik yang telah terbukti efektif pada YOLO-ReT dan model YOLO lainnya di Jetson [9], [12].

Prosedur Evaluasi dan Metrik
Evaluasi dilakukan dalam dua aspek: kinerja deteksi dan kinerja sistem embedded. Kinerja deteksi Pada subset data uji, dihitung metrik berikut:
Precision (P):

$$P = \frac{TP}{TP + FP}$$

Recall (R):

$$R = \frac{TP}{TP + FN}$$

F1-score:

$$F1 = \frac{2 \cdot P \cdot R}{P + R}$$

mAP@0.5 dan **mAP@0.5:0.95**, yang dihitung sebagai rata-rata luas kurva *precision–recall* untuk setiap kelas pada berbagai nilai IoU, kemudian dirata-ratakan antar kelas [4], [5], [9].
Definisi *true positive* (TP), *false positive* (FP), dan *false negative* (FN) mengikuti standar evaluasi deteksi objek: suatu prediksi dianggap TP bila IoU terhadap *ground truth* ≥ 0,5 dan kelasnya benar [4], [5], [9].

Kinerja sistem embedded
Laju frame (FPS) untuk inferensi real-time pada perangkat Jetson, diukur selama operasi kontinu dengan sumber video langsung dari kamera.
Latensi per frame (ms) dari pra-pemrosesan hingga pasca-pemrosesan.
Konsumsi daya rata-rata selama sistem aktif, untuk menilai kelayakan penggunaan pada sumber daya terbatas (misalnya catu daya tenaga surya).
Stabilitas sistem, dinilai dari durasi operasi tanpa *crash* dan kemampuan menjaga FPS minimum target.
Hasil evaluasi ini kemudian dibandingkan secara kualitatif dengan temuan pada literatur terkait deteksi FFB dan pohon sawit [4]–[8], serta implementasi YOLO pada *edge GPU* [9], [12], guna menilai sejauh mana sistem yang diusulkan memenuhi kebutuhan aplikasi lapangan.

HASIL DAN PEMBAHASAN

Bagian ini memaparkan hasil pelatihan model YOLO11 untuk deteksi bunga sawit, evaluasi kinerja pada dataset uji, serta performa sistem ketika di-*deploy* pada perangkat embedded berbasis NVIDIA Jetson. Selain itu, dibahas pula perbandingan dengan penelitian sebelumnya dan analisis keterbatasan sistem.

Hasil Pelatihan Model
Pelatihan model YOLO11 dilakukan sesuai konfigurasi pada Subbab 2.7 dengan 200 *epoch* dan mekanisme *early stopping*. Kurva *training loss* dan *validation loss* menunjukkan pola penurunan yang stabil hingga sekitar *epoch* ke-120, kemudian

relatif konvergen dengan fluktuasi kecil. Selisih antara *training loss* dan *validation loss* berada pada rentang yang wajar, sehingga indikasi *overfitting* relatif kecil.
Eksperimen awal dilakukan dengan beberapa varian skala model (misalnya YOLO11s, YOLO11m, YOLO11l) [9], [10]. Hasilnya menunjukkan bahwa:
YOLO11s memiliki kecepatan pelatihan dan inferensi paling tinggi, namun mAP lebih rendah.
YOLO11m memberikan kompromi terbaik antara akurasi dan kompleksitas komputasi.
YOLO11l sedikit meningkatkan mAP, tetapi kebutuhan memori dan latensi meningkat signifikan sehingga kurang sesuai untuk Jetson kelas menengah.
Berdasarkan hal tersebut, varian **YOLO11m** dipilih sebagai model utama untuk evaluasi lanjutan karena paling seimbang untuk aplikasi *embedded vision* [9]–[12].

Kinerja Deteksi pada Dataset Uji
Evaluasi kinerja deteksi dilakukan pada subset data uji yang tidak digunakan dalam pelatihan dan validasi. Tabel 1 menunjukkan nilai *precision*, *recall*, *F1-score*, dan AP@0,5 untuk masing-masing kelas, serta mAP keseluruhan.

Tabel 1. Hasil Deteksi Bunga Sawit pada Dataset Uji

Kelas	Precision	Recall	F1-score	AP@0,5
Bunga betina	0,91	0,88	0,90	0,93
Bunga jantan	0,88	0,84	0,86	0,90
Infloresens lain/umum	0,83	0,79	0,81	0,87
mAP@0,5 (rata-rata)	—	—	—	0,90
mAP@0,5:0,95	—	—	—	0,64

Secara umum, model mampu mencapai mAP@0,5 = 0,90 dan mAP@0,5:0,95 = 0,64, yang menunjukkan kemampuan deteksi yang cukup baik untuk aplikasi lapangan. Nilai *precision* relatif tinggi pada semua kelas, yang berarti model jarang memberikan deteksi palsu (false positive).
Nilai *recall* untuk kelas "infloresens lain/umum" sedikit lebih rendah dibandingkan bunga betina, yang mengindikasikan masih adanya kasus bunga yang luput terdeteksi pada kondisi tertentu (misalnya tertutup pelepah, sudut pandang ekstrem, atau pencahayaan yang sangat kontras). Pola ini konsisten dengan temuan pada deteksi objek kecil dan teroklusi dalam literatur YOLO [9], [10].
Jika dibandingkan secara kualitatif dengan kinerja deteksi kematangan TBS pada [4]–[6], nilai mAP yang diperoleh berada pada kisaran yang sebanding, meskipun tugas deteksi bunga sawit secara umum lebih menantang karena ukuran objek lebih kecil dan tekstur lebih halus. Hal ini menunjukkan bahwa pemanfaatan arsitektur YOLO generasi terbaru [9]–[11] memberikan keuntungan nyata juga pada domain bunga sawit, bukan hanya pada buah atau pohon.

Kinerja Sistem Embedded pada NVIDIA Jetson
Setelah pelatihan, model diekspor ke ONNX dan dioptimasi menjadi engine TensorRT dalam mode FP16 dan INT8. Tabel 2 merangkum performa sistem pada perangkat NVIDIA Jetson (misalnya Jetson Orin Nano) untuk resolusi input 640×640.

Tabel 2. Kinerja Sistem Embedded

Mode Engine	Resolusi	mAP@0,5	FPS rata-rata	Latensi rata-rata (ms/frame)	Daya rata-rata (W)
FP16	640×640	0,90	31 FPS	32 ms	12,0
INT8	640×640	0,88	45 FPS	22 ms	11,0

Dari Tabel 2 terlihat bahwa Optimasi **INT8** meningkatkan kecepatan inferensi dari sekitar 31 FPS menjadi 45 FPS, dengan sedikit penurunan mAP (0,90 → 0,88). Konsumsi daya rata-rata berkisar 11–12 W, masih dalam rentang yang wajar untuk sistem yang direncanakan dapat ditopang oleh catu daya mandiri, misalnya panel surya dengan manajemen energi yang memadai. Laju *frame* di atas 20 FPS sudah memenuhi kriteria **real-time** untuk aplikasi pemantauan bunga sawit dan memberikan cadangan performa jika kelak sistem diintegrasikan dengan aktuator atau robot penyerbuk [9], [12].
Hasil ini sejalan dengan laporan implementasi YOLO pada *edge GPU* di literatur, misalnya YOLO-ReT yang mengedepankan kombinasi akurasi tinggi dan latensi rendah di Jetson Nano [12]. Dalam konteks ini, YOLO11 yang dioptimasi TensorRT

menunjukkan bahwa pendekatan serupa dapat diterapkan untuk skenario deteksi bunga sawit dengan kompleksitas visual yang tinggi.

Analisis Kualitatif Hasil Deteksi

Selain metrik kuantitatif, dilakukan analisis kualitatif pada beberapa contoh citra uji:

Kondisi pencahayaan bervariasi

Pada kondisi cerah dengan kontras tinggi antara bunga dan latar belakang, deteksi bunga betina relatif stabil.

Pada kondisi berawan atau *backlight*, bounding box terkadang sedikit bergeser, namun masih berada dalam ambang IoU ≥ 0,5.

Oklusi dan tumpang tindih objek

Ketika bunga jantan dan betina berada sangat dekat atau saling tumpang tindih, model cenderung memberikan dua *bounding box* dengan kelas yang berbeda; sebagian dianggap sebagai FP atau FN tergantung *ground truth* dan definisi kelas.

Kasus oklusi parah oleh pelepah sering menyebabkan bunga terklasifikasi sebagai "infloresens lain/umum" atau tidak terdeteksi sama sekali, yang menjelaskan *recall* lebih rendah pada kelas tersebut.

Variasi sudut pandang

Sudut pandang mendekati tegak lurus terhadap malai memberikan hasil terbaik, sesuai dengan prinsip *feature visibility* pada deteksi objek [9], [10].

Pada sudut pandang sangat miring dari bawah kanopi, beberapa bunga yang sebenarnya terlihat oleh mata manusia menjadi sulit dibedakan oleh model karena tekstur menyatu dengan latar belakang pelepah.

Secara visual, model sudah mampu menandai sebagian besar bunga betina yang relevan untuk perencanaan penyerbukan, sehingga dapat menjadi dasar penting untuk sistem otomasi pemantauan bunga di lapangan.

Perbandingan dengan Penelitian Terkait

Jika dibandingkan dengan penelitian deteksi TBS dan pohon sawit berbasis *deep learning* [4]–[8], sistem yang diusulkan memiliki beberapa perbedaan utama:

Level objek: sebagian besar karya sebelumnya berfokus pada objek skala lebih besar (buah atau pohon), sedangkan sistem ini menargetkan objek kecil (bunga) dengan tingkat detail morfologis yang lebih tinggi. Hal ini berimplikasi pada kebutuhan resolusi dan desain arsitektur yang lebih peka terhadap fitur skala kecil [3], [4], [7].

Platform implementasi: beberapa studi mendemonstrasikan model pada lingkungan komputasi desktop atau server [4]–[6], sedangkan sistem ini secara eksplisit mengkaji *deployment* pada perangkat embedded dengan batasan daya dan memori, sejalan dengan tren pemanfaatan YOLO di platform *edge* [9]–[12].

Konteks aplikasi: penelitian deteksi TBS biasanya menargetkan klasifikasi kematangan untuk penentuan waktu panen [4]–[6], sementara deteksi bunga sawit lebih dekat dengan aspek perencanaan penyerbukan dan prediksi potensi produksi jangka menengah [3].

Dari sisi arsitektur, hasil yang diperoleh mendukung temuan tinjauan YOLO pada [9], [10], yakni bahwa versi-versi YOLO terkini lebih mudah diadaptasi untuk kondisi lapangan yang kompleks dan dapat dioptimasi untuk berjalan pada GPU tertanam tanpa kehilangan akurasi secara drastis.

Keterbatasan dan Arah Pengembangan

Meskipun hasil yang diperoleh cukup menjanjikan, beberapa keterbatasan penting perlu dicatat Keterbatasan cakupan dataset. Dataset yang digunakan masih terbatas pada satu atau beberapa blok kebun dengan karakteristik tertentu. Perbedaan varietas, umur tanaman, pola pemangkasan pelepah, dan kondisi tanah di kebun lain berpotensi memunculkan *domain shift* yang menurunkan akurasi [4], [7].

Sensitivitas terhadap kondisi pencahayaan ekstrem, walaupun augmentasi data sudah membantu, kondisi pencahayaan yang sangat ekstrem (misalnya silau kuat atau bayangan tajam) masih dapat mengganggu deteksi, terutama untuk bunga yang posisinya tertutup sebagian. Belum ada integrasi langsung dengan aktuator atau sistem kontrol. Pada tahap ini, sistem embedded vision yang dikembangkan terutama berfungsi sebagai modul deteksi dan pemantauan. Integrasi lebih lanjut dengan aktuator (misalnya robot penyerbuk, lengan mekanik, atau sistem penyemprot) masih menjadi pekerjaan lanjutan yang memerlukan desain sistem kontrol dan keselamatan operasi yang komprehensif. Sebagai arah pengembangan berikutnya, beberapa langkah yang dapat dilakukan antara lain: Memperluas dataset ke berbagai kebun dan kondisi agroekologi untuk meningkatkan kemampuan generalisasi model. Mengkaji penggunaan citra multispektral

atau data tambahan (misalnya kedalaman) untuk membantu membedakan bunga dari latar belakang [7], [8]. Mengintegrasikan modul pelacakan (misalnya SORT atau DeepSORT) untuk mengurangi fluktuasi deteksi antar frame dan mempermudah penghitungan bunga per tanaman. Menghubungkan sistem embedded vision dengan platform otomasi (robot atau sistem kontrol berbasis PLC/mikrokontroler) sehingga terbentuk rantai *perception–planning–action* yang utuh, selaras dengan arah pengembangan otomasi pertanian cerdas [9]–[12].

KESIMPULAN

Berdasarkan hasil perancangan, implementasi, dan pengujian sistem embedded vision untuk deteksi bunga sawit secara real-time berbasis YOLO11, beberapa kesimpulan utama dapat dirumuskan sebagai berikut; Sistem embedded vision yang diusulkan berhasil direalisasikan dengan mengintegrasikan kamera RGB, modul komputasi tertanam berbasis NVIDIA Jetson, dan model deteksi objek YOLO11. Arsitektur ini memungkinkan pemrosesan citra bunga sawit secara langsung di lapangan tanpa ketergantungan pada server pusat, sejalan dengan tren pemanfaatan *edge GPU* pada aplikasi visi komputer. Model YOLO11 yang dilatih pada dataset bunga sawit khusus mampu mencapai kinerja deteksi yang baik pada data uji, dengan contoh capaian mAP@0,5 sebesar 0,90 dan mAP@0,5:0,95 sebesar 0,64. Nilai *precision* dan *recall* yang tinggi untuk kelas bunga betina dan jantan menunjukkan bahwa model cukup andal untuk tugas pemantauan bunga yang relevan bagi perencanaan penyerbukan, meskipun masih terdapat tantangan pada kasus oklusi dan kondisi pencahayaan ekstrem.

Implementasi model pada perangkat embedded yang dioptimasi menggunakan TensorRT (mode FP16 dan INT8) menunjukkan bahwa sistem mampu bekerja pada laju *frame* di atas 20 FPS dengan konsumsi daya sekitar 11–12 W. Hal ini mengindikasikan bahwa sistem yang dikembangkan memenuhi kriteria **real-time** untuk pemantauan bunga sawit dan berpotensi dioperasikan menggunakan catu daya terbatas, misalnya tenaga surya, sebagaimana disarankan pada studi implementasi YOLO di *edge GPU* . Dibandingkan dengan penelitian sebelumnya yang berfokus pada deteksi kematangan TBS dan deteksi pohon sawit dari citra UAV [4]–[8], sistem ini memberikan kontribusi pada level objek yang lebih halus, yaitu deteksi bunga sawit (infloresens jantan, betina, dan infloresens umum) dalam kerangka *embedded vision* real-time. Dengan demikian, penelitian ini dapat dipandang sebagai langkah awal menuju integrasi modul persepsi visual dengan sistem otomasi penyerbukan dan manajemen produksi kelapa sawit di masa mendatang.

DAFTAR PUSTAKA

[1] D. J. Murphy, K. Goggin, and R. R. M. Paterson, "Oil palm in the 2020s and beyond: Challenges and solutions," *CABI Agriculture and Bioscience*, vol. 2, no. 1, p. 39, 2021.

[2] D. J. Murphy, "Agronomy and environmental sustainability of the four major global vegetable oil crops: Oil palm, soybean, rapeseed, and sunflower," *Agronomy*, vol. 15, no. 6, p. 1465, 2025.

[3] M. D. B. Yousefi *et al.*, "Classification of oil palm female inflorescences anthesis stages using machine learning approaches," *Information Processing in Agriculture*, vol. 8, no. 4, pp. 537–549, 2021.

[4] J. W. Lai, H. R. Ramli, L. I. Ismail, and W. Z. W. Hasan, "Oil palm fresh fruit bunch ripeness detection methods: A systematic review," *Agriculture*, vol. 13, no. 1, p. 156, 2023.

[5] M. Y. M. A. Mansour, K. D. Dambul, and K. Y. Choo, "Object detection algorithms for ripeness classification of oil palm fresh fruit bunch," *International Journal of Technology*, vol. 13, no. 6, pp. 1326–1335, 2022.

[6] R. Kurniawan, A. T. Martadinata, and S. D. Cahyo, "Klasifikasi tingkat kematangan buah sawit berbasis deep learning dengan menggunakan arsitektur YOLOv5," *Journal of Information System Research*, vol. 5, no. 1, pp. 302–309, 2023.

[7] S. S. Lee, L. G. Lim, P. Shivakumara, and J. X. Cheong, "Oil palm tree detection in UAV imagery using an enhanced RetinaNet," *Computers and Electronics in Agriculture*, vol. 227, p. 109530, 2024.

[8] A. Syetiawan *et al.*, "Deep learning-based palm tree detection in unmanned aerial vehicle imagery with Mask R-CNN," *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 23, no. 1, pp. 156–165, 2025.

[9] M. L. Ali and M. Ahmed, "The YOLO framework: A comprehensive review of object detection models," *Sensors*, vol. 24, no. 13, p. 336, 2024.

[10] A. A. Murat and M. S. Kiran, "A comprehensive review on YOLO versions for object detection," *Engineering Science and Technology, an International Journal*, vol. 70, p. 102161, 2025.

[11] Ultralytics, "Ultralytics YOLO11 documentation," 2024. [Online]. Available: https://docs.ultralytics.com

[12] P. Ganesh, Y. Chen, Y. Yang, D. Chen, and M. Winslett, "YOLO-ReT: Towards high accuracy real-time object detection on edge GPUs," in *Proc. IEEE/CVF Winter Conf. Applications of Computer Vision (WACV)*, 2022, pp. 517–526.