SKRIPSI

OPTIMASI GERAKAN ROBOT SEPAK BOLA MENGGUNAKAN *KALMAN FILTER* PADA *TRACKING* OBJEK BERBASIS YOLO

Oleh:

Fikri Rivandi
2207112583

PROGRAM STUDI TEKNIK INFORMATIKA S1
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS RIAU
2025

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1. Latar Belakang

Perkembangan teknologi di era modern telah memberikan dampak signifikan pada berbagai aspek kehidupan manusia. Salah satu bidang yang mengalami kemajuan pesat adalah teknologi *artificial intelligence* (AI) dan *machine learning* (ML) (Widodo et al., 2024). Teknologi ini memungkinkan sistem komputer tidak hanya melakukan perhitungan, tetapi juga belajar dari data, mengenali pola, serta mengambil keputusan secara mandiri (Wijaya & Yuniarto, 2024). Salah satu cabang dari AI yang banyak dimanfaatkan adalah *computer vision* (CV). CV merupakan cabang dari kecerdasan buatan yang berfokus pada kemampuan komputer dan sistem untuk mengekstraksi informasi bermakna dari citra digital, video, maupun data visual lainnya. Dalam sistem yang bersifat dinamis, informasi visual ini digunakan untuk mengenali, memantau, serta melacak (*tracking*) posisi objek secara kontinu guna mengambil keputusan atau menjalankan tindakan tertentu secara akurat (Baskoro et al., 2022).

Robotika merupakan bidang interdisipliner yang berfokus pada kajian mengenai robot, di mana ilmu pengetahuan, teknologi, dan rekayasa saling berhubungan. Hasil akhirnya berupa mesin yang dapat menirukan perilaku manusia atau menjalankan instruksi tertentu melalui pemrograman (Hendrik & Awal, 2023). Robot konvensional umumnya didesain untuk beroperasi pada lingkungan yang terstruktur dan terbatas dengan algoritma tugas yang telah diprogram secara statis. Akan tetapi, kenyataannya lingkungan kerja sering bersifat dinamis dan tidak terduga, sehingga dibutuhkan sistem yang lebih adaptif dan fleksibel. Pada titik inilah kecerdasan buatan memiliki peran penting. Dengan memanfaatkan teknologi seperti *machine learning*, *computer vision*, dan kecerdasan buatan lainnya, robot modern tidak hanya mampu mengenali objek, tetapi juga dituntut untuk melakukan pelacakan (*tracking*) posisi objek secara kontinu guna memahami kondisi lingkungan dan mengambil keputusan yang lebih responsif serta mandiri (Ritonga & Hasibuan, 2025).

Salah satu implementasi nyata robotika yang mendapat perhatian khusus adalah robot sepak bola, khususnya dalam Kontes Robot Sepak Bola Indonesia (KRSBI) Beroda. Dalam ajang ini, robot dituntut untuk mampu beroperasi secara mandiri di tengah lingkungan lapangan yang sangat dinamis, di mana kecepatan dan ketepatan dalam melakukan pelacakan (*tracking*) terhadap pergerakan bola menjadi faktor krusial yang menentukan keberhasilan sistem navigasi dan strategi permainan robot.

Universitas Riau (UNRI) memiliki klub robotika bernama *Engineering Robotic Club* (ERC) yang berfokus pada pengembangan robot KRSBI Beroda. Pada tahun 2021 dan 2022, tim ERC UNRI berhasil mencapai tingkat nasional, yang menjadi pencapaian signifikan dalam sejarah keikutsertaannya di KRSBI Beroda. Sejak berpartisipasi pertama kali pada tahun 2019, ERC UNRI telah mengalami berbagai perkembangan signifikan, baik dalam aspek perangkat keras maupun perangkat lunak. Namun, perubahan regulasi sejak tahun 2023 membuat tim harus melakukan penyesuaian ulang agar robot tetap kompetitif dan relevan.

Dalam pertandingan KRSBI Beroda, kemampuan *tracking* bola dan gawang menjadi faktor yang sangat krusial. Awalnya, metode *tracking* objek berbasis HSV (*Hue, Saturation, Value*) digunakan karena ringan dan dapat berjalan *real-time*. Citra dengan format HSV dimanfaatkan dalam penerapan algoritma *image thresholding* guna mendeteksi warna objek. Selanjutnya, hasil dari proses *thresholding* tersebut digunakan dalam algoritma *Convex Hull*, di mana apabila bola berhasil terdeteksi, maka sistem akan menghitung ukuran objek, menentukan titik pusatnya, serta memperoleh koordinat posisi objek (x, y) relatif terhadap posisi robot saat ini. Kelemahan metode HSV adalah ketergantungannya pada kondisi pencahayaan, sehingga akurasi *tracking* menjadi tidak stabil dalam situasi lapangan yang dinamis (Nanda et al., 2023).

Seiring perkembangan teknologi *computer vision*, metode *deep learning* seperti YOLO (*You Only Look Once*) akhirnya digunakan untuk meningkatkan akurasi *tracking* objek pada robot KRSBI Beroda milik ERC UNRI. YOLO bekerja dengan membagi citra ke dalam grid dan memprediksi *bounding box* serta kelas objek secara langsung dalam satu tahap komputasi, sehingga mampu memberikan

tracking real-time dengan akurasi yang tinggi (F. B. Saputra et al., 2023). Secara konsep, metode ini sangat ideal untuk kebutuhan robot sepak bola beroda, yang menuntut respon cepat terhadap pergerakan bola di lapangan.

Namun dalam implementasi YOLO pada robot KRSBI Beroda, masalah pertama yang didapat adalah kendala signifikan terkait keterbatasan sumber daya perangkat keras. Sistem robot saat ini menggunakan laptop ASUS K401U sebagai unit pemrosesan utama, yang hanya memiliki spesifikasi standar seperti prosesor Intel Core i5-7200U, RAM 8GB, serta dukungan GPU Nvidia 940MX. Spesifikasi tersebut belum cukup mumpuni untuk menjalankan YOLO dengan baik. Akibatnya, proses *tracking* sering mengalami penurunan performa, seperti *lag* dan ketidakstabilan *frame rate*, sehingga informasi posisi bola diterima oleh robot dengan jeda waktu. Keterlambatan ini berdampak langsung terhadap kualitas pergerakan robot. Robot dapat terlambat merespons perubahan posisi bola, bergerak tidak stabil, atau bahkan salah memperkirakan arah gerak bola. Dengan kata lain, meskipun YOLO memiliki akurasi tinggi, keterbatasan perangkat keras pada robot KRSBI Beroda menyebabkan efisiensinya menurun dan menjadi salah satu sumber utama masalah dalam sistem navigasi berbasis visi.

Miharja et al. (2025) menyebutkan bahwa metode YOLO adalah salah satu cara efektif dalam *object detection*, tapi tidak cukup efisien untuk sumber daya perangkat keras yang digunakan, karena membutuhkan kapasitas komputasi yang mumpuni. Pada penelitian lainnya juga menyebutkan bahwa menggunakan perangkat *Single Board Computer* berkinerja tinggi, seperti NVIDIA Jetson Series, juga perlu dipertimbangkan untuk meningkatkan kecepatan deteksi (FPS) guna mencapai performa *tracking* objek secara *real-time* yang lebih optimal (Firdaus & Lelono, 2025).

Selain itu, pada proses *tracking* objek juga sering terjadi permasalahan seperti *occlusion* (oklusi) dan *false negative detection* yang merupakan masalah kedua bagi penelitian ini. Fenomena oklusi, di mana sebagian objek tertutup oleh penghalang tertentu, merupakan masalah fundamental dalam *tracking* objek. Oklusi merupakan penyebab utama yang meningkatkan *false-negative detection rate*, yang dapat menurunkan kinerja *tracking* secara keseluruhan. Objek yang teroklusi ini

dikategorikan sebagai *hard-positive examples* yang sulit dideteksi oleh model. Oleh karena itu, oklusi dan *false-negative* adalah hal yang harus diperhatikan untuk mencapai optimalitas dalam penggunaan *object detection* (Ryu & Chung, 2021).

Untuk menjawab tantangan tersebut, penelitian ini mengusulkan penggunaan *Kalman Filter* pada hasil deteksi bola berbasis YOLO. *Kalman Filter* merupakan salah satu algoritma estimasi yang banyak digunakan dalam sistem dinamis karena mampu memprediksi keadaan berikutnya dari suatu objek berdasarkan data pengamatan sebelumnya. Metode *Kalman Filter* menggunakan informasi dari objek yang terdeteksi di suatu *frame* dan status objek dari *frame* sebelumnya untuk mendapatkan status yang baru dari objek tersebut (C. Saputra, 2023).

Pada konteks robot sepak bola, *Kalman Filter* sangat relevan untuk digunakan karena bola sering mengalami perubahan posisi secara cepat, mendadak, dan terkadang tidak terduga. YOLO sebagai detektor objek hanya memberikan posisi bola pada setiap *frame*, tanpa mempertimbangkan dinamika gerakan dari bola tersebut. Akibatnya, jika bola bergerak terlalu cepat atau terjadi keterlambatan proses inferensi, robot bisa kehilangan akurasi dalam mengejar bola. Dengan *Kalman Filter*, sistem tidak hanya bergantung pada deteksi saat ini, tetapi juga dapat memprediksi posisi bola pada *frame* berikutnya, sehingga pergerakan robot menjadi lebih stabil, responsif, dan efisien.

Selain itu, *Kalman Filter* memiliki keunggulan dari sisi efisiensi komputasi. Dibandingkan dengan metode prediksi berbasis *deep learning* yang umumnya membutuhkan sumber daya tinggi, *Kalman Filter* relatif ringan dan dapat diimplementasikan pada perangkat dengan keterbatasan komputasi seperti robot KRSBI Beroda. Penelitian sebelumnya yang dilakukan oleh Yuztiawan & Utaminingrum (2017) juga memperkuat hal ini, di mana integrasi model YOLOv8N dengan *Kalman Filter* pada kondisi lingkungan dengan banyak objek mampu meningkatkan akurasi dan stabilitas deteksi serta pelacakan hingga 91,66%. Bahkan, penggunaan *Kalman Filter* terbukti meningkatkan kinerja pelacakan sebesar 25% dibandingkan hanya menggunakan YOLOv8N saja. Menariknya, penambahan algoritma tersebut hanya memberikan tambahan waktu komputasi rata-rata sekitar 0,0076 detik per *frame* (sekitar 7,92%), sehingga sistem tetap

mampu bekerja secara *real-time*. Dengan demikian, *Kalman Filter* tidak hanya meningkatkan akurasi pelacakan, tetapi juga tetap mempertahankan efisiensi komputasi yang sangat penting bagi robot kompetitif dengan keterbatasan perangkat keras.

Berdasarkan permasalahan yang telah diuraikan, peneliti tertarik untuk meneliti dengan judul "Optimasi Gerakan Robot Sepak Bola Menggunakan *Kalman Filter* Pada *Tracking* Objek Berbasis YOLO". Penelitian ini bertujuan untuk mengembangkan dan mengimplementasikan sistem deteksi serta pelacakan bola pada robot KRSBI Beroda dengan memanfaatkan kamera *omnidirectional* dan pendekatan *deep learning* berbasis CNN. Deteksi objek akan dilakukan menggunakan algoritma YOLO, sementara prediksi pergerakan bola diperkuat dengan penerapan *Kalman Filter* untuk mengatasi keterbatasan *tracking* berbasis *frame* tunggal. Dengan kombinasi ini, robot diharapkan tidak hanya mampu mengenali bola dan gawang secara akurat, tetapi juga dapat melakukan respons gerakan yang lebih mulus, stabil, dan efisien meskipun bola bergerak cepat atau kondisi lapangan penuh gangguan visual.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka permasalahan dalam penelitian ini dapat dirumuskan ke dalam beberapa pertanyaan sebagai berikut:

1. Bagaimana meningkatkan kestabilan *tracking* bola pada robot KRSBI Beroda ketika YOLO mengalami keterbatasan performa akibat rendahnya kapasitas perangkat keras (laptop ASUS K401U) sehingga menghasilkan *frame rate* yang tidak stabil?
2. Bagaimana mengurangi dampak *occlusion* dan *false-negative detection* yang sering terjadi pada proses *tracking* bola berbasis YOLO, sehingga robot tetap mampu mengetahui dan memprediksi posisi bola secara konsisten pada kondisi lapangan yang dinamis?

1.3. Tujuan Penelitian

Berdasarkan rumusan masalah yang telah dikemukakan, maka tujuan dari penelitian ini adalah sebagai berikut:

1. Mengembangkan sistem deteksi dan pelacakan bola yang lebih stabil dan responsif pada robot KRSBI Beroda dengan mengoptimalkan proses *tracking* objek menggunakan algoritma *Kalman Filter* untuk mengatasi keterbatasan perangkat keras.
2. Meningkatkan kontinuitas pelacakan bola dengan memanfaatkan kemampuan prediksi *Kalman Filter* untuk mengurangi pengaruh *occlusion* dan *false-negative detection*, sehingga robot dapat bergerak lebih efisien, tepat sasaran, dan adaptif terhadap dinamika permainan.

1.4. Batasan Masalah

Adapun batasan masalah dari penelitian ini adalah sebagai berikut:

1. Penelitian ini hanya membahas deteksi dan pelacakan bola menggunakan kamera *omnidirectional* pada robot KRSBI Beroda.
2. Algoritma deteksi objek yang digunakan adalah YOLO yang sudah ada, tanpa melakukan pengembangan arsitektur baru.
3. Prediksi posisi bola dilakukan dengan menggunakan *Kalman Filter*, tanpa membandingkannya dengan algoritma prediksi lain.
4. Fokus penelitian adalah pada optimasi gerakan robot terhadap bola, tidak mencakup aspek strategi tim, komunikasi antar robot, atau kontrol *hardware* secara mendetail.

1.5. Manfaat Penelitian

Berikut adalah manfaat dilakukannya penelitian ini bagi beberapa pihak:

1. Peneliti akan memperoleh pengalaman berharga dalam mengimplementasikan algoritma YOLO yang dipadukan dengan *Kalman Filter* pada sistem robotika, khususnya dalam konteks *tracking* objek bergerak.
2. Penelitian ini dapat membantu meningkatkan performa robot dalam pertandingan melalui sistem *tracking* bola yang lebih akurat dan gerakan robot yang lebih stabil serta efisien.

1.6. Sistematika Penulisan

Untuk mempermudah dalam memahami lebih jelas tentang penulisan penelitian ini, maka penelitian ini ditulis dalam beberapa bab yang masing-masing berkaitan satu sama lainnya, dengan sistematika penulisan sebagai berikut:

BAB I PENDAHULUAN

Bagian ini berisi tentang deskripsi umum dari penelitian yang akan dilakukan meliputi Latar Belakang, Perumusan Masalah, Tujuan Penelitian, Batasan Masalah, Manfaat Penelitian dan Sistematika Penulisan.

BAB II LANDASAN TEORI

Bagian ini membahas penelitian terdahulu, teori-teori dan pendapat para ahli yang berhubungan dengan penelitian yang dilakukan.

BAB III METODOLOGI PENELITIAN

Bagian ini berisi tentang alat dan bahan penelitian yang dilakukan, metode dan alur penelitian, metode pengembangan sistem cerdas, metode pengumpulan data, teknik mengolah data, dan teknik menguji hasil olahan data.

BAB IV HASIL DAN PEMBAHASAN

Bab ini menjelaskan tentang hasil perancangan dan analisa yang telah dilakukan sesuai dengan metodologi penelitian, sekaligus mengevaluasi hasil pengujian terhadap parameter-parameter uji yang telah ditetapkan.

BAB V KESIMPULAN DAN SARAN

Bab ini menjelaskan tentang simpulan hasil penelitian yang diperoleh sesuai dengan tujuan penelitian serta memuat saran mengenai masalah dan kemungkinan pemecahannya untuk penelitian selanjutnya.

DAFTAR PUSTAKA

LAMPIRAN

BAB II
LANDASAN TEORI

2.1. Penelitian Terdahulu

Penelitian mengenai sistem deteksi objek, khususnya dalam penggunaan model *Convolutional Neural Network* (CNN) seperti *You Only Look Once* (YOLO), telah banyak dilakukan sebelumnya. Penelitian-penelitian tersebut memberikan kontribusi penting dalam pengembangan metode deteksi objek maupun optimasi model menggunakan algoritma bantuan seperti estimator. Dengan meninjau penelitian terdahulu, penulis memperoleh wawasan terkait kelemahan metode sebelumnya, potensi pengembangan, serta peluang penerapan algoritma terbaru yang lebih efektif.

Pertama, penelitian yang berjudul "*CNN-Based Ball and Goal Detection for KRSBI Robot with Omnidirectional Camera*" dilakukan oleh Farhan & Candra (2025). Penelitian ini berfokus pada pengembangan sistem deteksi bola dan gawang berbasis *computer vision* pada robot yang mengikuti Kompetisi Robot Sepak Bola Indonesia (KRSBI) Beroda menggunakan kamera *omnidirectional*. Metode tradisional seperti HSV *color filtering* sebelumnya banyak digunakan karena implesmentasinya sederhana, namun performanya sangat bergantung pada intensitas cahaya sehingga tidak stabil dalam kondisi pencahayaan yang bervariasi. Untuk mengatasi kelemahan tersebut, penelitian ini mengusulkan penerapan algoritma YOLO yang berbasis CNN guna meningkatkan akurasi dan keandalan deteksi objek secara *real-time*. Dataset yang digunakan terdiri dari 1.125 citra dengan variasi pencahayaan dan posisi objek yang berbeda, kemudian dibagi menjadi 80% data pelatihan dan 20% data validasi. Model YOLOv8 dilatih menggunakan *Ultralytics* di platform Google Colab selama 100 *epoch*. Hasil pelatihan menunjukkan performa deteksi yang sangat tinggi, dengan tingkat akurasi sebesar 95,87%, *precision* mencapai 1.00 pada *confidence threshold* 0.921, *recall* sebesar 0.99, dan nilai *F1-Score* maksimum 0.97 pada *confidence* 0.149. Berdasarkan kurva *precision–confidence*, model dapat menghasilkan deteksi tanpa *false positive* pada ambang kepercayaan tinggi, sedangkan kurva *recall–confidence*

menunjukkan kemampuan model mendeteksi hampir semua objek meskipun pada nilai *confidence* yang rendah. Hasil penelitian ini membuktikan bahwa model YOLOv8 berbasis CNN mampu memberikan solusi yang tangguh dan efisien untuk sistem deteksi bola dan gawang secara *real-time* pada robot sepak bola beroda. Selain itu, performa model menunjukkan tingkat stabilitas yang baik terhadap perubahan pencahayaan dan posisi objek, sehingga metode ini dinilai efektif untuk diterapkan pada sistem persepsi visual robot KRSBI Beroda.

Kedua, penelitian yang berjudul "Implementasi *Object Detection* pada Robot Sepak Bola Beroda Berbasis Kamera *Omnidirectional* Menggunakan Opencv" dilakukan oleh Nanda et al. (2023). Penelitian ini mengangkat permasalahan terkait sistem deteksi pada robot KRSBI yang masih sangat dipengaruhi oleh kondisi pencahayaan. Perubahan intensitas cahaya di lapangan membuat sistem kesulitan mempertahankan akurasi deteksi, sehingga kinerja robot menjadi tidak stabil ketika kondisi cahaya berubah. Metode yang digunakan dalam penelitian ini adalah *object detection* berbasis kamera *omnidirectional* dengan bantuan pustaka OpenCV. Proses pengujian dilakukan dengan cara mengukur variasi intensitas cahaya di lapangan untuk melihat pengaruhnya terhadap kemampuan deteksi objek. Hasil yang diperoleh menunjukkan bahwa robot KRSBI mampu mengenali beberapa objek penting, seperti bola, gawang, serta robot lawan (*cyan* dan *magenta*), dengan tingkat akurasi sekitar 70%. Akan tetapi, nilai akurasi ini masih cukup dipengaruhi oleh perbedaan intensitas cahaya dari masing-masing objek. Dengan demikian, penelitian ini membuktikan bahwa penerapan *object detection* menggunakan kamera *omnidirectional* dan OpenCV dapat berjalan pada robot KRSBI, namun tetap memiliki keterbatasan signifikan terutama ketika menghadapi kondisi pencahayaan yang tidak stabil.

Ketiga, penelitian yang berjudul "Perancangan Sistem Pendeteksian Obyek Bola dengan Metode *Framework* YOLO V4" dilakukan oleh Nuralim et al. (2022). Latar belakang penelitian ini berangkat dari kebutuhan dalam KRSBI Beroda tahun 2018, di mana robot dituntut mampu melakukan navigasi serta menjalankan tugas utama, yaitu menemukan bola, menggiringnya, dan menendangnya ke arah gawang lawan. Untuk dapat melaksanakan fungsi tersebut, robot membutuhkan sistem

pendeteksian bola yang cepat, akurat, dan responsif. Metode yang digunakan dalam penelitian ini adalah *object detection* berbasis *framework* YOLOv4. Pengujian sistem dilakukan menggunakan *confusion matrix* untuk mengukur performa deteksi, termasuk dalam kondisi ketika bola sebagian terhalang oleh objek lain. Hasil pengujian menunjukkan bahwa sistem mampu mendeteksi bola meskipun terdapat penghalang dengan tingkat persentase 50%, 60%, hingga 70%. Namun, kemampuan deteksi menurun drastis ketika tingkat penghalang mencapai 80% ke atas. Pada kondisi tersebut (80%, 90%, hingga 100% penghalang), robot tidak lagi dapat mengenali keberadaan bola. Penelitian ini membuktikan bahwa YOLOv4 cukup efektif dalam mendeteksi bola pada kondisi yang dinamis dan kompleks, termasuk ketika sebagian objek tertutup. Akan tetapi, keterbatasan tetap muncul pada tingkat *occlusion* yang tinggi, yang menunjukkan perlunya optimasi atau integrasi metode tambahan untuk meningkatkan keandalan deteksi dalam skenario pertandingan nyata.

Penelitian berikutnya dilakukan oleh Sholehurrohman et al. (2023) dengan judul "Analisis Metode *Kalman Filter*, *Particle Filter* dan *Correlation Filter* Untuk Pelacakan Objek" membahas mengenai tantangan dalam pelacakan objek (*object tracking*) pada bidang *computer vision*. Penelitian ini mengimplementasikan tiga metode, yaitu *Kalman Filter*, *Particle Filter*, dan *Correlation Filter* untuk pelacakan objek pada data video lalu lintas dan video sirkuit Naskar. Hasil penelitian menunjukkan bahwa metode *Kalman Filter* memiliki akurasi tertinggi mencapai 96,89%, sedangkan metode *Correlation Filter* lebih unggul dalam aspek performa komputasi dengan rata-rata 26,69 FPS, sementara *Particle Filter* berada di bawah *Kalman Filter* dalam hal akurasi. Kesimpulan dari penelitian ini menegaskan bahwa *Kalman Filter* sangat potensial digunakan dalam pelacakan objek yang membutuhkan akurasi tinggi, sementara *Correlation Filter* lebih sesuai untuk kebutuhan aplikasi *real-time* karena efisiensi komputasinya. Dengan demikian, penelitian ini dapat dijadikan acuan penting dalam mendukung pemanfaatan *Kalman Filter* pada penelitian terkait optimasi pergerakan robot sepak bola, khususnya dalam memprediksi pergerakan bola secara akurat.

Penelitian kelima berjudul "Implementasi Algoritma SIFT (*Scale-Invariant Feature Transform*) dan Algoritma *Kalman Filter* dalam Mendeteksi Objek Bola" yang dilakukan oleh C. Saputra (2023) berfokus pada pengembangan sistem pendeteksian bola pada robot KRSBI Beroda. Latar belakang penelitian ini adalah tuntutan agar robot mampu mendeteksi, melacak, serta menggiring bola menuju gawang lawan secara efektif dalam KRSBI Beroda. Metode pendeteksian berbasis *color filtering* sebelumnya memang dinilai cukup baik dalam mengidentifikasi objek, namun masih memiliki kelemahan dalam aspek pelacakan (*tracking*). Untuk mengatasi hal tersebut, penelitian ini menggabungkan algoritma SIFT yang berfungsi membandingkan fitur citra guna memastikan objek yang terdeteksi benar-benar bola, serta algoritma *Kalman Filter* yang berperan sebagai estimator dalam memprediksi arah pergerakan bola berdasarkan status objek pada *frame* sebelumnya. Hasil dari penelitian ini menunjukkan bahwa kombinasi algoritma SIFT dan *Kalman Filter* dapat meningkatkan akurasi serta kecepatan pendeteksian bola. *Tracking* yang dilakukan dengan *Kalman Filter* mampu memprediksi pergerakan objek dengan baik, di mana koordinat y akan semakin kecil jika bola bergerak ke atas *frame* dan semakin besar jika bergerak ke bawah, sedangkan koordinat x akan semakin kecil ketika bola bergerak ke kiri dan semakin besar saat bergerak ke kanan. Dari hasil pengujian, sistem berhasil mendeteksi objek bola dengan sempurna pada jarak tertentu, dengan rata-rata *error* pengukuran *Kalman Filter* sebesar 1,06 untuk koordinat x dan 7,34 untuk koordinat y. Dengan demikian, penelitian ini menegaskan potensi integrasi SIFT dan *Kalman Filter* untuk menghasilkan sistem deteksi dan pelacakan bola yang lebih andal dalam mendukung performa robot KRSBI.

Penelitian selanjutnya yang relevan berjudul "*Detection and Recognition of Moving Video Objects: Kalman Filtering with Deep Learning*" yang dilakukan oleh Mohammed & Hussain (2021). Penelitian ini bertujuan untuk meningkatkan akurasi dalam proses deteksi dan pengenalan objek bergerak dalam urutan video. Tujuan ini diangkat karena adanya faktor penghalang seperti jarak deteksi kamera atau kekaburan (*blurring*) gambar yang dapat mengurangi akurasi teknik yang ada. Untuk mencapai akurasi yang lebih tinggi, metode yang diusulkan adalah sistem

hibrida yang menggabungkan *Kalman Filter* dengan CNN. *Kalman Filter* diterapkan pada tahap awal deteksi untuk menghilangkan latar belakang dan memotong objek, serta berfungsi sebagai estimator rekursif yang mampu memprediksi lokasi objek di masa depan, mengurangi *noise* dari deteksi yang salah, dan mengasosiasikan multi-objek ke treknya. Setelah itu, model CNN akan memprediksi kategori objek yang sudah dideteksi dan dipotong. Hasil eksperimen menunjukkan bahwa pendekatan hibrida ini berhasil mencapai akurasi pengenalan hingga 100% pada 8 video berbeda. Hasil ini menunjukkan superioritas sistem yang diusulkan dibandingkan dengan enam algoritma lain yang ada.

Penelitian selanjutnya adalah studi berjudul "*Optimized Object Tracking Technique Using Kalman Filter*" yang dilakukan oleh Taylor et al. (2016). Penelitian ini berfokus pada pengembangan teknik pelacakan objek yang efisien untuk mengatasi masalah waktu pemrosesan yang tinggi dalam pendeteksian objek di tengah adegan yang berantakan (*cluttered scene*). Metode konvensional pelacakan berbasis fitur seperti SIFT atau SURF (*Speeded Up Robust Feature*) dinilai akurat, tetapi membutuhkan waktu pemrosesan yang lebih tinggi. Sebaliknya, metode berbasis warna memiliki waktu pemrosesan yang lebih cepat, namun akurasi yang terbatas. *Kalman Filter* digunakan dalam penelitian ini adalah untuk mengatasi kompromi tersebut dan meningkatkan efisiensi komputasi, khususnya untuk aplikasi *real-time* seperti pada sistem robotik. *Kalman Filter* berperan sebagai estimator rekursif yang efisien untuk memprediksi lokasi objek di *frame* berikutnya berdasarkan status saat ini dan model gerakan objek (diasumsikan kecepatan konstan). Prediksi lokasi ini memungkinkan sistem hanya mencari objek dalam jendela gambar yang dipotong (*cropped image*) yang ukurannya jauh lebih kecil daripada keseluruhan *frame* video. Dengan demikian, waktu pemrosesan pendeteksian objek dapat diminimalkan secara signifikan. Hasil dari penelitian ini menunjukkan bahwa integrasi *Kalman Filter* dengan teknik *cropping* secara signifikan mempercepat waktu pemrosesan sambil mempertahankan tingkat akurasi yang tinggi. Waktu pemrosesan menjadi lebih cepat ketika ukuran jendela pencarian (*search window*) diperkecil. Untuk menyeimbangkan antara waktu pemrosesan yang minimal dan kesalahan jarak (*distance error*) yang minimal,

penelitian ini menyimpulkan bahwa ukuran jendela pencarian yang optimal adalah 2.16 kali dimensi terbesar objek. Pada ukuran ini, terdapat penurunan signifikan dalam waktu pemrosesan sekaligus memastikan objek terdeteksi dengan tingkat keberhasilan yang tinggi, serta pusat objek yang terdeteksi cukup dekat dengan pusat sebenarnya.

Penelitian selanjutnya oleh Egi (2022), yang berjudul "*Basketball self training shooting posture recognition and trajectory estimation using computer vision and Kalman Filter*", secara khusus berfokus pada peningkatan performa *self-shooting* bola basket, terutama untuk pemain muda, yang sering mengalami kesulitan dan keengganan akibat postur yang salah dan tembakan yang meleset tanpa bimbingan pelatih. Penelitian ini dibuat sebagai upaya untuk menyediakan sistem umpan balik otomatis yang dapat melacak gerakan pemain, mengenali postur, dan mengestimasi lintasan proyektil secara *real-time*. Penelitian ini menggunakan teknik *computer vision* seperti pemisahan saluran warna RGB, *Median Filter*, binarisasi, dan *Area Opening* untuk mendeteksi serta melabeli objek penting, yaitu bola basket dan *T-shirt* pemain. Dengan mengestimasi lintasan proyektil, penelitian ini menunjukkan bahwa lintasan tersebut dipengaruhi secara signifikan oleh ketidakpastian lingkungan, khususnya gaya hambat udara (*air drag force*). Untuk mengatasi *noisy medium* (medium bising) ini dan mengoptimalkan lintasan yang sebenarnya, algoritma *Kalman Filter* digunakan sebagai filter rekursif canggih untuk memprediksi posisi, kecepatan, dan percepatan objek bergerak serta mengurangi kesalahan prediksi secara berulang. Hasilnya, penelitian ini berhasil menentukan bahwa sudut tembakan optimal (*best shooting angle*) yang menghasilkan skor sukses paling tinggi bagi pemain dengan tinggi $1.73\ m$ dan kecepatan awal $9.5\ m/s$ adalah $50°$ ketika gaya hambat udara diperhitungkan. Perhitungan ini menunjukkan deviasi yang signifikan; tanpa hambatan udara, jangkauan maksimum horizontal (x_{max}) adalah $10.76\ m$, namun dengan hambatan udara, x_{max} turun menjadi $5.47\ m$, menegaskan peran krusial *Kalman Filter* dalam memprediksi lintasan secara akurat dalam kondisi dunia nyata.

Penelitian selanjutnya adalah "*ConfTrack: Kalman Filter-based Multi-Person Tracking by Utilizing Confidence Score of Detection Box*" yang diteliti oleh

Jung et al. (2024). Urgensi penelitian ini muncul dari kelemahan mendasar *tracker* berbasis *Kalman Filter-based Tracking-by-Detection* (KFTBD) konvensional ketika dihadapkan pada hasil deteksi yang *noisy* (kotak dengan *confidence* rendah) dalam situasi ramai. Deteksi berkepercayaan rendah ini terbukti berhubungan dengan nilai *Intersection over Union* (IoU) yang lebih rendah, yang secara langsung mengganggu kinerja pelacakan dan berpotensi menyebabkan *ID switch* atau kegagalan *track*. Untuk mengatasi masalah ini, *ConfTrack* menyarankan peningkatan fungsi *Kalman Filter* yang adaptif terhadap skor kepercayaan. Fungsi utama *Kalman Filter* dalam penelitian ini tetap sebagai estimator yang memprediksi dan memperbarui status *track*, namun ditambahkan tiga metode utama untuk menanggulangi *noise*, yaitu *Confidence Weighted Kalman-Update* (CW) untuk melakukan modifikasi pada pengukuran target *Kalman Filter*. Jika skor kepercayaan deteksi rendah, pengukuran yang digunakan akan lebih condong ke prediksi *Kalman Filter* daripada ke deteksi mentah (*noisy*). Kemudian menggunakan *Noise Scale Adaptive Kalman Filter* (NK) untuk memperkuat kovariansi *noise* ruang pengukuran yang digunakan untuk menghitung *Kalman gain* ketika skor kepercayaan rendah. Hal ini secara efektif memberikan penalti pada deteksi berkepercayaan rendah di tahap *matching*. Terakhir, menggunakan *Constant Box Prediction* (CP) untuk menstabilkan prediksi *Kalman Filter* untuk *lost track* dengan mengatur variasi ukuran kotak menjadi nol, meminimalkan dampak *noise* pada prediksi dimensi kotak. Hasil penelitian menunjukkan bahwa kombinasi metode berbasis *Kalman Filter* ini, khususnya NK, memberikan kontribusi paling konsisten dalam menaikkan metrik pelacakan. *ConfTrack* terbukti paling kuat di lingkungan ramai dan berkinerja baik dalam berbagai situasi, mencapai skor *Higher Order Tracking Accuracy* (HOTA) dan *Identification F1-Score* (IDF1) tertinggi pada *dataset Multiple Object Tracking Benchmark* 2020 (MOT20). Pencapaian IDF1 tertinggi menunjukkan bahwa *ConfTrack* sangat kuat terhadap *noise* deteksi di lingkungan ramai, memungkinkannya melacak secara stabil dengan tingkat *missing track* yang rendah dan menghasilkan *trajectory* yang stabil saat terjadi oklusi. Dengan demikian, penelitian ini menegaskan kembali

peran krusial *Kalman Filter* dalam *multi-person tracking*, khususnya melalui penyesuaian adaptif terhadap kualitas data input deteksi.

Penelitian selanjutnya dilakukan oleh Lee (2024) dalam artikel berjudul "*Confidence-Guided Frame Skipping to Enhance Object Tracking Speed*". Urgensi penelitian ini didasari oleh dilema antara kecepatan pemrosesan dan akurasi pelacakan pada perangkat dengan sumber daya komputasi terbatas, di mana algoritma pelacakan yang kuat cenderung memonopoli sumber daya. Untuk mengatasi hal tersebut, Lee mengusulkan pendekatan dua tingkat (*two-tiered approach*) yang mengintegrasikan metode pelacakan ringan (*lightweight*) berbasis *block-matching* dengan algoritma pelacakan yang kompleks dan kuat (*robust*). Inti dari metode ini adalah penggunaan skor kepercayaan (*confidence level*, C_L) yang dihitung berdasarkan nilai *Sum of Absolute Differences* (SAD) dan gradien tekstur objek untuk menentukan kebutuhan intervensi algoritma berat secara adaptif. Jika C_L berada di atas ambang batas, sistem hanya menjalankan pelacakan ringan, namun jika C_L rendah atau telah mencapai batas maksimal frame yang dilewati (S_N), algoritma berat akan dipanggil untuk memastikan reliabilitas pelacakan. Hasil eksperimen menunjukkan bahwa metode ini mampu meningkatkan kecepatan pemrosesan hingga 1,5 kali lipat dengan degradasi akurasi yang sangat minimal dibandingkan menjalankan algoritma berat di setiap *frame*. Penelitian ini menegaskan bahwa strategi *frame skipping* yang dipandu oleh evaluasi kepercayaan sangat efektif untuk mengoptimalkan penggunaan sumber daya komputasi tanpa mengorbankan kualitas pelacakan secara signifikan.

Penelitian yang terakhir adalah "*Automatic pear and apple detection by videos using deep learning and a Kalman Filter*" oleh Takura et al. (2021). Penelitian ini mengatasi tantangan krusial dalam otomatisasi pertanian, yaitu estimasi jumlah buah di kebun secara akurat. Proses penghitungan buah dibagi menjadi dua langkah penting: deteksi buah dan pelacakan buah (*tracking*) melalui urutan *frame* gambar. Urgensi pelacakan muncul karena kondisi lapangan yang tidak stabil, seperti perubahan pencahayaan di bawah kanopi pohon, serta kemungkinan buah tertutup oleh daun atau ranting (*occlusion*), yang dapat menyebabkan kegagalan deteksi di beberapa *frame*. Tanpa pelacakan yang andal,

buah yang muncul kembali setelah tersembunyi dapat dihitung ganda (*double-counted*), sehingga hasil penghitungan menjadi tidak akurat. Untuk mengatasi masalah ini, metode deteksi objek berbasis *deep learning* YOLO v2 digunakan untuk mengidentifikasi buah di setiap *frame*. Kemudian, *Kalman Filter* diterapkan untuk pelacakan objek. Fungsi utama *Kalman Filter* adalah sebagai estimator yang memprediksi dan mengoreksi status (posisi dan kecepatan) buah yang sedang dilacak, bahkan ketika deteksi buah gagal di *frame* tersebut. Ini memastikan bahwa buah yang sama dapat terus diidentifikasi di *frame* yang berurutan, meskipun sempat tidak terdeteksi. Hasilnya menegaskan efektivitas integrasi ini: sistem berhasil menghitung buah pir dan apel secara otomatis dengan kesalahan absolut kurang dari 10%. Secara spesifik, dari total 234 buah pir, 226 buah berhasil dihitung dengan benar, menghasilkan nilai F1 sebesar 0.972. Performa *Kalman Filter* juga terbukti jauh lebih unggul dibanding algoritma pelacakan fitur Kanade-Lucas-Tomasi (KLT), yang menyebabkan penghitungan berlebih (*over-counted*). Dengan demikian, penelitian ini menunjukkan bahwa *Kalman Filter* sangat efektif dan esensial dalam menstabilkan proses penghitungan objek bergerak pada video di lingkungan yang tidak terkontrol.

Tabel 2.1 Penelitian Terdahulu

No	Peneliti	Tahun	Judul	Masalah	Metode	Hasil	Perbedaan
1	T. Mohd. Farhan, Feri Chandra	2024	*CNN-Based Ball and Goal Detection for KRSBI Robot with Omnidirectional Camera*	Metode deteksi objek penyaringan warna HSV memiliki kinerja yang buruk di bawah variasi kondisi pencahayaan pada robot KRSBI-Beroda	Menggunakan CNN berbasis algoritma YOLOv8 dengan kamera *omnidirectional*. Model dilatih menggunakan 1.125 gambar selama 100 epochs.	Model mencapai Akurasi 95,87%, Presisi 1.00 (pada confidence 0.921) 8, Recall 0.99, dan F1-Score 0.97 (pada confidence 0.149).	Penelitian ini fokus meningkatkan kinerja robot dari menggunakan HSV ke YOLO, sedangkan penelitian sekarang menambahkan optimasi gerakan dengan *Kalman Filter*.
2	Bagas Musamma Nanda, Simon Siregar, dan Muhammad Ikhsan Sani	2023	Implementasi *Object Detection* pada Robot Sepak Bola Beroda Berbasis Kamera *Omnidirectional* Menggunakan Opencv	Deteksi objek yang bergantung pada intensitas cahaya, sehingga membuat hasil deteksi menjadi kurang efektif	OpenCV *Object Detection*	Robot bisa deteksi bola, gawang, robot *cyan* & magenta dengan akurasi ±70%, tapi hasil dipengaruhi intensitas cahaya.	Penelitian ini hanya menggunakan OpenCV sederhana, sedangkan penelitian sekarang memakai YOLO + *Kalman Filter*.

No	Peneliti	Tahun	Judul	Masalah	Metode	Hasil	Perbedaan
3	Jalu Nuralim, Nifty Fath, Akhmad Musafa, Sujono, Drs. Suwasti Broto	2022	Perancangan Sistem Pendeteksian Obyek Bola dengan Metode *Framework* YOLO V4	Robot butuh deteksi cepat untuk navigasi & menendang bola, namun terhambat jika bola terhalang sebagian.	YOLOv4 + *Confusion Matrix*	Robot masih bisa deteksi bola hingga 70% tertutup, gagal pada >80%.	Fokus pada akurasi YOLOv4 dengan halangan, sedangkan penelitian sekarang menambahkan prediksi pergerakan menggunakan *Kalman Filter*.
4	Ridho Sholehurrohman, Mochammad Reza Habibi, Igit Sabda Ilman, Rahman Taufiq, Muhaqiqin	2023	Analisis Metode *Kalman Filter*, *Particle Filter* dan *Correlation Filter* Untuk Pelacakan Objek	*Tracking* objek sering gagal karena oklusi dan deformasi target.	*Kalman Filter*, *Particle Filter*, *Correlation Filter*	*Kalman Filter* akurasi 96,89%, *Correlation Filter* paling cepat 26,69 FPS.	Penelitian ini membandingkan metode tracking umum, penelitian sekarang fokus *Kalman Filter* untuk robot sepak bola berbasis YOLO.
5	M. Irwan Bustami, Chindra Saputra, Desi Kisbianty, Arjuna Panji Prakarsa	2023	Implementasi Algoritma SIFT (*Scale-Invariant Feature Transform*) dan *Kalman Filter* dalam	Metode *color filtering* saja kurang handal untuk *tracking* bola.	SIFT + *Kalman Filter*	SIFT mengenali objek, *Kalman Filter* memprediksi arah bola. *Error* rata-rata 1.06 (x) dan 7.34 (y).	Penelitian ini gabungkan SIFT + *Kalman Filter*, sedangkan penelitian sekarang pakai YOLO untuk deteksi dan *Kalman Filter* untuk prediksi.

No	Peneliti	Tahun	Judul	Masalah	Metode	Hasil	Perbedaan
			Mendeteksi Objek Bola				
6	Hind Rustum Mohammed & Zahir M. Hussain	2021	*Detection and recognition of moving video objects: Kalman Filtering with deep learning*	Kebutuhan akan akurasi yang lebih tinggi dalam deteksi dan pengenalan objek bergerak, terutama menghadapi kendala seperti jarak kamera atau *blurring*.	*Kalman Filter* + CNN	Akurasi pengenalan mencapai 100% pada 8 video berbeda. Sistem menunjukkan superioritas dibandingkan enam algoritma yang ada.	*Kalman Filter* digunakan sebagai tahap awal untuk deteksi dan *tracking*. Metode deteksi awal tidak menggunakan YOLO, melainkan *Temporal Median* dan *thresholding* untuk menghilangkan *background*.
7	Liana Ellen Taylor, Midriem Mirdanies, Roni Permana Saputra	2016	*Optimized Object Tracking Technique Using Kalman Filter*	Waktu pemrosesan yang tinggi pada deteksi objek, terutama untuk aplikasi *real-time*, sambil mempertahankan akurasi dalam adegan yang	*Kalman Filter* + *Color Segmentation* + *Cropped Image*	Proses pelacakan menjadi lebih cepat secara signifikan (waktu pemrosesan berkurang) dengan mempertahankan akurasi tinggi. Ukuran jendela pencarian optimal ditemukan pada	Penelitian ini menggunakan Segmentasi Warna dan *Kalman Filter* untuk mengoptimalkan waktu pemrosesan melalui teknik *Cropped Image*. Sedangkan penelitian sekarang menggunakan YOLO (model *deep*

No	Peneliti	Tahun	Judul	Masalah	Metode	Hasil	Perbedaan
				berantakan (*cluttered scene*).		2.16 kali dimensi terbesar objek, menyeimbangkan waktu pemrosesan dan kesalahan jarak.	*learning* yang kompleks) sebagai detektor, dan *Kalman Filter* digunakan untuk stabilisasi *tracking* dan mengatasi oklusi/*false negative* akibat beban komputasi YOLO yang tinggi.
8	Yunus Egi	2022	*Basketball self training shooting posture recognition and trajectory estimation using computer vision and Kalman Filter*	Kurangnya umpan balik dari pelatih membuat *self-shooting* menyulitkan dan memakan waktu, di mana postur salah menyebabkan tembakan meleset dan menimbulkan keengganan. Diperlukan algoritma untuk	RGB *Channelization, Median Filter*, Binarisasi, *Area Opening* + *Kalman Filter*	Sistem berhasil mengklasifikasikan objek dan menentukan sudut postur terbaik. *Kalman Filter* terbukti penting karena tanpa hambatan udara (x_{max} 10.76 m), hasil berbeda signifikan dari kondisi nyata dengan hambatan udara (x_{max} 5.47	Penelitian ini berfokus pada analisis postur manusia dan estimasi lintasan proyektil fisik (bola basket) dalam domain olahraga, menggunakan CV tradisional. *Kalman Filter* berfungsi untuk mengoreksi *noise* fisik (gaya hambat udara) pada lintasan. Penelitian sekarang berfokus pada optimasi gerakan robot dalam domain Robot

No	Peneliti	Tahun	Judul	Masalah	Metode	Hasil	Perbedaan
				memberikan *feedback* postur dan lintasan *real-time*.		m), sehingga filter berhasil mengoreksi prediksi lintasan fisik. Sudut terbaik yang terdeteksi 51°.	Sepak Bola, menggunakan CV berbasis YOLO. *Kalman Filter* berfungsi untuk menstabilkan *noise* deteksi objek (oklusi/ *false negative*) dan mereduksi beban komputasi.
9	Hyeonchul Jung, Seokjun Kang, Takgen Kim, HyeongKi Kim	2024	*ConfTrack*: *Kalman Filter-based Multi-Person Tracking by Utilizing Confidence Score of Detection Box*	Kegagalan *tracking* dan *ID switch* pada *tracker Kalman Filter* konvensional yang disebabkan oleh hasil deteksi berkepercayaan rendah di lingkungan yang ramai	*Kalman Filter*, *Confidence Weighted Kalman-Update* (CW), *Noise Scale Adaptive Kalman Filter* (NK)	*ConfTrack* mencapai skor HOTA dan IDF1 tertinggi pada *dataset* MOT20, membuktikan *robustness* yang tinggi terhadap *noise* deteksi di lingkungan yang padat.	*ConfTrack* memodifikasi *Kalman Filter* untuk mempenalti/menghukum deteksi berkepercayaan rendah guna stabilitas ID. Penelitian sekarang menggunakan *Kalman Filter* untuk memprediksi pergerakan bola secara stabil, mengoptimasi gerakan robot, dan mengurangi beban komputasi pada YOLO.

No	Peneliti	Tahun	Judul	Masalah	Metode	Hasil	Perbedaan
10	Yun Gu Lee	2024	*Confidence-Guided Frame Skipping to Enhance Object Tracking Speed*	Konsumsi daya komputasi yang tinggi dari algoritma pelacakan berat pada perangkat terbatas.	Pendekatan dua tingkat menggunakan *Block-Matching* (ringan) dan pelacak *Robust* yang dipandu skor kepercayaan (C_L) .	Kecepatan meningkat hingga 1.5x dengan akurasi yang hampir identik dengan metode dasar.	Menggunakan *block-matching* untuk *frame skipping*, sementara penelitian ini menggunakan YOLOv8 dan *Kalman Filter* dengan bantuan *frame skipping* untuk robot KRSBI.
11	Kenta Itakura, Yuma Narita, Shuhei Noaki, & Fumiki Hosoi	2021	*Automatic pear and apple detection by videos using deep learning and a Kalman Filter*	Penghitungan buah di kebun dari video sulit karena perubahan kondisi cahaya yang cepat dan tidak stabil, buah dapat tertutup daun/ranting (*occlusion*).	YOLOv2 + *Kalman Filter*.	Rata-rata presisi (*Average Precision*) deteksi pir dan apel tinggi (0.97). Penghitungan pir benar: 226 dari 234 (F1 = 0.972). *Kalman Filter* mampu melacak buah yang sempat tidak terdeteksi (*occluded*).	Penelitian ini fokus pada objek statis (buah) di lingkungan pertanian yang tidak terkontrol (pencahayaan, oklusi, pergerakan kamera *handheld*), sedangkan penelitian sekarang fokus pada objek dinamis (bola) yang bergerak cepat oleh robot di lingkungan yang lebih terkontrol.

Secara keseluruhan, tinjauan penelitian terdahulu menunjukkan adanya konsensus bahwa algoritma *deep learning* seperti YOLO unggul dalam akurasi deteksi objek, tetapi memiliki kerentanan terhadap isu *real-time* dan penggunaan sumber daya yang intensif, yang sangat krusial dalam sistem robotika dinamis. Penelitian sebelumnya telah membuktikan bahwa integrasi antara *tracker* berbasis filter dengan detektor berbasis *vision* mampu meningkatkan performa pelacakan secara signifikan. Secara spesifik, *Kalman Filter* (KF) menjadi pilihan optimal karena sifatnya yang ringan secara komputasi dan efektif dalam mereduksi *noise* serta memprediksi posisi objek saat terjadi oklusi.

Namun, menjalankan algoritma deteksi yang kompleks di setiap *frame* dianggap tidak efisien karena tidak semua *frame* memerlukan kalkulasi ulang yang berat. Mengadopsi strategi *frame skipping* yang diusulkan oleh Lee (2024), penelitian ini mengusulkan optimasi melalui pembagian beban kerja komputasi. Implementasi ini dilakukan dengan menjalankan YOLOv8 sebagai detektor pada *frame* awal, kemudian memanfaatkan *Kalman Filter* untuk memprediksi posisi objek secara mandiri pada tiga *frame* berikutnya ($S_L = 3$) sebelum memanggil kembali detektor untuk sinkronisasi ulang.

Perbedaan mendasar yang menjadi celah penelitian ini terletak pada implementasi yang mengintegrasikan secara spesifik YOLOv8, *Kalman Filter*, dan skema *frame skipping* adaptif untuk optimasi gerakan robot pada KRSBI Beroda UNRI. Dengan demikian, penelitian ini bertujuan untuk menjembatani celah tersebut dengan menguji secara kuantitatif efektivitas integrasi metode tersebut guna menghasilkan sistem kontrol gerak robot yang lebih stabil, efisien secara daya komputasi, dan kompetitif dalam situasi pertandingan.

2.2. Kontes Robot Sepak Bola Beroda Indonesia

Salah satu ajang kompetisi robotik di Indonesia adalah Kontes Robot Indonesia atau lebih sering disebut dengan KRI. KRI diselenggarakan oleh Pusat Prestasi Nasional (PUSPRESNAS) dan Kementrian Pendidikan dan Kebudayaan Republik Indonesia. KRI merupakan acara yang diadakan setiap tahun dan diikuti oleh mahasiswa dari berbagai wilayah di Indonesia mulai dari Timur, Tengah dan

Barat. KRI terbagi menjadi 6 kategori, yang salah satunya adalah Kontes Robot Sepak Bola Indonesia (KRSBI) Beroda (Nanda et al., 2023).

Kontes Robot Sepakbola Beroda Indonesia diadakan untuk meningkatkan keilmuan dan kreatifitas mahasiswa di bidang robotika. Di dalam kontes ini, mahasiswa dituntut untuk bisa mengembangkan kemampuan dalam mekanika, manufaktur, elektronika, pemrograman, *articial intelligent*, *image processing*, komunikasi digital, dan strategi, sekaligus diperlukan pengembangan ke arah disiplin, toleransi, sportifitas, kerjasama, saling menghargai, kontrol emosi dan kemampuan *softskill* lainnya (Kusumoputro et al., 2024).

Gambar 2.1 Ilustrasi Pertandingan KRSBI Beroda

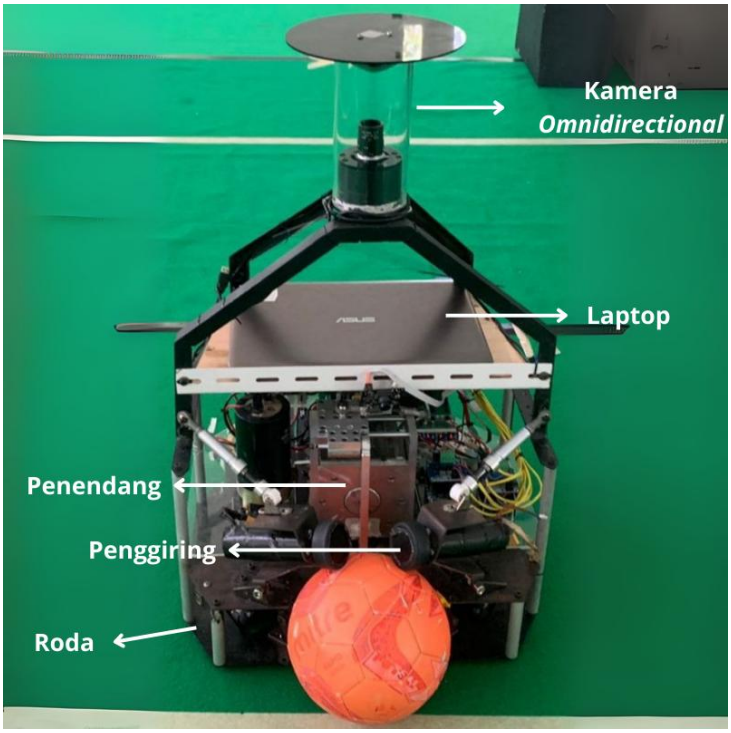
Sumber : Rochardjo, 2024

2.3. Robot Sepak Bola Beroda

Dalam KRSBI Beroda, robot yang digunakan pada tahap ini merupakan robot yang sama seperti yang dipakai pada pertandingan tingkat nasional, dengan beberapa ketentuan khusus. Jumlah robot yang diizinkan adalah dua unit, yaitu Robot 1 (R1) dan Robot 2 (R2), keduanya bertipe robot penyerang. Adapun spesifikasi fisik robot diatur dengan ukuran proyeksi ke lantai minimal 30 cm × 30 cm dan maksimal 52 cm × 52 cm, dengan tinggi robot antara 40 cm hingga 80 cm. Jika tinggi robot melebihi 60 cm, maka bagian tubuh robot di atas ketinggian tersebut harus berada dalam silinder berdiameter 25 cm. Berat maksimum robot

ditetapkan 40 kg, sedangkan bentuk robot dibuat bebas selama sesuai regulasi, dan warna yang digunakan adalah hitam (Kusumoputro et al., 2024).

Robot yang digunakan dalam Kontes Robot Sepak Bola Indonesia (KRSBI) Beroda memiliki bentuk khas menyerupai kubus dengan berbagai komponen penting yang berfungsi mendukung pergerakan serta kemampuan bermain sepak bola. Gambar 2.1 memperlihatkan salah satu contoh robot KRSBI Beroda yang dilengkapi dengan berbagai sistem pendukung.



Gambar 2.2 Robot Sepak Bola Beroda

Kamera *omnidirectional* ditempatkan di bagian atas robot dan berfungsi untuk mendeteksi objek seperti bola, gawang, maupun robot lawan. Konsep kamera *omnidirectional* adalah menangkap citra dari segala arah (depan, belakang, kiri, dan kanan) menggunakan teknik pantulan cermin cembung yang diarahkan ke bawah (Surya et al., 2025). Dengan cara ini, kamera mampu memperoleh citra lapangan secara menyeluruh hanya dari satu titik pengamatan. Robot dilengkapi dengan penendang berbasis solenoid. Solenoid adalah sebuah komponen elektromagnetik yang berfungsi mengubah energi listrik menjadi energi mekanis. Energi mekanis

yang dihasilkan dapat berupa gerakan mendorong (*push*) maupun menarik (*pull*) (Shofi et al., 2023).

Pada bagian depan robot terdapat roda kecil yang dipasangi motor penggerak dan berputar ke arah dalam tubuh robot. Roda ini berfungsi untuk menangkap dan menahan bola ketika robot bergerak, sehingga bola tetap berada di depan robot dan tidak mudah terlepas. Bagian terakhir adalah roda yang berfungsi sebagai sistem penggerak robot. Pada robot KRSBI Beroda digunakan tiga roda utama agar pergerakan lebih fleksibel. Setiap roda terdiri dari dua komponen, yaitu motor dengan kecepatan putar sekitar 500 rpm yang menjadi sumber tenaga penggerak, serta *omni-wheel* yang memungkinkan robot dapat bergerak bebas ke berbagai arah.

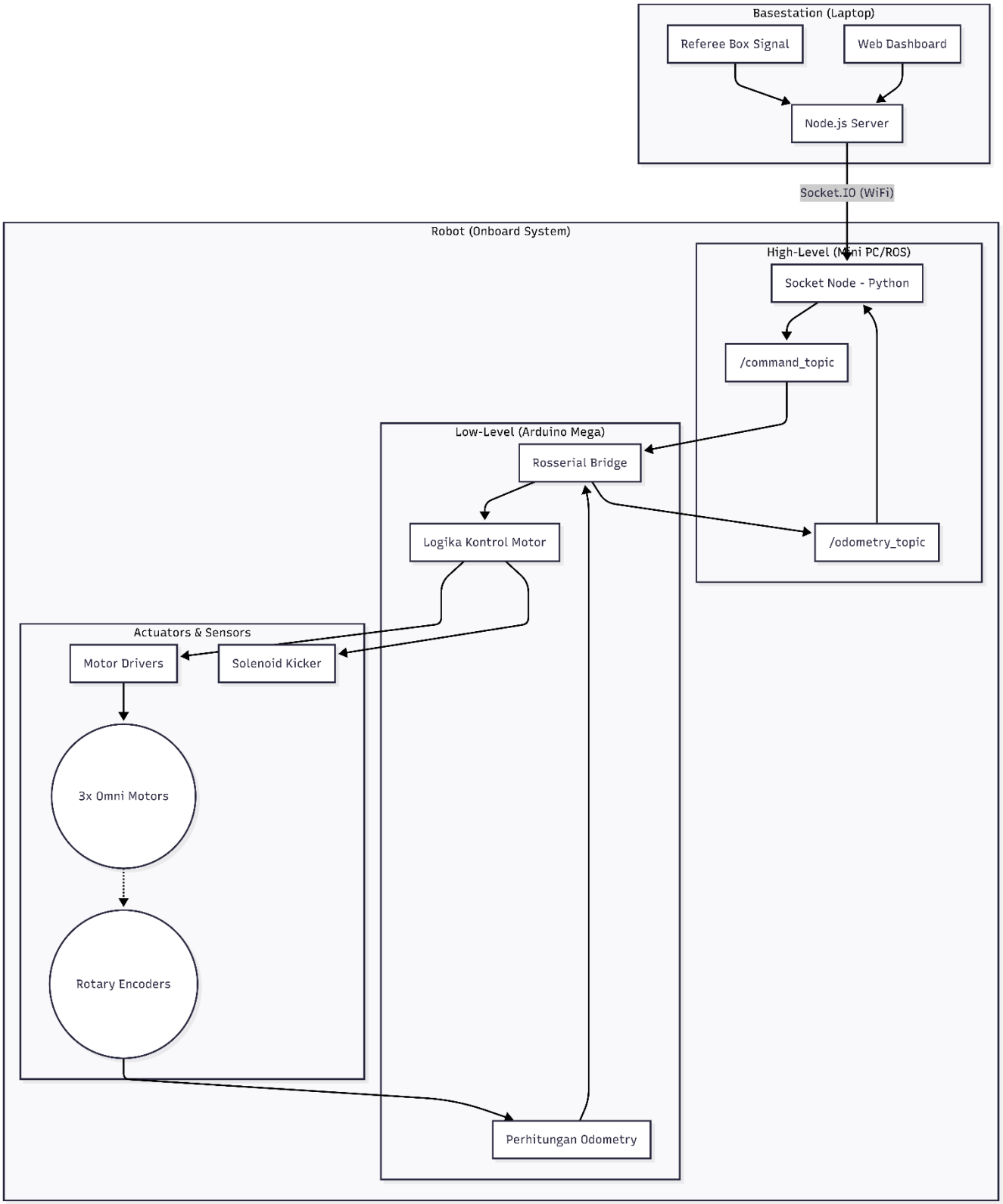
Dalam pengembangan sistem navigasi dan *tracking* objek pada robot sepak bola beroda ini, arsitektur perangkat lunak dirancang secara terdistribusi untuk memisahkan beban komputasi antara pemrosesan tingkat tinggi dan kontrol tingkat rendah. Pada sisi perangkat keras, robot menggunakan laptop ASUS K401U sebagai unit pemrosesan utama yang menjalankan program python berbasis ROS (*Robot Operating System*) untuk mengelola komunikasi antar-node, serta mikrokontroller Arduino Mega 2560 yang berfungsi sebagai pengendali motor dan sensor. Komunikasi antara kedua unit ini difasilitasi oleh `rosserial`, sementara interaksi dengan antarmuka pengguna (*Basestation*) dilakukan secara nirkabel menggunakan protokol `Socket.IO`. Untuk mendukung mobilitas yang tinggi di lapangan, robot dilengkapi dengan tiga unit motor DC yang disusun dalam konfigurasi roda *omni-directional*, memungkinkan pergerakan *holonomic* ke segala arah tanpa perlu mengubah orientasi hadap robot.

Alur kerja robot dimulai dari subsistem visi komputer yang bertugas menangkap citra lingkungan sekitar menggunakan kamera *webcam*. Data citra ini kemudian diproses oleh `node vision` pada ROS yang mengimplementasikan model *Deep Learning* YOLOv8 (*You Only Look Once*) untuk mendeteksi objek-objek krusial seperti bola, gawang, dan robot lawan secara *real-time*. Informasi koordinat objek yang diperoleh dari proses deteksi ini kemudian diterjemahkan menjadi perintah diskrit oleh algoritma navigasi, seperti `maju`, `kanan`, `kiri`, `putarKanan`, dan lain-lain. Perintah ini selanjutnya dikirim ke mikrokontroller untuk dikonversi

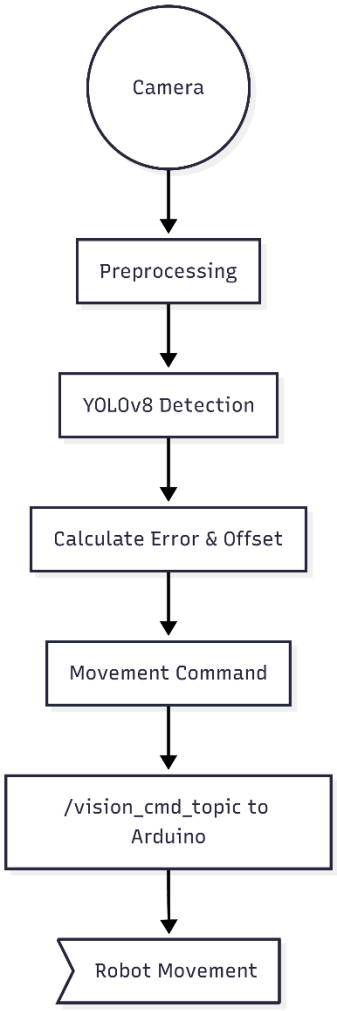
menjadi sinyal PWM bagi masing-masing motor, sehingga robot dapat bergerak menuju target atau melakukan manuver menghindar. Umpan balik dari *encoder* pada setiap roda digunakan untuk menghitung *odometry*, memberikan estimasi posisi robot di lapangan yang terus diperbarui untuk menjaga akurasi navigasi.

Salah satu tantangan utama dalam implementasi sistem visi berbasis *Deep Learning* pada perangkat bergerak adalah tingginya beban komputasi yang dapat menyebabkan latensi pada respons robot. Oleh karena itu, penelitian ini mengusulkan integrasi algoritma YOLOv8 dengan *Kalman Filter* untuk mengoptimalkan kinerja pelacakan objek. Dalam skema ini, YOLOv8 tidak dijalankan pada setiap *frame* video, melainkan hanya pada interval *frame* tertentu (misalnya setiap 3-5 *frame*) untuk mendapatkan pengukuran posisi aktual objek yang akurat. Pada *frame-frame* di antaranya, di mana deteksi YOLO tidak dilakukan untuk menghemat sumber daya komputasi, *Kalman Filter* mengambil alih peran estimasi dengan memprediksi posisi objek berdasarkan model pergerakan linier dari data sebelumnya.

Pendekatan hibrida ini memberikan solusi yang efisien untuk masalah pelacakan objek *real-time* pada robot sepak bola, di mana kecepatan dan ketepatan adalah faktor kunci. Ketika deteksi YOLO berhasil dilakukan, data posisi tersebut digunakan sebagai langkah koreksi (*measurement update*) untuk memperbaiki estimasi *state* pada *Kalman Filter*, mencegah terjadinya *drift* atau penyimpangan prediksi seiring berjalannya waktu. Sebaliknya, pada saat terjadi oklusi sesaat atau *motion blur* yang menyebabkan kegagalan deteksi YOLO, *Kalman Filter* tetap mampu memberikan estimasi posisi bola yang mulus berdasarkan momentum pergerakan sebelumnya. Dengan demikian, beban komputasi dapat dikurangi secara signifikan tanpa mengorbankan akurasi pelacakan, memungkinkan robot untuk merespons dinamika permainan dengan lebih cepat dan stabil.



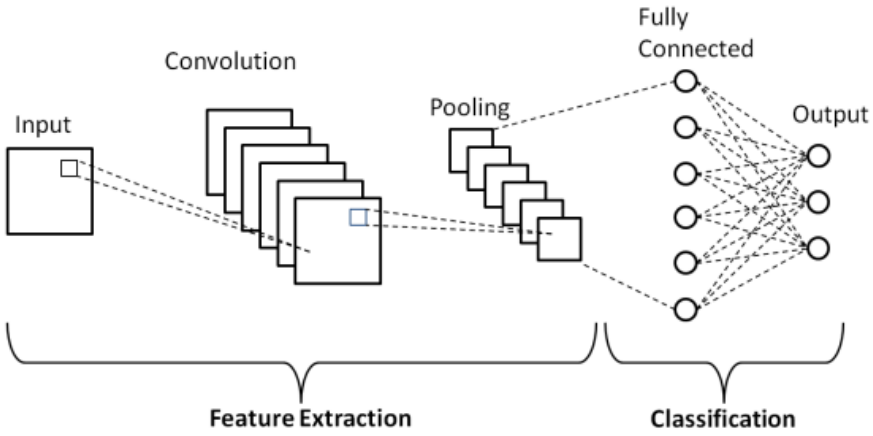
Gambar 2.3 Diagram Alur Kerja Robot



Gambar 2.4 Diagram Sistem *Vision*

2.4. *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) adalah salah satu jenis jaringan saraf tiruan yang dirancang khusus untuk mengolah data dengan struktur topologi berbentuk grid. Contohnya adalah data deret waktu yang dapat direpresentasikan sebagai grid satu dimensi, serta data citra yang tersusun dalam grid piksel dua dimensi. Istilah *convolutional* pada CNN merujuk pada penggunaan operasi matematika konvolusi di dalam arsitekturnya. Operasi konvolusi ini merupakan bentuk khusus dari operasi linier yang digunakan sebagai pengganti perkalian matriks konvensional pada satu atau lebih lapisan jaringan (Wakhidah et al., 2023).



Gambar 2.5 Arsitektur CNN

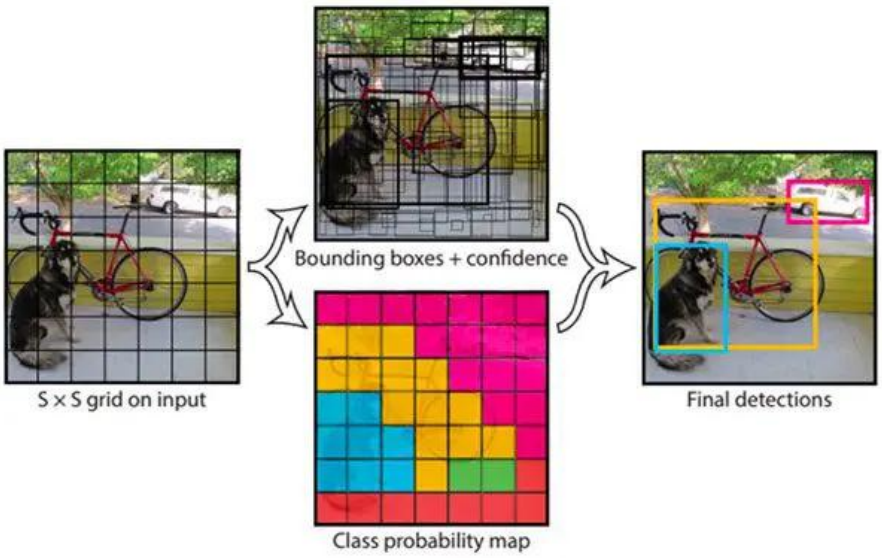
Sumber : analyticsvidhya.com, 2022

Pada CNN, setiap neuron direpresentasikan dalam bentuk dua dimensi sehingga metode ini sangat sesuai untuk pengolahan data citra. Arsitektur CNN umumnya terdiri atas beberapa tahap utama, yaitu lapisan *input*, proses ekstraksi fitur, tahap klasifikasi, dan lapisan *output*. Tahap ekstraksi fitur tersusun dari sejumlah lapisan tersembunyi yang meliputi lapisan konvolusi, fungsi aktivasi ReLU, serta lapisan *pooling*. Proses kerja CNN bersifat hierarkis, di mana keluaran dari lapisan konvolusi awal akan menjadi masukan bagi lapisan konvolusi berikutnya untuk mengekstraksi fitur yang semakin kompleks. Selanjutnya, pada tahap klasifikasi digunakan lapisan *fully connected* yang dipadukan dengan fungsi aktivasi *softmax* untuk menghasilkan keluaran berupa kelas atau label objek yang dikenali (Romario & Kadarina, 2020).

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

YOLO (*You Only Look Once*) merupakan salah satu model deteksi objek yang dikembangkan berdasarkan arsitektur *Convolutional Neural Network* (CNN). Model ini menerapkan pendekatan deteksi objek modern yang memungkinkan proses pendeteksian dilakukan secara *real-time*. Pada proses pengolahan citra, YOLO memiliki alur kerja yang relatif sederhana, di mana citra masukan terlebih dahulu diubah ukurannya agar sesuai dengan kebutuhan model. Selanjutnya, citra tersebut diproses menggunakan satu jaringan konvolusional secara menyeluruh untuk memprediksi lokasi dan kelas objek. Hasil deteksi kemudian diseleksi

menggunakan nilai ambang batas (*threshold*) berdasarkan tingkat kepercayaan (*confidence*) yang dihasilkan oleh model (Redmon et al., 2016).



Gambar 2.6 Ilustrasi Algoritma YOLO

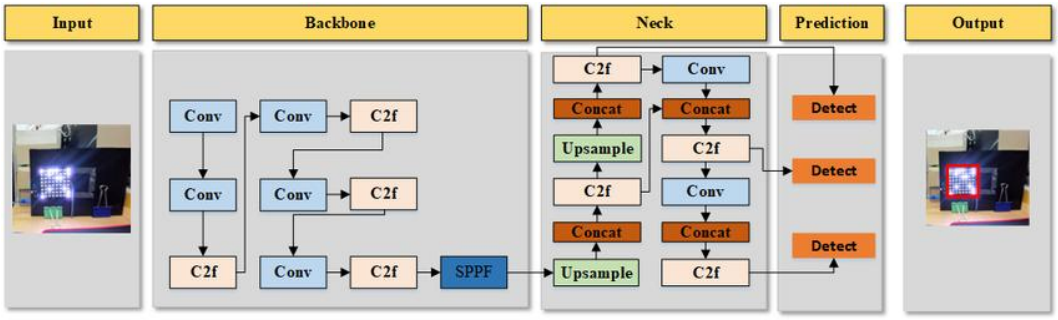
Sumber : Redmon et al., 2016

Gambar 2.3 menampilkan arsitektur dasar dari algoritma YOLO. Pada tahap awal, citra masukan dibagi ke dalam sejumlah grid berukuran $S \times S$. Setiap sel grid bertugas untuk memprediksi sejumlah *bounding box* B, beserta nilai *confidence* dan probabilitas kelas C. Nilai *confidence* menunjukkan tingkat keyakinan model terhadap keberadaan objek di dalam *bounding box* yang diprediksi. Setiap *bounding box* menghasilkan lima parameter, yaitu koordinat pusat objek (x, y), lebar (w), tinggi (h), serta nilai *confidence*. Koordinat x dan y merepresentasikan posisi pusat *bounding box* relatif terhadap sel grid, sedangkan w dan h menyatakan ukuran *bounding box* terhadap dimensi citra masukan. Seluruh hasil prediksi YOLO secara umum direpresentasikan dalam bentuk tensor berdimensi $[S, S, B \times 5 + C]$ (Redmon et al., 2016).

YOLOv8 merupakan algoritma *You Only Look Once* yang dikembangkan oleh Ultralytics untuk memberikan keseimbangan optimal antara kecepatan dan akurasi pada deteksi objek secara *real-time*. YOLOv8 adalah model yang akan digunakan penelitian ini. Arsitektur model ini dibangun sebagai sebuah jaringan saraf tunggal yang terdiferensiasi secara *end-to-end*, yang secara garis besar terdiri

dari tiga komponen utama: *Backbone*, *Neck*, dan *Head*. Pada bagian *Backbone*, YOLOv8 menggunakan struktur *Convolutional Neural Network* (CNN) canggih yang mengintegrasikan modul *Cross Stage Partial* (CSP) *bottleneck* untuk mengekstraksi fitur multi-skala dari citra masukan sekaligus mengurangi redundansi komputasi dan meningkatkan penggunaan kembali fitur.

Selanjutnya, bagian *Neck* pada YOLOv8 berfungsi untuk menyempurnakan dan menggabungkan fitur-fitur multi-skala yang telah diekstraksi melalui optimasi *Path Aggregation Network* (PANet) dan *Feature Pyramid Network* (FPN). Integrasi fitur ini sangat krusial untuk meningkatkan aliran informasi antar level fitur, sehingga model mampu mendeteksi objek dengan berbagai ukuran dan konteks yang berbeda secara lebih efektif. Inovasi paling signifikan pada YOLOv8 terletak pada bagian *Head*, di mana model ini beralih dari metode berbasis *anchor* ke pendekatan *anchor-free* dalam prediksi *bounding box*. Pendekatan ini menyederhanakan proses prediksi dengan menghilangkan kebutuhan akan kotak acuan (*anchor boxes*) yang ditentukan sebelumnya, sehingga meningkatkan fleksibilitas model terhadap objek dengan rasio aspek yang bervariasi serta mengurangi jumlah hiperparameter yang harus dikelola.



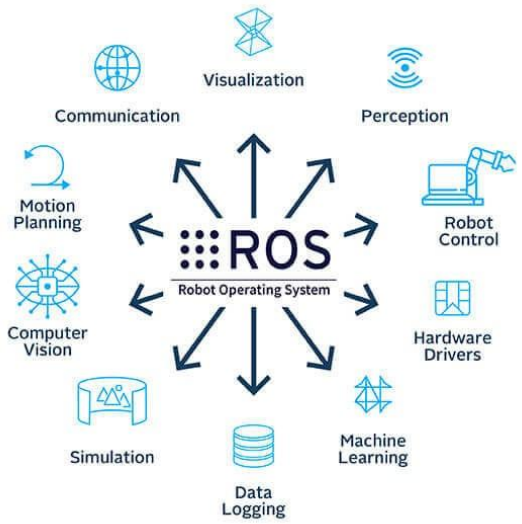
Gambar 2.7 Arsitektur YOLOv8

Sumber : Yaseen, 2024

2.6. *Robot Operating System* (ROS)

Robot Operating System (ROS) merupakan sebuah sistem operasi robot yang bersifat *open-source*. Namun, penting untuk dipahami bahwa ROS bukanlah sistem operasi dalam pengertian tradisional yang menangani manajemen proses dan penjadwalan seperti Windows atau Linux. Sebaliknya, ROS berfungsi sebagai

lapisan komunikasi terstruktur (*structured communications layer*) yang berjalan di atas sistem operasi *host* pada kluster komputasi yang heterogen. ROS menyediakan kerangka kerja yang fleksibel untuk penulisan perangkat lunak robot, yang mencakup berbagai *tools* dan pustaka (*libraries*) yang bertujuan untuk menyederhanakan tugas pembuatan perilaku robot yang kompleks dan kuat di berbagai platform robotik (Quigley et al., 2009).



Gambar 2.7 ROS (*Robot Operating System*)

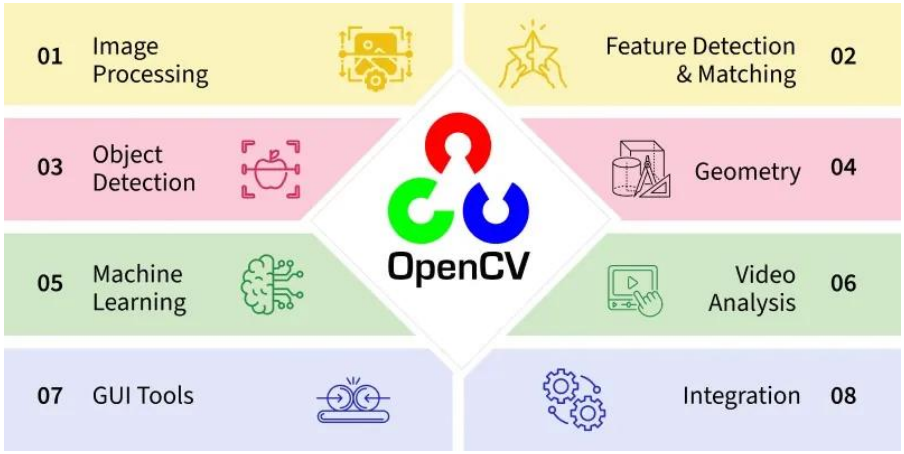
Sumber : Louda et al., 2023

Pengembangan ROS dilatarbelakangi oleh tantangan besar dalam penulisan perangkat lunak untuk robot, di mana skala dan cakupan robotika terus berkembang pesat. Berbagai jenis robot memiliki perangkat keras yang sangat bervariasi, membuat penggunaan ulang kode (*code reuse*) menjadi hal yang tidak trivial. Selain itu, besarnya ukuran kode yang diperlukan sering kali melampaui kemampuan peneliti tunggal, sehingga diperlukan arsitektur yang mendukung upaya integrasi perangkat lunak berskala besar. ROS dirancang untuk menjawab tantangan spesifik yang ditemui saat mengembangkan robot layanan berskala besar sebagai bagian dari proyek STAIR (*STanford AI Robot*) di Stanford University dan Program Robot Personal di Willow Garage. Meskipun bermula dari domain robot layanan dan manipulasi seluler, arsitektur yang dihasilkan bersifat jauh lebih umum dan dapat diterapkan pada berbagai sistem robotik lainnya (Quigley et al., 2009).

Secara teknis, implementasi ROS berpusat pada konsep *nodes*, *messages*, *topics*, dan *services*. *Node* adalah proses yang melakukan komputasi. Komunikasi antar *node* terjadi dengan mengirimkan *messages* (struktur data yang diketik secara ketat). Pengiriman pesan ini umumnya menggunakan mekanisme *publish-subscribe* melalui *topic*, di mana sebuah *node* mempublikasikan pesan ke topik tertentu dan *node* lain yang berlangganan topik tersebut akan menerimanya. Selain itu, untuk transaksi sinkron, ROS menyediakan mekanisme *service* yang berbasis permintaan dan respons (*request-response*) (Quigley et al., 2009).

2.7. OpenCV v3.4 (*Open Source Computer Vision version* 3.4)

OpenCV (*Open Source Computer Vision*) adalah sebuah pustaka perangkat lunak yang dirancang khusus untuk pengolahan citra secara *real-time*. Pustaka ini awalnya dikembangkan oleh Intel dan kini didukung oleh Willow Garage serta Itseez. OpenCV menyediakan antarmuka yang kompatibel dengan berbagai bahasa pemrograman seperti C++, C, Python, dan Java, serta dapat dijalankan pada beragam sistem operasi, di antaranya Windows, Linux, Mac OS, iOS, dan Android. Pustaka ini dirancang untuk efisiensi komputasi dengan fokus utama pada aplikasi berbasis *real-time* (C. Saputra, 2023).



Gambar 2.8 Kegunaan OpenCV

Sumber : geeksforgeeks.org, 2025

Modul pustaka OpenCV dirancang dengan tingkat fleksibilitas dan ketangguhan yang tinggi untuk menangani berbagai permasalahan dalam bidang *computer vision*. Berbagai solusi telah tersedia di dalamnya, seperti pemotongan

citra (*cropping*), peningkatan kualitas citra melalui penyesuaian kecerahan, ketajaman, dan kontras, pendeteksian bentuk, segmentasi citra, pelacakan objek bergerak, hingga pengenalan objek, serta beragam fungsi lainnya (Ratna, 2020).

2.8. NumPy v2.3 (*Numerical Python version* 2.3)

Gambar 2.9 Logo NumPy

Sumber : python.plainenglish.io, 2025

NumPy (*Numerical Python*) adalah paket dasar dan fundamental untuk komputasi ilmiah dengan Python. Sebagai pustaka *open-source*, NumPy menyediakan objek *array* N-dimensi (*ndarray*) yang sangat kuat, yang berfungsi sebagai struktur data universal untuk pertukaran data multi-dimensi dalam ekosistem ilmiah Python. Objek *ndarray* ini memungkinkan operasi yang cepat dan efisien pada data homogen (semua elemen harus bertipe data yang sama) dalam jumlah besar, sebuah keunggulan signifikan dibandingkan list bawaan Python, yang dicapai melalui inti kode yang dioptimalkan dalam bahasa C. Selain struktur *array* multidimensi, NumPy juga menawarkan koleksi lengkap fungsi matematika tingkat tinggi, termasuk aljabar *linear*, transformasi Fourier, dan kemampuan angka acak, menjadikannya standar *de-facto* untuk komputasi *array* dan menjadi fondasi bagi pustaka ilmu data dan *machine learning* lainnya (NumPy, 2025).

2.9. *Root Mean Square Error* (RMSE)

Root Mean Square Error (RMSE) adalah metode pengukuran yang menghitung akar kuadrat dari rata-rata kuadrat selisih antara nilai prediksi ($\hat{y}$) dan nilai observasi aktual atau ground truth (y). Menurut Hodson (2022), RMSE merupakan metrik yang optimal untuk mengevaluasi model yang memiliki distribusi *error* bersifat normal atau Gaussian.

Karakteristik utama RMSE adalah sensitivitasnya terhadap kesalahan yang besar (*outliers*). Karena selisih *error* dikuadratkan sebelum dirata-ratakan, kesalahan besar akan memberikan penalti yang lebih berat dibandingkan kesalahan kecil (Hodson, 2022). Dalam konteks robotika sepak bola, karakteristik ini penting karena kesalahan posisi yang ekstrem (misalnya robot kehilangan jejak bola) dianggap jauh lebih fatal daripada deviasi kecil yang konsisten.

Secara matematis, RMSE dirumuskan sebagai berikut (Hodson, 2022):

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

Keterangan:

n : Jumlah sampel data atau *frame*

$\hat{y}_i$: Nilai posisi hasil prediksi *Kalman Filter* pada frame ke-i.

y_i : Nilai posisi aktual (*ground truth*) pada frame ke-i.

Untuk mengukur efektivitas sistem pelacakan objek dan kualitas prediksi yang dihasilkan oleh algoritma *Kalman Filter*, diperlukan metode evaluasi statistik yang objektif. Salah satu metrik standar yang paling umum digunakan dalam pengukuran akurasi model adalah RMSE (Hodson, 2022).

2.10. *Kalman Filter*

Kalman Filter merupakan algoritma yang sangat kuat untuk memperkirakan keadaan suatu sistem dinamis berdasarkan serangkaian pengukuran yang tidak lengkap dan mengandung *noise*. Algoritma ini dikembangkan oleh Rudolf E. Kálmán, dan awalnya digunakan dalam bidang navigasi penerbangan. Namun, seiring waktu, *Kalman Filter* menjadi komponen penting di berbagai bidang, seperti robotika, ekonomi, dan terutama *computer vision* (CV). *Kalman Filter* bekerja melalui dua tahap utama yang berlangsung secara berulang, yaitu tahap prediksi dan tahap pembaruan (koreksi). *Kalman Filter* mampu menghasilkan estimasi keadaan objek, seperti posisi dan kecepatannya, secara halus, akurat, dan stabil, meskipun data sensor yang diterima tidak sempurna (Ultralytics, 2025).

Kalman Filter dinamai dari penciptanya, Rudolf Kalman. Algoritma ini bekerja dengan menerima sejumlah data yang mengandung *noise*, kemudian menyaringnya untuk meminimalkan gangguan tersebut. Karena dirancang pada ruang *linear*, *Kalman Filter* juga dikenal dengan istilah *linear quadratic estimation* (Sholehurrohman et al., 2023). Metode *Kalman Filter* menggunakan informasi dari objek yang terdeteksi di suatu *frame* dan status objek dari *frame* sebelumnya untuk mendapatkan status yang baru dari objek tersebut (C. Saputra, 2023).

Dalam konteks AI, *Kalman Filter* banyak digunakan dalam sistem pelacakan objek (*object tracking*). Setelah model deteksi seperti Ultralytics YOLO berhasil mengidentifikasi objek pada suatu *frame*, *Kalman Filter* digunakan untuk memperkirakan posisi objek pada *frame* berikutnya. Perkiraan ini didasarkan pada model gerak (*motion model*), yang umumnya mengasumsikan bahwa objek bergerak dengan kecepatan atau percepatan konstan (Ultralytics, 2025).

Ketika *frame* berikutnya diterima, model deteksi memberikan hasil pengukuran baru berupa koordinat *bounding box* objek. *Kalman Filter* kemudian menjalankan tahap pembaruan (*update*), yaitu memperbaiki hasil prediksi awal berdasarkan data baru tersebut. Proses ini sangat efektif karena beberapa alasan penting:

- **Reduksi *noise*:** *Kalman Filter* mampu menghaluskan hasil deteksi yang bergetar atau tidak stabil, sehingga jalur pelacakan menjadi lebih halus dan konsisten.
- **Penanganan *occlusion*:** Saat objek tidak terdeteksi selama beberapa *frame*, *Kalman Filter* tetap dapat memperkirakan posisinya. Dengan demikian, ketika objek muncul kembali, sistem dapat langsung mengenalinya tanpa kehilangan jejak.
- **Estimasi keadaan (*state estimation*):** Selain posisi, *Kalman Filter* juga mampu memperkirakan parameter lain seperti kecepatan dan arah gerak objek.

Banyak algoritma pelacakan modern seperti BoT-SORT dan ByteTrack menggunakan *Kalman Filter* sebagai komponen inti prediksi gerakannya. Bahkan, model Ultralytics terbaru seperti YOLO11 juga memanfaatkan pendekatan ini

dalam mode pelacakannya. Ultralytics *framework* menggunakan *Kalman Filter* dalam algoritma *object tracking*, sementara pustaka OpenCV juga menyediakan modul *Kalman Filter* yang banyak digunakan untuk pelacakan objek secara *real-time* dalam aplikasi *computer vision* (Ultralytics, 2025).

Secara sederhana, berikut merupakan persamaan matematika algoritma *Kalman Filter* (Yuztiawan & Utaminingrum, 2017):

$$\hat{X}_{\bar{k}} = K_k \, . \, Z_k + (1 - K_k) \, . \hat{X}_{k-1}$$

Keterangan:

$\hat{X}_{\bar{k}}$: Estimasi saat ini

K_k : *Kalman Gain*

Z_k : Nilai dari hasil pendeteksian prediksi

$\hat{X}_{k-1}$: Estimasi sebelumnya

Kalman Filter beroperasi sebagai estimator jenis *predictor-corrector* yang bekerja dalam dua langkah utama: Prediksi dan Pembaruan (Ultralytics, 2025). Langkah prediksi menggunakan estimasi keadaan dari langkah waktu sebelumnya $(k - 1)$ untuk memproyeksikan atau memprediksi keadaan sistem saat ini (X_k^-).

- **Tujuan:** Memperkirakan di mana sistem akan berada pada langkah waktu k berdasarkan dinamika sistem yang diketahui dan *input* kontrol yang diterapkan.
- **Proses:** Filter memproyeksikan *mean* keadaan (X) dan kovariansi keadaan (P) dari langkah waktu sebelumnya ke langkah waktu saat ini.
- **Kovariansi:** Dalam proses prediksi, ketidakpastian (diwakili oleh kovariansi P) diperbesar oleh *noise* proses (Q).

Fungsi `kf_predict` menghitung mean keadaan (X) dan kovariansi (P) yang diprediksi.

```
from numpy import dot

def kf_predict(X, P, A, Q, B, U):
    # Prediksi Keadaan: X = (A * X_sebelum) + (B * U)
    X = dot(A, X) + dot(B, U)
    # Prediksi Kovariansi: P = (A * P_sebelum * A.T) + Q
    P = dot(A, dot(P, A.T)) + Q
    return(X,P)
```

Kode 2.1 Contoh kode prediksi *Kalman Filter* dengan NumPy

Setelah prediksi selesai, langkah pembaruan menggunakan nilai yang baru (Y_k) untuk mengoreksi prediksi tersebut.

- **Tujuan:** Menghitung estimasi akhir yang diperbarui untuk keadaan (X_k) dan kovariansi (P_k) pada langkah waktu k setelah melihat pengukuran.
- **Residual/Inovasi (V_k):** Langkah pertama adalah menghitung residual atau inovasi. Ini adalah perbedaan antara pengukuran aktual (Y_k) dan pengukuran yang diprediksi (berdasarkan keadaan yang diprediksi (X_k^-)).
- ***Kalman Gain* (K_k):** Residual kemudian dikalikan dengan *Kalman Gain* (K_k). *Kalman Gain* bertindak sebagai "berat" (*weight*) yang menentukan seberapa besar prediksi harus dikoreksi oleh pengukuran baru.
  - Jika *noise* pengukuran (R) tinggi, *gain* akan kecil, artinya filter lebih memercayai prediksinya sendiri.
  - Jika kovariansi prediksi (P^-) tinggi (ketidakpastian prediksi besar), *gain* akan besar, artinya filter lebih memercayai pengukuran baru.
- **Koreksi:** Hasil koreksi ini ditambahkan ke keadaan yang diprediksi (X_k^-) untuk mendapatkan estimasi keadaan akhir (X_k). Kovariansi juga diperbarui (P_k) untuk mencerminkan ketidakpastian yang telah berkurang.

Fungsi `kf_update` menghitung *gain* dan mengoreksi *mean* (X) serta kovariansi (P).

```
from numpy import dot, linalg
from numpy.linalg import inv

def kf_update (X, P, Y, H, R):
    # Mean Prediksi Pengukuran (IM): H * X_prediksi
    IM = dot(H, X)
    # Kovariansi Prediksi Pengukuran (IS): H * P_prediksi * H.T + R
    IS = dot(H, dot(P, H.T)) + R
    # Kalman Gain (K): P_prediksi * H.T * inv(IS)
    K = dot(P, dot(H.T, inv(IS)))
    # Pembaruan Keadaan: X_baru = X_prediksi + K * (Y - IM)
    X = X + dot(K, (Y - IM))
    # Pembaruan Kovariansi: P_baru = P_prediksi - K * IS * K.T
    P = P - dot(K, dot(IS, K.T))

    return (X, P, K, IM, IS, LH) # Mengembalikan mean dan kovariansi
yang diperbarui
```

Kode 2.2 Contoh kode perbaruan *state Kalman Filter* dengan NumPy

2.11. *Frame Skipping* dalam Pelacakan Objek

Pelacakan objek (*object tracking*) merupakan salah satu aspek penting dalam bidang visi komputer. Namun, banyak metode pelacakan modern yang berbasis *deep learning* memiliki kompleksitas tinggi dan memerlukan kapasitas komputasi yang besar. Pada sistem dengan keterbatasan perangkat keras, seperti robot atau perangkat bergerak, penggunaan algoritma pelacakan yang terlalu berat dapat menghabiskan sebagian besar sumber daya CPU maupun GPU. Kondisi ini berpotensi mengganggu kinerja sistem secara keseluruhan, terutama ketika terdapat proses lain yang harus dijalankan secara bersamaan. Strategi *frame skipping* diterapkan sebagai bentuk optimasi untuk meningkatkan efisiensi pemrosesan dengan tidak mengeksekusi algoritma deteksi atau pelacakan yang berat pada setiap *frame* video. Dengan menurunkan frekuensi penggunaan algoritma utama, beban komputasi pada perangkat keras dapat ditekan secara signifikan. Pendekatan ini memungkinkan peningkatan kinerja sistem secara keseluruhan, khususnya dalam hal peningkatan *frame rate* (FPS) selama proses pemrosesan berlangsung (Lee, 2024).

Dalam pendekatan tradisional (M_{SKIP}), sistem hanya melakukan proses deteksi penuh menggunakan algoritma utama pada interval *frame* tertentu. Misalnya, jika parameter jumlah pelompatan *frame* (S_N) diatur sebesar 5, maka sistem hanya akan menjalankan deteksi pada *frame* ke-1, 6, 11, dan seterusnya. Pada metode tradisional yang bersifat statis, posisi objek pada *frame* yang dilompati sering kali hanya disalin (*copy-paste*) dari hasil *frame* sebelumnya tanpa adanya pembaruan posisi aktif. Meskipun metode ini mampu meningkatkan kecepatan hingga berkali-kali lipat (contoh: dari 58.3 FPS menjadi 428.2 FPS), terdapat risiko teknis berupa penurunan akurasi dan ketangguhan (*robustness*) pelacakan karena objek yang bergerak cepat akan melampaui posisi *bounding box* lama sebelum deteksi berikutnya dilakukan (Lee, 2024).

Untuk mengatasi kelemahan metode tradisional, penelitian terbaru mengusulkan pendekatan dua tingkat. Strategi ini membagi proses pelacakan menjadi dua kategori utama:

1. **Pelacak Kuat (*Robust Tracker*):** Algoritma deteksi yang akurat namun berat (seperti YOLO), digunakan untuk inisialisasi dan koreksi pada skenario sulit.
2. **Pelacak Ringan (*Lightweight Tracker*):** Algoritma dengan beban komputasi rendah yang bertugas menjaga kontinuitas pelacakan selama fase *frame skipping*.

Penggunaan pelacak ringan bertujuan untuk memastikan bahwa selama algoritma berat "beristirahat", posisi objek tetap diperbarui berdasarkan estimasi pergerakan, bukan sekadar statis.

Dalam penelitian ini, mekanisme *frame skipping* dioptimalkan dengan mengintegrasikan *Kalman Filter* sebagai pelacak tingkat ringan. Berbeda dengan pelompatan *frame* tradisional yang bersifat indiskriminat, integrasi ini memungkinkan sistem untuk melakukan estimasi lintasan objek secara aktif selama fase pelompatan. *Kalman Filter* berperan memprediksi koordinat objek pada setiap *frame* yang dilewati oleh detektor YOLO. Hal ini memberikan dua keuntungan utama secara ilmiah:

- **Reduksi *Error* Akumulatif:** Menghindari kegagalan pelacakan (*lost tracking*) yang sering terjadi pada metode M_{SKIP} tradisional akibat pergerakan objek yang dinamis.
- **Stabilitas Kecepatan:** Menjaga *frame rate* tetap tinggi karena kalkulasi matriks pada Kalman Filter jauh lebih ringan dibandingkan proses inferensi *neural network* pada YOLO.

Meskipun pelacak ringan dapat menjaga posisi objek, ketergantungan penuh pada estimasi dalam waktu lama dapat menyebabkan fenomena *drift* (pergeseran posisi). Oleh karena itu, diterapkan mekanisme pemicu paksa (*forced invocation*) berdasarkan parameter S_N (Lee, 2024). Setelah sistem memproses sejumlah S_N *frame* berturut-turut menggunakan *Kalman Filter*, algoritma YOLO akan dipaksa

berjalan satu kali untuk melakukan sinkronisasi ulang posisi dan ukuran *bounding box* asli objek guna menjamin reliabilitas pelacakan jangka panjang.

2.12. Hipotesis Penelitian

Berdasarkan rumusan masalah dan tinjauan pustaka yang telah diuraikan, maka hipotesis yang diajukan dalam penelitian ini adalah:

- **H_0 (Hipotesis Nol):** Penerapan algoritma *Kalman Filter* pada sistem deteksi objek berbasis YOLO tidak memberikan pengaruh yang signifikan terhadap stabilitas gerakan robot dan akurasi pelacakan bola pada Robot Sepak Bola Beroda.
- **H_1 (Hipotesis Alternatif):** Penerapan algoritma Kalman Filter pada deteksi objek berbasis YOLOv8 menghasilkan stabilitas gerakan robot yang lebih baik (pengurangan *jitter*) dan akurasi pelacakan yang lebih tinggi (RMSE lebih rendah) dibandingkan dengan deteksi murni YOLOv8, terutama saat terjadi *occlusion* (halangan).

Untuk menentukan penerimaan atau penolakan terhadap hipotesis yang diajukan, dilakukan perbandingan statistik deskriptif antara hasil pengujian pada kondisi *baseline* (YOLO murni) dan kondisi *optimized* (YOLO + *Kalman Filter*). Hipotesis nol (H_0) akan ditolak dan hipotesis alternatif (H_1) akan diterima apabila hasil eksperimen memenuhi kriteria sebagai berikut:

- **Peningkatan Akurasi:** Nilai rata-rata *Root Mean Square Error* (RMSE) pada sistem *optimized* lebih kecil dibandingkan sistem *baseline* ($\mu_{RMSE\_opt} < \mu_{RMSE\_base}$), yang mengindikasikan bahwa prediksi posisi bola menjadi lebih dekat dengan posisi sebenarnya.
- **Peningkatan Stabilitas Gerak:** Frekuensi perubahan perintah gerak (*command switching frequency*) pada log sistem *optimized* lebih rendah dibandingkan sistem *baseline*, yang menandakan berkurangnya *jitter* atau osilasi pada respons robot.
- **Efisiensi Komputasi:** Sistem *optimized* mampu mempertahankan rata-rata *Frame Rate* (FPS) yang setara atau lebih tinggi dibandingkan batas minimum

operasional *real-time* robot (>10 FPS), membuktikan bahwa penambahan algoritma *Kalman Filter* tidak membebani kinerja laptop secara signifikan.

Jika hasil pengujian tidak memenuhi kriteria-kriteria tersebut, maka H_0 diterima, yang berarti penambahan algoritma *Kalman Filter* tidak memberikan dampak perbaikan yang signifikan pada sistem.

BAB III
METODOLOGI PENELITIAN

3.1. Metode Penelitian

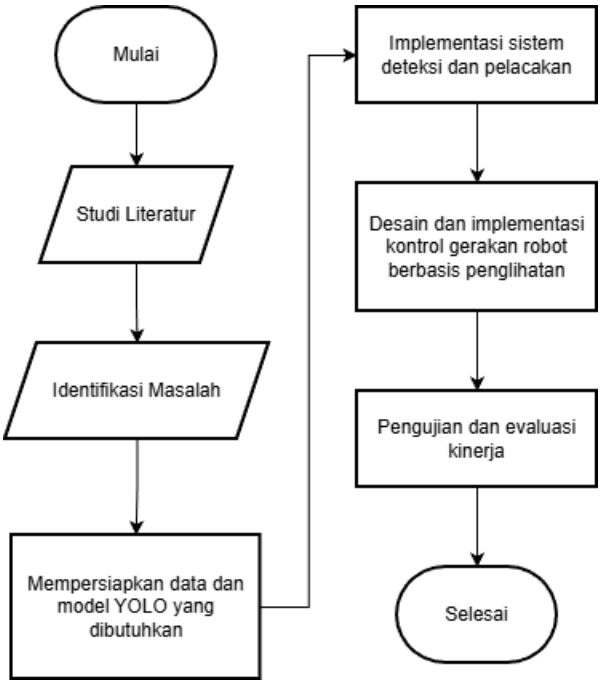
Penelitian ini dilaksanakan menggunakan pendekatan kuantitatif dengan strategi penelitian eksperimental (*Experimental Research*). Pendekatan kuantitatif dipilih karena penelitian ini berfokus pada pengukuran data numerik yang objektif untuk menguji hipotesis kinerja sistem. Menurut Sugiyono (2013), metode kuantitatif didefinisikan sebagai metode penelitian yang berlandaskan pada filsafat positivisme, digunakan untuk meneliti pada populasi atau sampel tertentu dengan tujuan untuk menguji hipotesis yang telah ditetapkan. Sementara itu, strategi eksperimental diterapkan untuk mengetahui hubungan sebab-akibat antar variabel melalui manipulasi kondisi terkontrol. Sebagaimana dinyatakan oleh Sugiyono (2013), metode eksperimental adalah metode penelitian yang digunakan untuk mencari pengaruh perlakuan tertentu terhadap yang lain dalam kondisi yang terkendalikan. Dalam konteks penelitian ini, strategi tersebut digunakan untuk mengukur dan membandingkan performa antara sistem *tracking* objek murni berbasis YOLO (*baseline*) dengan sistem yang dioptimasi menggunakan *Kalman Filter* (*optimized*).

Data yang dikumpulkan dalam penelitian ini sepenuhnya bersifat numerik dan difokuskan pada tiga parameter evaluasi utama. Parameter pertama adalah metrik komputasi, yang mencakup pengukuran *Frames Per Second* (FPS) dan waktu inferensi (*inference time*) guna memvalidasi efisiensi sistem pada perangkat keras laptop ASUS K401U. Parameter kedua berfokus pada akurasi pelacakan, di mana *Root Mean Square Error* (RMSE) digunakan untuk mengukur deviasi posisi prediksi terhadap posisi sebenarnya (*ground truth*), serta tingkat keberhasilan pelacakan saat terjadi halangan (*occlusion rate*). Parameter ketiga adalah metrik respons gerak yang menganalisis stabilitas log perintah (*command logs*) yang dikirimkan dari ROS ke mikrokontroler untuk mendeteksi adanya pengurangan *jitter* atau perubahan perintah yang tidak perlu. Pengujian eksperimental ini akan diimplementasikan langsung pada Robot Sepak Bola Beroda dalam berbagai

skenario dinamis, meliputi uji lintasan lurus, uji *occlusion*, dan uji guncangan (*jitter test*), untuk membuktikan secara empiris efektivitas algoritma yang diusulkan.

3.2. Kerangka Pikiran

Pada penelitian ini akan dilakukan beberapa tahap yang digambarkan pada gambar 3.1, yaitu dimulai dengan studi literatur, lalu mengidentifikasi masalah, mempersiapkan data dan model YOLO yang dibutuhkan, implementasi sistem deteksi dan pelacakan, desain dan implementasi kontrol gerakan robot berbasis penglihatan, pengujian dan evaluasi kinerja, dan selesai.



Gambar 3.1 Kerangka Pikiran Penelitian

3.3. Studi Literatur

Tahap studi literatur dalam metodologi penelitian ini bertujuan untuk mengumpulkan, menganalisis, dan mensintesis informasi teoretis dan praktis yang relevan dan dibutuhkan untuk perancangan serta implementasi sistem optimasi gerakan robot. Proses ini memastikan bahwa metode yang dipilih, khususnya integrasi YOLO dan *Kalman Filter*, didasarkan pada prinsip-prinsip teknis yang solid dan terkini.

Studi literatur difokuskan pada tiga area utama yang secara langsung mendukung perancangan sistem:

3.3.1. Algoritma Deteksi Objek

Mengidentifikasi model YOLO yang paling sesuai untuk kebutuhan *real-time* dengan keterbatasan komputasi pada robot KRSBI Beroda, yaitu YOLOv8. Kemudian mempelajari format *output* data deteksi dari YOLO (koordinat pusat x, y, lebar w, dan tinggi h *bounding box*) untuk dikonversi menjadi *input* data pengukuran yang dapat diproses oleh *Kalman Filter*.

3.3.2. Algoritma Prediksi dan Pelacakan

Mengkaji prinsip dasar *Kalman Filter*, terutama dalam konteks pelacakan objek bergerak (*object tracking*). Kemudian mempelajari implementasi *Kalman Filter* menggunakan pustaka yang efisien, seperti NumPy dan modul yang tersedia pada OpenCV atau Ultralytics untuk memastikan kompatibilitas dan efisiensi komputasi *real-time*.

3.3.3. Sistem Robotika dan Kontrol Gerak

Mempelajari spesifikasi teknis Robot Sepak Bola Beroda, khususnya mengenai sistem penglihatan (kamera *omnidirectional*) dan aktuator yang menentukan batasan dan kemampuan sistem kontrol gerak. Kemudian mengumpulkan informasi mengenai konversi data posisi terprediksi (*output Kalman Filter*) menjadi perintah gerakan yang akan dieksekusi oleh robot.

3.4. Identifikasi Masalah

Identifikasi masalah ini berasal dari celah (*gap*) antara metode deteksi berbasis YOLO yang memiliki beban komputasi tinggi dan kebutuhan akan gerakan robot yang stabil dan *real-time*. Masalah-masalah teknis yang perlu dipecahkan dalam penelitian ini meliputi:

3.4.1. Ketidakstabilan Data Deteksi Akibat Keterbatasan Komputasi

Model deteksi YOLO membutuhkan kapasitas komputasi yang tinggi. Keterbatasan perangkat keras pada robot KRSBI Beroda seringkali menyebabkan penurunan *frame rate* atau keterlambatan proses inferensi. Masalah ini menghasilkan data posisi bola yang tidak stabil atau terlambat diterima oleh sistem

kontrol, yang pada akhirnya mengakibatkan pergerakan robot menjadi tidak mulus atau tidak responsif.

3.4.2. Kerentanan Terhadap *Occlusion* dan *False Negative Detection*

Pada kondisi pertandingan yang dinamis, sering terjadi *occlusion* (bola tertutup sebagian) yang menyebabkan *false-negative detection* (bola tidak terdeteksi). Tanpa mekanisme prediksi, robot akan berhenti bergerak atau kehilangan jejak bola saat *occlusion* sesaat terjadi, menghambat kontinuitas permainan.

3.4.3. Kebutuhan untuk Prediksi Posisi Gerak Dinamis

Detektor berbasis *frame* tunggal (YOLO) hanya memberikan posisi bola saat ini tanpa mempertimbangkan dinamika gerak bola tersebut. Untuk mengoptimalkan gerakan robot yang responsif terhadap perubahan posisi bola yang cepat dan mendadak, dibutuhkan sebuah mekanisme prediksi yang mampu mengestimasi posisi bola di *frame* berikutnya.

Tiga masalah teknis di atas akan diselesaikan melalui implementasi dan integrasi *Kalman Filter* pada *output* deteksi YOLO. *Kalman Filter* berperan sebagai estimasi rekursif untuk memprediksi posisi, menghaluskan data yang bising (*noisy*), dan mengatasi data yang hilang (*occlusion*).

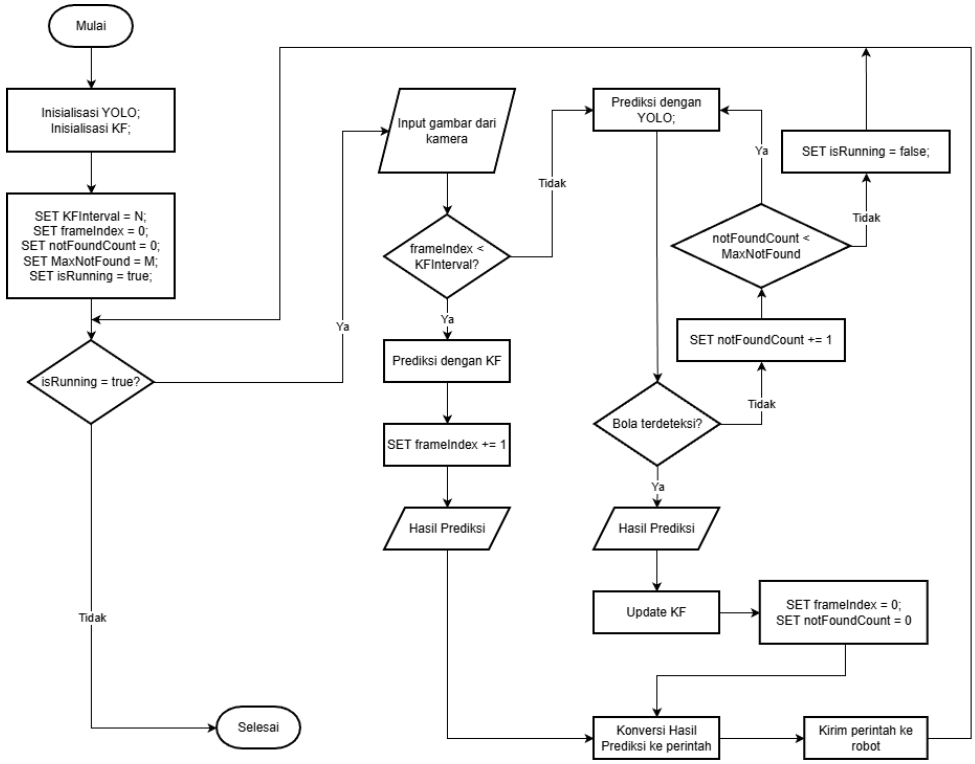
3.5. Persiapan Data dan Konfigurasi Model Deteksi

Model deteksi yang digunakan dalam penelitian ini telah dilatih sebelumnya menggunakan kumpulan data citra yang diambil langsung dari kamera *omnidirectional* robot KRSBI. *Dataset* ini dikumpulkan untuk merepresentasikan berbagai variasi pencahayaan dan posisi bola di lapangan, guna memastikan model memiliki ketahanan terhadap perubahan lingkungan. Proses pra-pemrosesan data melibatkan pemberian label (*labelling*) pada objek bola menggunakan *bounding box*. Keluaran dari proses pelabelan ini menghasilkan koordinat *ground truth* yang kemudian digunakan untuk pelatihan model *deep learning*.

Model YOLO bertanggung jawab untuk menghasilkan prediksi lokasi bola yang kemudian menjadi input bagi algoritma pelacakan *Kalman Filter*. Setelah model YOLO berhasil mendeteksi bola, data hasil deteksi diekstraksi. Data ini

berbentuk bounding box dengan format (x_c, y_c, w, h, C), di mana (x_c, y_c) adalah koordinat pusat bola pada citra, dan C adalah *confidence score*. Koordinat pusat (x_c, y_c) inilah yang dijadikan Vektor Pengukuran (Z_k) yang bising (*noisy*) dan siap diolah lebih lanjut oleh *Kalman Filter*. *Confidence score* juga dapat dimanfaatkan untuk memverifikasi keandalan pengukuran saat proses pembaruan (koreksi) pada *Kalman Filter*.

3.6. Implementasi Sistem Deteksi dan Pelacakan



Gambar 3.2 Strategi Implementasi Sistem Deteksi Dan Pelacakan

Sistem ini mengadopsi arsitektur *Tracking-by-Detection* di mana model YOLO berfungsi sebagai detektor objek yang menghasilkan *measurement* (pengukuran), dan *Kalman Filter* berfungsi sebagai *tracker* (pelacak) yang melakukan prediksi dan koreksi. Keluaran dari YOLO berupa koordinat pusat bola (x_c, y_c) pada setiap *frame* digunakan sebagai input pengukuran (Z_k) yang bising ke *Kalman Filter*. *Kalman Filter* kemudian menjalankan siklus prediksi dan pembaruan untuk menghasilkan estimasi keadaan bola yang optimal. Keluaran

Kalman Filter adalah posisi dan kecepatan bola yang telah dihaluskan dan diprediksi, siap diteruskan ke modul kontrol gerak robot.

Kalman Filter dikonfigurasi untuk memodelkan keadaan bola dengan 8 status (*state*) dan menerima 4 pengukuran (*measurement*).

- **Vektor Keadaan ($\boldsymbol{X}$):** Mewakili posisi, dimensi, dan kecepatan perubahan objek, didefinisikan sebagai $[x, y, w, h, \dot{x}, \dot{y}, \dot{w}, \dot{h}]^T$. Di sini, x, y adalah koordinat pusat, w, h adalah lebar dan tinggi *bounding box*, dan $\dot{x}, \dot{y}, \dot{w}, \dot{h}$ adalah laju perubahan (kecepatan) dari variabel-variabel tersebut.
- **Vektor Pengukuran ($\boldsymbol{Z}$):** Mewakili data yang diterima dari detektor YOLO pada *frame* tertentu, didefinisikan sebagai: $[x, y, w, h]^T$.

Strategi *Frame Skipping* diterapkan untuk optimasi komputasi dan stabilisasi gerakan. Alur kerjanya adalah sebagai berikut:

1. Pengukuran YOLO (misal *frame* 1): Pada *frame* awal, YOLO dijalankan untuk mendapatkan data pengukuran Z. Data ini digunakan untuk menginisialisasi atau memperbarui status *Kalman Filter*.
2. Prediksi *Kalman Filter* (*Skip Frames*): Untuk beberapa *frame* berikutnya (misalnya *frame* 2 hingga *frame* 5), sistem tidak menjalankan YOLO. Sebaliknya, *Kalman Filter* berada dalam mode prediksi penuh, menghasilkan posisi bola yang diestimasi berdasarkan model gerakan internalnya. Hal ini mengurangi beban komputasi secara signifikan.
3. Pengukuran Ulang YOLO (*frame* 6): Pada *frame* setelah pengukuran terakhir, YOLO diaktifkan kembali untuk memberikan data pengukuran baru (Z_k) yang akan digunakan *Kalman Filter* untuk pembaruan (*update*). Siklus ini kemudian berulang.

Strategi ini juga meningkatkan ketahanan sistem terhadap *occlusion* sesaat. Jika YOLO gagal mendeteksi bola pada *frame* yang seharusnya (misalnya *frame* 1 atau *frame* 6), *Kalman Filter* akan diinstruksikan untuk tetap menjalankan langkah prediksi tanpa *update* dari YOLO. Untuk menjaga integritas pelacakan, batas toleransi kegagalan deteksi diatur maksimal 5 kali kejadian berturut-turut. Jika *Kalman Filter* harus terus memprediksi lebih dari 5 *frame* tanpa input YOLO

(misalnya karena bola hilang atau terhalang total), *track* (jejak) dianggap hilang, dan sistem akan kembali ke mode pencarian penuh.

Implementasi *Kalman Filter* dilakukan menggunakan modul bawaan OpenCV, yang menyediakan fungsi-fungsi terstruktur untuk inisialisasi, prediksi, dan pembaruan, memanfaatkan operasi matriks yang dioptimalkan untuk performa *real-time*. *Output* status (x, y) yang dihaluskan dan diprediksi dari *Kalman Filter* kemudian digunakan oleh modul kontrol robot untuk menghasilkan perintah gerakan yang stabil dan responsif.

3.7. Desain dan Implementasi Kontrol Gerak Robot Berbasis Visi

Perancangan sistem menggunakan pendekatan pembagian fungsi antara dua platform, yaitu:

1. **Unit Pemrosesan Visi (ROS):** Laptop pada robot menjalankan ROS yang berfungsi membaca citra dari kamera, melakukan deteksi objek menggunakan YOLO dan *Kalman Filter*, menentukan lokasi objek pada grid, dan menghasilkan perintah gerak diskrit.
2. **Unit Eksekusi Gerak (Arduino Mega):** Berfungsi menerima perintah gerak dari ROS melalui komunikasi *Serial*, menerjemahkannya menjadi aksi motor, dan menjalankan manuver sesuai perintah.

Alur data utama pada sistem meliputi: kamera → deteksi YOLO → prediksi posisi *Kalman Filter* → identifikasi grid → perintah gerak → pengiriman melalui topik "`master/command`" → eksekusi pada Arduino. Pendekatan distribusi seperti ini memastikan proses pengolahan citra yang berat dilakukan di ROS, sedangkan Arduino hanya menjalankan fungsi aktuasi motor.

Informasi posisi objek dikonversi menjadi instruksi gerak melalui pembagian area pengamatan menjadi grid. Pembagian grid ditentukan berdasarkan hasil kalibrasi kamera sehingga setiap kotak merepresentasikan area tertentu pada bidang pandang. Objek yang terdeteksi (misalnya bola) akan memiliki koordinat grid tertentu, misalnya (X_{grid}, Y_{grid}). Koordinat tersebut dikategorikan ke dalam beberapa zona tindakan, seperti zona kiri, zona kanan, zona tengah, atau zona jauh. Setiap zona dipetakan ke instruksi gerak tertentu, antara lain:

- Zona tengah atas → robot bergerak maju.
- Zona kiri → robot melakukan koreksi arah ke kiri.
- Zona kanan → robot melakukan koreksi ke kanan.
- Zona bawah → robot melakukan putaran pencarian.
- Objek tidak terdeteksi → robot berhenti.

Pendekatan berbasis grid ini membuat sistem mudah diimplementasikan karena instruksi gerak ditentukan oleh lokasi visual secara langsung.

Komunikasi antara ROS dan Arduino dilakukan melalui *Serial* dengan format pesan berbasis *string*, misalnya "`maju`", "`putarKanan`", atau "`majuPelan`". *Node* ROS menerbitkan perintah ini ke topik "`master/command`", kemudian dikirimkan ke Arduino. Arduino berfungsi membaca pesan yang diterima, melakukan *parsing*, dan memanggil fungsi gerak yang sesuai. Setiap perintah memiliki fungsi motor tersendiri, misalnya fungsi untuk bergerak maju, berbelok, atau bergerak dengan kecepatan rendah. Implementasi ini memungkinkan robot bergerak secara responsif berdasarkan keluaran pengolahan citra pada ROS.

3.8. Pengujian dan Evaluasi Kinerja

Tahap pengujian dilakukan untuk memvalidasi efektivitas integrasi algoritma *Kalman Filter* pada sistem deteksi YOLO dalam mengatasi keterbatasan perangkat keras dan kondisi dinamis lapangan. Pengujian dirancang menggunakan pendekatan komparatif, yaitu membandingkan kinerja sistem dalam dua kondisi percobaan. Pertama, sistem hanya menggunakan deteksi YOLO pada setiap *frame* tanpa prediksi yang akan disebut kondisi *baseline*. Kedua, sistem menggunakan strategi *Frame Skipping* dengan YOLO dan prediksi *Kalman Filter* (YOLO-KF) yang akan disebut kondisi *optimized*.

Evaluasi dilakukan pada tiga aspek utama: kinerja komputasi perangkat keras, akurasi pelacakan objek (termasuk penanganan *occlusion*), dan kualitas respons perintah gerak robot.

3.8.1. Skenario Pengujian

Pengujian dilakukan dengan serangkaian skenario terkontrol sebagai berikut:

1. **Pengujian Kinerja Komputasi (*Computational Performance Test*)**

   Skenario ini bertujuan untuk membuktikan efisiensi sistem *Optimized* pada laptop ASUS K401U.

   - **Prosedur:** Robot ditempatkan dalam posisi diam menghadap bola yang statis. Sistem dijalankan selama 60 detik untuk masing-masing kondisi (*baseline* dan *optimized*).
   - **Pengambilan Data:** Sistem akan mencatat waktu proses (*inference time*) dan jumlah frame yang berhasil diproses per detik (*Frames Per Second* / FPS) secara *real-time* ke dalam *log file*.

2. **Pengujian Akurasi Pelacakan dan Occlusion (*Tracking Accuracy & Occlusion Test*)**

   Skenario ini bertujuan menguji kemampuan algoritma mempertahankan posisi bola saat terjadi gangguan visual.

   - **Prosedur Uji Lintasan Lurus:** Bola digelindingkan pada lintasan lurus yang telah ditandai di lantai (sebagai *Ground Truth*) sepanjang 2 meter. Posisi koordinat (x, y) yang dihasilkan sistem direkam dan dibandingkan dengan garis lintasan sebenarnya.
   - **Prosedur Uji *Occlusion*:** Sebuah penghalang (kardus) ditempatkan di tengah lintasan sehingga bola tertutup total dari pandangan kamera robot selama beberapa saat.
   - **Observasi:** Pada kondisi *optimized*, sistem diharapkan tetap menghasilkan koordinat prediksi posisi bola (jalur imajiner) di belakang penghalang, sedangkan pada kondisi *baseline*, data posisi akan hilang.

3. **Pengujian Respons Gerak Robot (*Robot Response Test*)**

   Skenario ini bertujuan melihat kestabilan perintah gerak yang dikirimkan oleh unit pemrosesan visi (ROS) ke unit eksekusi gerak (Arduino).

   - **Prosedur Uji *Jitter* (Guncangan):** Bola digerakkan ke kiri dan ke kanan dengan cepat (dikocok) di depan robot dalam area pandang yang sempit.

- **Prosedur Uji Intersepsi:** Bola digelindingkan menyilang di depan robot, dan robot harus mengirimkan perintah untuk merespons pergerakan tersebut.
- **Pengambilan Data:** *Data logger* akan merekam perubahan perintah (*command switching*) yang dikirim ke topik *Serial* "`master/command`" (seperti `"maju"`, `"putarKanan"`, `"putarKiri"`, `"berhenti"`). Fokus pengamatan adalah frekuensi perubahan perintah yang tidak perlu akibat *noise* deteksi.

3.8.2. Metrik Evaluasi Kinerja

Evaluasi kinerja difokuskan pada dua kelompok metrik, yaitu:

1. **Metrik Kinerja Komputasi**
   - **Rata-rata FPS (*Average Frames Per Second*):** Nilai FPS yang lebih tinggi menunjukkan sistem lebih responsif dan ringan dijalankan pada perangkat keras yang tersedia.
   - **Stabilitas *Frame Rate*:** Mengukur seberapa sering terjadi penurunan FPS (*frame drop*) drastis yang dapat menyebabkan *lag* pada robot.
2. **Metrik Akurasi Pelacakan**
   - ***Root Mean Square Error* (RMSE):** Mengukur selisih rata-rata antara posisi bola hasil deteksi/prediksi sistem dengan posisi sebenarnya (*Ground Truth*) pada lintasan lurus. Nilai RMSE yang lebih kecil menunjukkan akurasi yang lebih baik.
   - ***Success Rate on Occlusion*:** Persentase keberhasilan sistem dalam memberikan *output* koordinat bola secara terus-menerus saat bola melewati area tertutup (*occlusion*).
3. **Metrik Kualitas Respons (*Log* Perintah)**
   - **Frekuensi *Command Switching*:** Menghitung jumlah pergantian perintah gerak dalam satuan waktu tertentu saat merespons target yang sama. Contohnya, jika dalam 1 detik perintah berubah-ubah dengan cepat (`"kiri"` → `"kanan"` → `"kiri"`), ini mengindikasikan ketidakstabilan (jitter). Sistem *optimized* diharapkan menghasilkan

aliran perintah yang lebih konsisten (misalnya: `"kiri"` → `"kiri"` → `"kiri"` → `"maju"`) berkat penghalusan data oleh *Kalman Filter*.

- **Latensi Re-akuisisi:** Waktu yang dibutuhkan sistem untuk mengirimkan perintah gerak yang tepat kembali setelah bola keluar dari kondisi *occlusion*.

Dengan membandingkan hasil pengujian *baseline* (YOLO murni) dengan hasil *optimized* (YOLO + *Kalman Filter*), penelitian ini akan secara kuantitatif membuktikan sejauh mana *Kalman Filter* mampu mengoptimalkan gerakan robot Sepak Bola Beroda.

DAFTAR PUSTAKA

Baskoro, G. Y., Afrisal, H., & Sofwan, A. (2022). Perancangan Sistem Deteksi Objek Berbasis Convolutional Neural Network Menggunakan Yolov4 Dan Opencv. *Transient: Jurnal Ilmiah Teknik Elektro*, *11*(4), 128–020637. https://ejournal3.undip.ac.id/index.php/transient

Egi, Y. (2022). *Basketball self training shooting posture recognition and trajectory estimation using computer vision and Kalman filter*. *73*, 19–27.

Farhan, T. M., & Candra, F. (2025). CNN-Based Ball and Goal Detection for KRSBI Robot with Omnidirectional Camera. *International Journal of Electrical, Energy and Power System Engineering*, *8*(2), 86–98. https://doi.org/10.31258/ijeepse.8.2.1-13

Firdaus, A. Z., & Lelono, D. (2025). *Sistem Klasifikasi Sampah Otomatis Berbasis Deteksi Objek Real-Time Pada Single Board Computer Dengan Algoritma YOLO*. *15*(1), 49–60. https://doi.org/10.22146/ijeis.104520

Hendrik, B., & Awal, H. (2023). Pengenalan Teknologi Robot Pada Anak Sekolah Dasar. *Jurmas Bangsa*, *1*(1), 46–52. https://doi.org/10.62357/jpb.v1i1.140

Hodson, T. O. (2022). *Root-Mean-Square Error (RMSE) Or Mean Absolute Error (MAE): When To Use Them Or Not*. *2*, 5481–5487.

Jung, H., Kang, S., Kim, T., Kim, H., Klemove, H. L., & Korea, R. (2024). ConfTrack : Kalman Filter-based Multi-Person Tracking by Utilizing Confidence Score of Detection Box. *CVF Open Access*, 6583–6592.

Kusumoputro, B., Purnomo, M. H., Rochardjo, H. S. B., Prabowo, G., Purwanto, D., Mozef, E., Indrawanto, Mutijarsa, K., & Muis, A. (2024). *Pedoman Kontes Robot Indonesia (Kri) Pendidikan Tinggi Tahun 2024*.

Lee, Y. G. (2024). Confidence-Guided Frame Skipping to Enhance Object Tracking Speed. *Sensors*, *24*(24). https://doi.org/10.3390/s24248120

Louda, S., Karkar, N., & Seghir, F. (2023). *Autonomous Navigation of a Differential Mobile Robot using Depth Camera Sensor-based ROS*.

Miharja, G. P., Nugraha, D. A., & Aziz, A. (2025). Analisis Perbandingan Kinerja YOLO dan Camshift Dalam Pelacakan Objek Berbasis Video. *Jurnal Riset*

Mahasiswa Bidang Teknologi Informasi Volume, *5*(Analisis Perbandingan Kinerja Yolo dan Camshift), 100–110.

Mohammed, H. R., & Hussain, Z. M. (2021). *Detection and Recognition of Moving Video Objects: Kalman Filtering with Deep Learning*. *12*, 154–157.

Nanda, B. M., Siregar, S., & Sani, M. I. (2023). *Implementasi Object Detection Pada Robot Sepak Bola Beroda Berbasis Kamera Omnidirectional Menggunakan OpenCV*. *9*(4), 2064–2068.

NumPy. (2025). *NumPy*. https://numpy.org/

Nuralim, J., Fath, N., Musafa, A., Sujono, & Broto, D. S. (2022). Perancangan Sistem Pendeteksian Obyek Bola Dengan Metode Framework YOLOv4. *Jurnal Maestro*, *5*(2), 289–294. https://jom.ft.budiluhur.ac.id/index.php/maestro/article/view/531

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., & Ng, A. (2009). *ROS : An Open-source Robot Operating System*. *Figure 1*.

Ratna, S. (2020). Pengolahan Citra Digital Dan Histogram Dengan Phyton Dan Text Editor Phycharm. *Technologia: Jurnal Ilmiah*, *11*(3), 181. https://doi.org/10.31602/tji.v11i3.3294

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-time Object Detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 779–788. https://doi.org/10.1109/CVPR.2016.91

Ritonga, A. A., & Hasibuan, E. R. (2025). *Eksplorasi Masa Depan Robotika : Integrasi Kecerdasan Buatan dalam Sistem Automasi Adaptif Berbasis Lingkungan*. *1*, 387–393.

Rochardjo, H. S. B. (2024). *Sosialisasi KRSBI Beroda 2024*.

Romario, M. H., & Kadarina, T. M. (2020). *Sistem Hitung Dan Klasifikasi Objek Dengan Metode Convolutional Neural Network*. *11*(2), 108–114.

Ryu, S. E., & Chung, K. Y. (2021). Detection Model Of Occluded Object Based On Yolo Using Hard-example Mining And Augmentation Policy Optimization. *Applied Sciences (Switzerland)*, *11*(15).

https://doi.org/10.3390/app11157093

Saputra, C. (2023). Implementasi Algoritma SIFT (Scale-Invariant Feature Transform) Dan Algoritma Kalman Filter Dalam Mendeteksi Objek Bola. *Jurnal PROCESSOR*, *18*(1), 73–82. https://doi.org/10.33998/processor.2023.18.1.791

Saputra, F. B., Kallista, M., & Setianingsih, C. (2023). Deteksi social distancing dan penggunaan masker di restoran menggunakan algoritma Residual Network (ResNet). *EProceedings of Engineering*, *10*(1), 284–295.

Shofi, M., Basuki, B. M., & Habibi, A. (2023). Rancang Bangun Penendang Bola Pada Robot Soccer Unisma Menggunakan Solenoid. *Science Electro*, *16*(4), 1–7.

Sholehurrohman, R., Habibi, M. R., Ilman, I. S., Taufiq, R., & Muhaqiqin, M. (2023). Analisis Metode Kalman Filter, Particle Filter dan Correlation Filter Untuk Pelacakan Objek. *Komputika : Jurnal Sistem Komputer*, *12*(2), 21–28. https://doi.org/10.34010/komputika.v12i2.9567

Sugiyono. (2013). *Metode Penelitian Kuantitatif, Kualitatif dan R&D*.

Surya, M., Nehemia Toscany, A., Saputra, C., Pratama, Y., & Bustami, M. I. (2025). Penggunaan Yolo Untuk Deteksi Robot Dan Gawang Pada Robot Sepak Bola Beroda. *The Indonesian Journal of Computer Science*, *14*(1), 1055–1070. https://doi.org/10.33022/ijcs.v14i1.4575

Takura, K. E. I., Arita, Y. U. M. A. N., & Oaki, S. H. N. (2021). *Automatic pear and apple detection by videos using deep learning and a Kalman filter*. *4*(5), 1688–1695.

Taylor, L. E., Mirdanies, M., & Saputra, R. P. (2016). *OPTIMIZED OBJECT TRACKING TECHNIQUE USING KALMAN*. *07*, 57–66. https://doi.org/10.14203/j.mev.2016.v7.57-66

Ultralytics. (2025). *Kalman Filter (KF)*. https://www.ultralytics.com/glossary/kalman-filter-kf

Wakhidah, N., Pungkasanti, P. T., & Pinem, A. P. R. (2023). *Deteksi Objek menggunakan Deep Learning*. *9*(3), 465–470.

Widodo, Y. B., Sibuea, S., & Narji, M. (2024). Kecerdasan Buatan dalam

Pendidikan: Meningkatkan Pembelajaran Personalisasi. *Jurnal Teknologi Informatika Dan Komputer*, *10*(2), 602–615. https://doi.org/10.37012/jtik.v10i2.2324

Wijaya, G. F., & Yuniarto, D. (2024). Tinjauan Penerapan Machine Learning pada Sistem Rekomendasi Menggunakan Model Klasifikasi. *Populer: Jurnal Penelitian Mahasiswa*, *3*(4), 144–153. https://doi.org/10.58192/populer.v3i4.2798

Yaseen, M. (2024). *What Is Yolov8: An In-Depth Exploration Of The Internal Features Of The Next-Generation Object Detector*. *8*, 1–10.

Yuztiawan, F. R., & Utaminingrum, F. (2017). Implementasi Metode Kalman Filter Dan Model Yolov8N Untuk Fitur Human-Following Pada Kursi Roda Pintar. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *1*(1), 2548–2964.