

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

Fikri Rivandi | 2207112583



Universitas Riau  
Program Studi S1 Teknik Informatika

"Menjadi Program Studi Teknik Informatika Terkemuka Berbasis Riset dan Teknologi yang Bermartabat di Kawasan Asia Tenggara pada Tahun 2035"

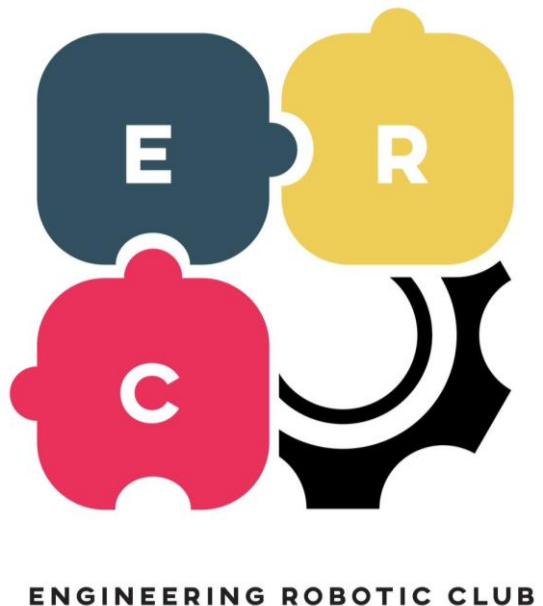


# BAB I

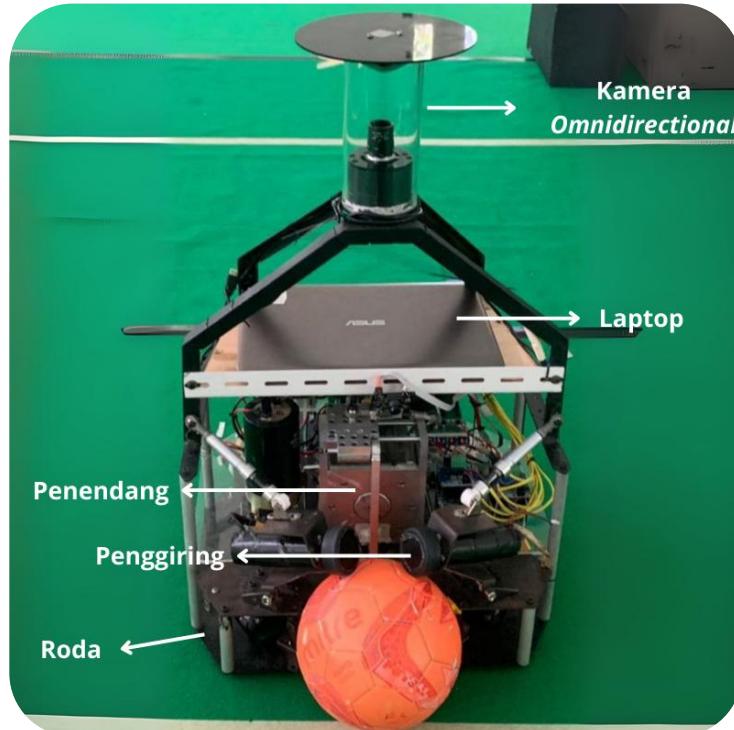
## Pendahuluan

# Latar Belakang

- **Objek Penelitian:** Tim ERC (*Engineering Robotic Club*) Universitas Riau (UNRI) yang berfokus pada KRSBI Beroda.



Gambar 1 Logo ERC UNRI



Gambar 2 Robot KRSBI Beroda ERC UNRI

# Latar Belakang

- **Tantangan Kompetisi:** Robot sepak bola beroda menuntut respon cepat terhadap lingkungan yang dinamis dan tidak terduga.



Gambar 3 Ilustrasi Pertandingan  
KRSBI Beroda

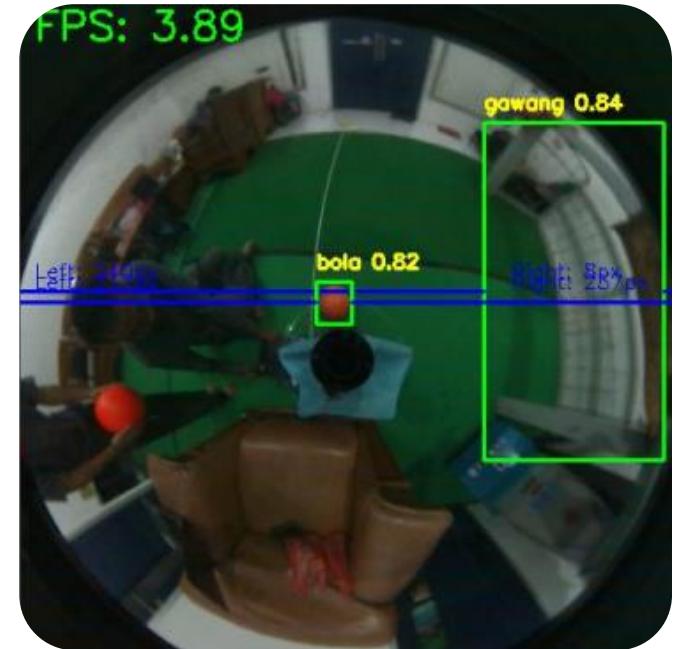
# Latar Belakang

## Perubahan Metode:

- **Metode Lama (2019-2024)** menggunakan HSV (*Color Filtering*). Ringan, tapi akurasi sangat bergantung pada kondisi cahaya (tidak stabil).
- Sedangkan **Metode Baru (2025)** beralih ke *Deep Learning* (YOLO) untuk akurasi deteksi yang lebih tinggi dan tahan perubahan cahaya.



Gambar 4 Implementasi HSV pada Robot KRSBI ERC UNRI



Gambar 5 Implementasi YOLO pada Robot KRSBI ERC UNRI

# Latar Belakang

## Masalah Pertama

- **Tuntutan Algoritma:** YOLO membutuhkan komputasi paralel tinggi dan dukungan GPU mumpuni agar berjalan stabil.
- **Keterbatasan Hardware:** Robot Tim ERC UNRI saat ini menggunakan laptop **ASUS K401U** dengan spesifikasi terbatas (Intel Core i5-7200U & GPU Nvidia 940MX).



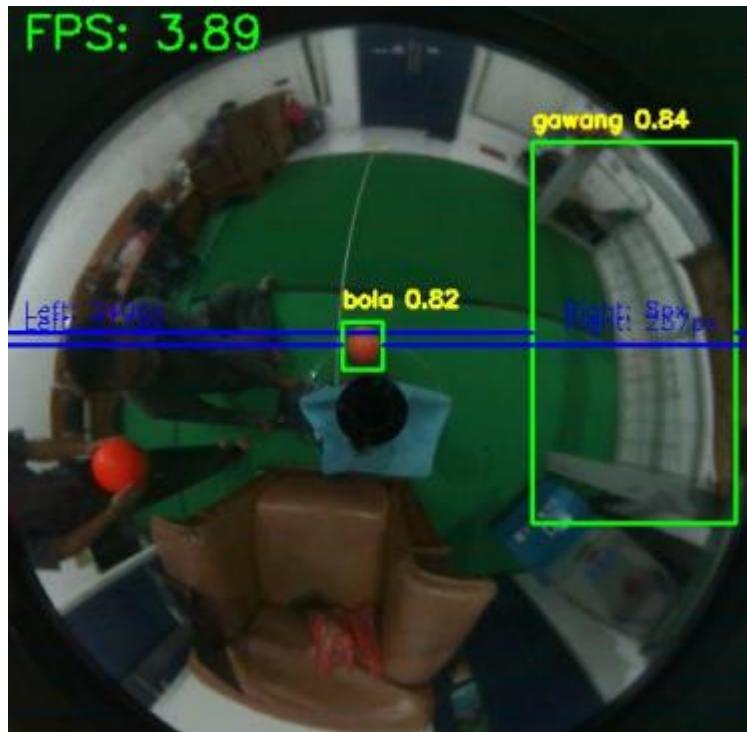
Gambar 6 Laptop ASUS K401U

# Latar Belakang

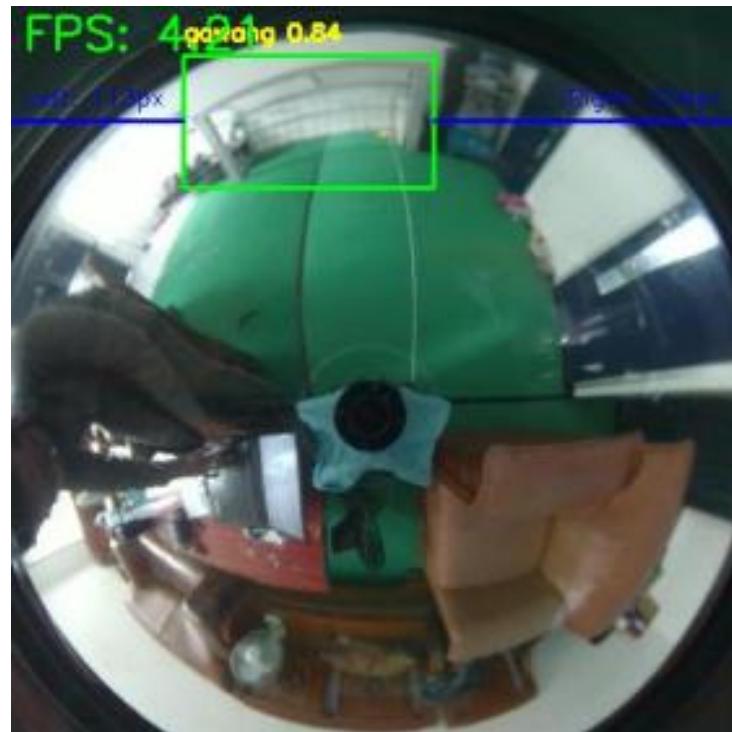
## Dampak Masalah Pertama

- Terjadi penurunan performa (*lag*) dan ketidakstabilan *frame rate*.
- Robot terlambat merespons posisi bola (ada jeda waktu informasi), menyebabkan gerakan tidak stabil.
- YOLO efektif secara teori, tapi tidak efisien untuk *resource* perangkat keras yang tersedia.

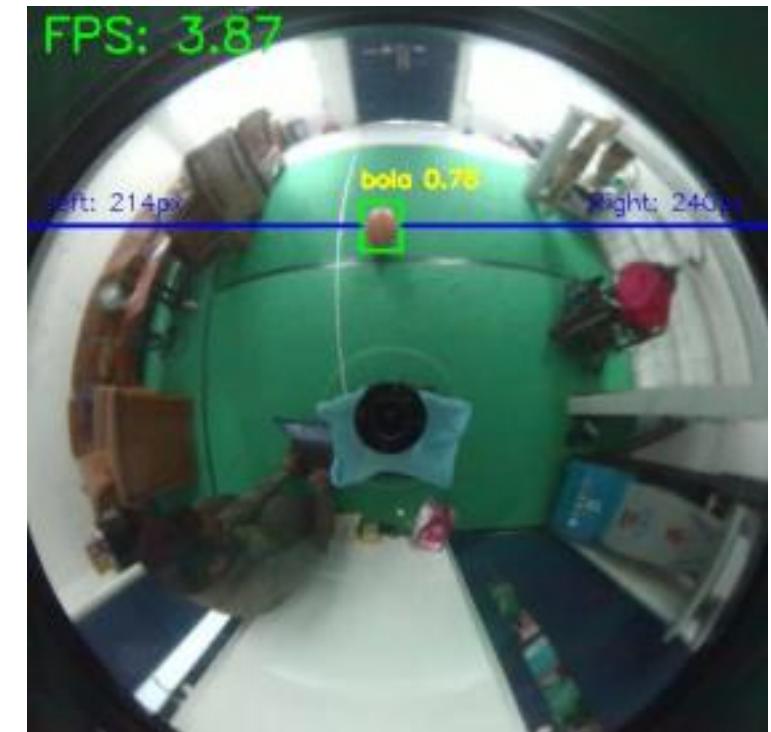
# Latar Belakang



Gambar 7 Contoh Hasil Deteksi dengan YOLO (1)



Gambar 8 Contoh Hasil Deteksi dengan YOLO (2)



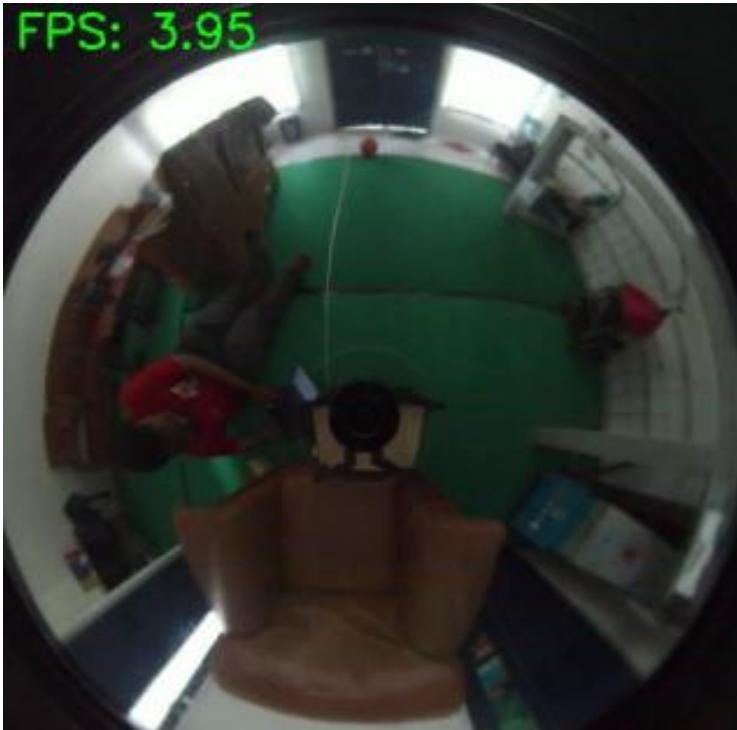
Gambar 9 Contoh Hasil Deteksi dengan YOLO (3)

# Latar Belakang

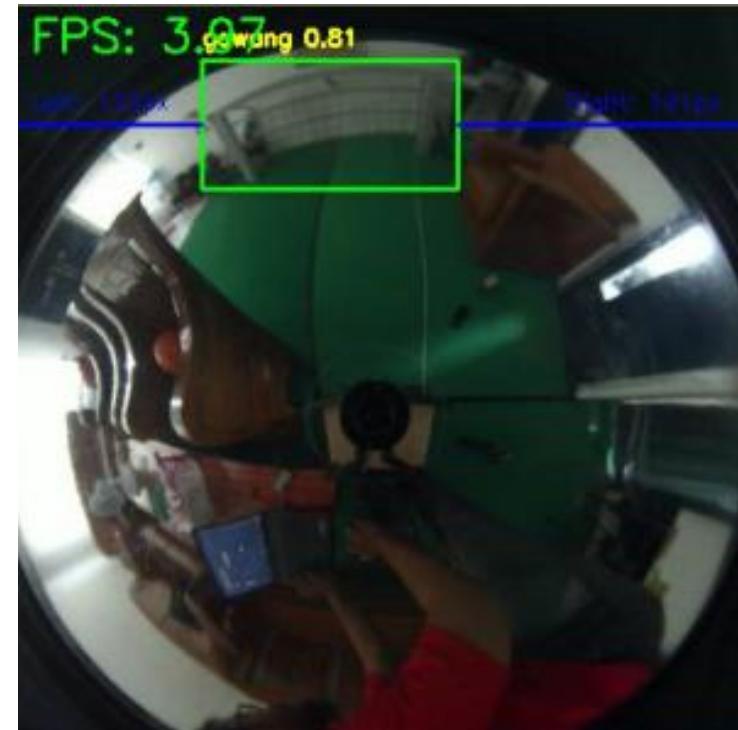
## Masalah Kedua

- **Fenomena Oklusi:** Bola sering tertutup sebagian oleh penghalang (robot lain atau tiang)
- *False Negative:* Saat tertutup, detektor gagal mengenali bola (*miss detection*), dianggap sebagai *hard-positive examples*
- **Kelemahan Detektor Tunggal:** YOLO hanya mendekripsi posisi di *frame* saat ini tanpa mengetahui dinamika gerakan bola. Jika bola bergerak cepat atau tertutup, robot kehilangan jejak

# Latar Belakang



**Gambar 10** Contoh Kejadian *False Negative* Pada YOLO



**Gambar 11** Contoh Kejadian Oklusi  
(*occlusion*)

# Latar Belakang

## Solusi Yang Ditawarkan

- **Integrasi Algoritma:** Menggabungkan deteksi YOLO dengan estimasi *Kalman Filter*
- **Peran *Kalman Filter*:**
  - **Prediksi:** Mengestimasi posisi bola di *frame* berikutnya berdasarkan data sebelumnya, bukan hanya mengandalkan penglihatan saat ini (Ultralytics, 2025)
  - **Efisiensi:** Algoritma ini ringan (hanya menambah beban komputasi ~7,92%), sangat cocok untuk laptop spesifikasi terbatas (Yuztiawan & Utaminingrum, 2017)
  - **Stabilitas:** Membantu robot tetap melacak bola meski terjadi *occlusion* atau *frame drop* (Ultralytics, 2025)
- **Goal Akhir:** Menciptakan gerakan robot yang **Mulus, Stabil, dan Efisien** meskipun hardware terbatas

# Rumusan Masalah

## Masalah Kestabilan pada Hardware Terbatas

- **Kondisi:** Algoritma YOLO berat untuk laptop ASUS K401U yang digunakan.
- **Pertanyaan:** Bagaimana cara meningkatkan kestabilan deteksi bola ketika YOLO mengalami penurunan performa (*frame rate* tidak stabil & *latency*) akibat kapasitas *hardware* yang rendah?

# Rumusan Masalah

## Masalah Kontinuitas Data (*Occlusion*)

- **Kondisi:** Di lapangan dinamis, bola sering tertutup objek lain (*occlusion*) atau terjadi *false-negative* (gagal deteksi).
- **Pertanyaan:** Bagaimana cara mengurangi dampak *occlusion* tersebut agar robot tetap bisa memprediksi posisi bola secara konsisten tanpa kehilangan jejak?

# Tujuan Penelitian

## Mengatasi Keterbatasan *Hardware*

- Mengembangkan sistem deteksi yang lebih **stabil dan responsif** pada robot KRSBI Beroda
- Mengoptimalkan proses berat pada YOLO menggunakan *Kalman Filter* agar tetap berjalan lancar meski spesifikasi perangkat keras terbatas

# Tujuan Penelitian

## Meningkatkan Kontinuitas Pelacakan (*Tracking*)

- Memanfaatkan kemampuan **prediksi** *Kalman Filter* untuk mengurangi dampak saat bola tertutup (*occlusion*) atau gagal terdeteksi (*false-negative*).
- Menghasilkan pergerakan robot yang lebih **efisien, tepat sasaran, dan adaptif** terhadap dinamika permainan.

# Batasan Masalah

- **Fokus Deteksi:** Penelitian hanya membahas deteksi dan pelacakan **bola** menggunakan kamera *omnidirectional* pada robot KRSBI Beroda.
- **Algoritma:**
  - Menggunakan algoritma **YOLO** yang sudah ada (tidak mengembangkan arsitektur baru).
  - Prediksi posisi hanya menggunakan *Kalman Filter*, tanpa membandingkan dengan algoritma prediksi lain.
- **Ruang Lingkup Sistem:** Fokus pada optimasi **gerakan robot terhadap bola** (individu), tidak mencakup strategi tim, komunikasi antar robot, atau kontrol *hardware* secara mendetail.

# Manfaat Penelitian

- **Bagi Peneliti:** Memperoleh pengalaman implementasi integrasi *Deep Learning* (YOLO) dengan estimator (*Kalman Filter*) pada sistem robotika nyata.
- **Bagi Tim Robotika (ERC UNRI):** Meningkatkan performa robot dalam pertandingan melalui sistem deteksi yang lebih akurat dan gerakan yang lebih stabil serta efisien.

# BAB II

## Tinjauan Pustaka

# Penelitian Terdahulu

## Evolusi Metode:

- **Tradisional (HSV/OpenCV)**: Ringan tapi gagal saat cahaya berubah (Nanda et al., 2023).
- **Deep Learning (CNN/YOLO)**: Akurasi tinggi, tapi berat secara komputasi (Farhan & Candra, 2025).
- **Tracking (*Kalman Filter*)**: Terbukti sukses untuk prediksi objek bergerak dan mengurangi *noise* (Sholehurrohman et al., 2023).

## Kebaruan (*Novelty*):

- Banyak penelitian pakai YOLO, atau pakai *Kalman Filter* secara terpisah.
- Penelitian ini **menggabungkan** YOLO (detektor modern) dengan *Kalman Filter* (stabilisator) khusus untuk **optimasi gerakan robot** di perangkat keras terbatas, yang belum banyak dibahas secara spesifik.

# Platform Pengembangan

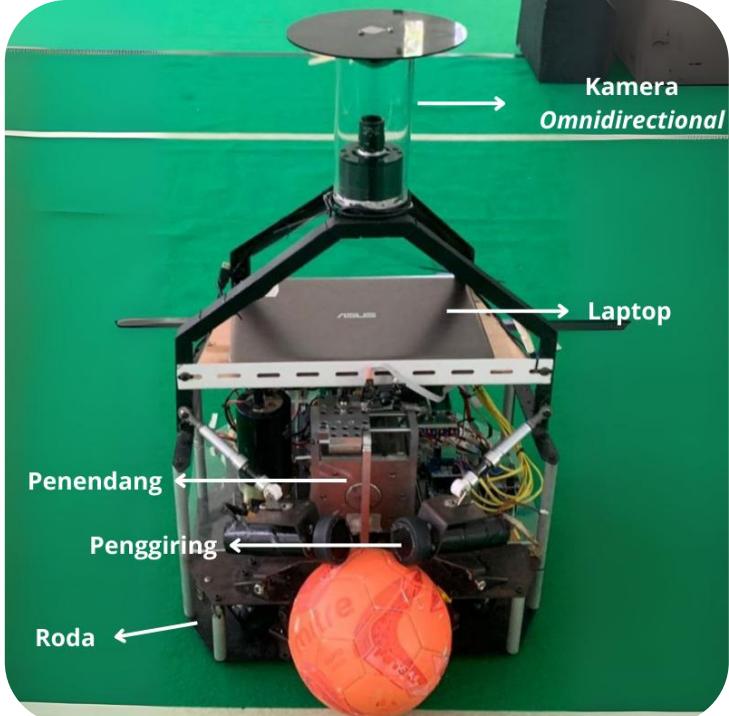
## Robot KRSBI Beroda:

- Menggunakan kamera *omnidirectional* (pandangan 360 derajat) untuk melihat bola di segala arah.
- Bergerak menggunakan *omni-wheels* (pergerakan segala arah).

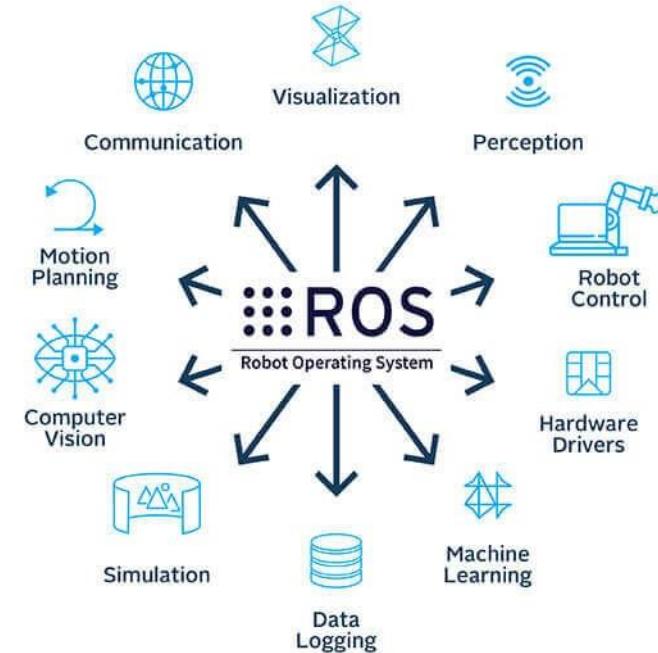
## ROS (*Robot Operating System*):

- Bukan OS biasa (seperti Windows), tapi *middleware* komunikasi antar program.
- Berperan sebagai jembatan komunikasi data antara kamera, laptop (otak pemrosesan), dan Arduino (penggerak motor).

# Platform Pengembangan



Gambar 12 Robot Sepak Bola Beroda



Gambar 13 *Robot Operating System* (ROS)

# YOLOv8 – Algoritma Deteksi

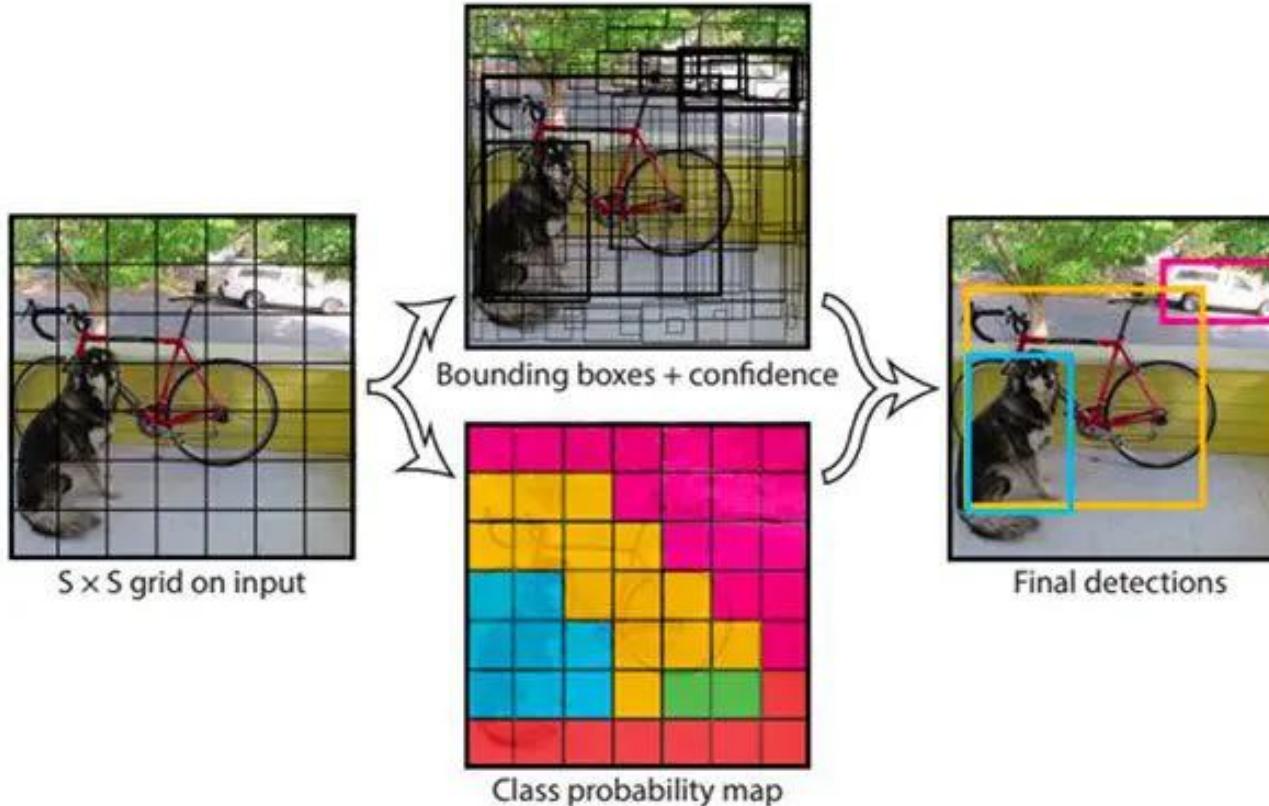
## Kenapa YOLOv8?

- Merupakan algoritma *state-of-the-art* yang menyeimbangkan **kecepatan (real-time)** dan **akurasi**.
- Lebih tahan terhadap perubahan kondisi cahaya dibanding metode warna (HSV).

## Cara Kerja:

- Sekali lihat (*You Only Look Once*), langsung memprediksi kotak pembatas (*bounding box*) dan probabilitas kelas objek (Bola).
- **Kelemahan:** Membutuhkan komputasi GPU yang tinggi, sehingga berat jika dijalankan di laptop lama tanpa optimasi.

# YOLOv8 – Algoritma Deteksi



**Gambar 14** Ilustrasi Algoritma YOLO

# *Kalman Filter – Algoritma Optimasi*

## **Definisi:**

- Algoritma estimator rekursif untuk memperkirakan posisi objek berdasarkan data masa lalu yang mengandung *noise*.

## **Mekanisme Kerja:**

- **Prediksi (Predict):** Menebak posisi bola di *frame* berikutnya berdasarkan kecepatan sebelumnya.
- **Koreksi (Update):** Memperbaiki prediksi saat data baru dari YOLO tersedia.

## **Keunggulan:**

- Ringan secara komputasi dan mampu "mengisi kekosongan" data saat deteksi YOLO gagal atau lambat.

# Hipotesis Penelitian

## Rumusan Hipotesis:

- **$H_0$  (Hipotesis Nol):** Penambahan *Kalman Filter* **tidak memberikan dampak signifikan** pada stabilitas gerakan robot maupun akurasi pelacakan bola.
- **$H_1$  (Hipotesis Alternatif):** Integrasi *Kalman Filter* pada YOLO menghasilkan gerakan robot yang **lebih stabil (minim jitter)** dan pelacakan yang **lebih akurat (RMSE rendah)**, terutama saat bola tertutup halangan (*occlusion*).

## Indikator Keberhasilan ( $H_1$ Diterima Jika):

- **Akurasi:** Nilai *Error* (RMSE) pada sistem *Optimized* lebih kecil daripada sistem *Baseline*.
- **Stabilitas:** Frekuensi perubahan perintah gerak (*Command Switching*) berkurang (robot tidak "bingung" atau *jitter*).
- **Efisiensi:** FPS tetap stabil di atas batas minimum *real-time* ( $>10$  FPS) meskipun ada tambahan algoritma.

# BAB III

## Metodologi Penelitian

# Metode Penelitian

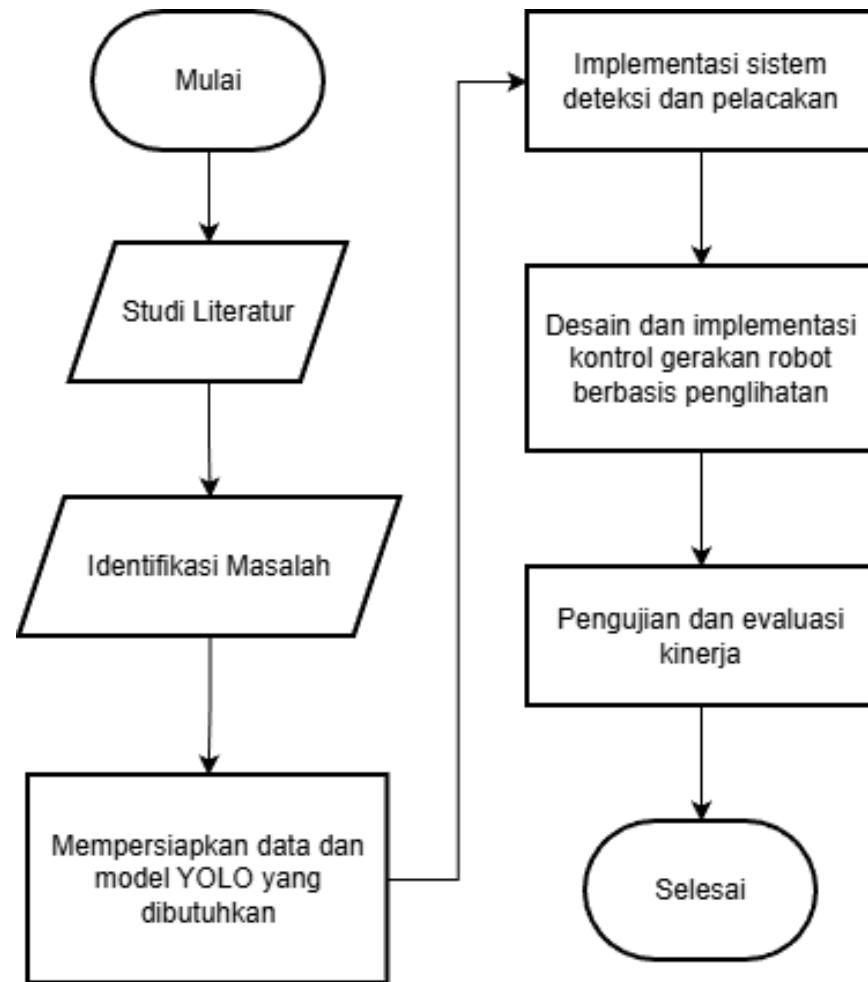
## Jenis Penelitian:

- Kuantitatif dengan strategi Eksperimental.

## Tahapan Utama:

- **Persiapan:** Studi literatur & penyiapan model YOLO
- **Implementasi:** Integrasi sistem deteksi YOLO dengan pelacakan *Kalman Filter*.
- **Pengujian:** Komparasi antara sistem *Baseline* (YOLO murni) vs *Optimized* (YOLO + KF).

# Kerangka Pikiran



Gambar 15 Kerangka Pikiran

# Arsitektur Sistem

## Unit Pemrosesan Visi (Laptop/ROS):

- Menjalankan ROS (*Robot Operating System*).
- Membaca kamera, menjalankan YOLO & *Kalman Filter*, serta menghitung logika gerak.

## Unit Eksekusi Gerak (Arduino Mega):

- Menerima perintah sederhana (*string*) dari ROS via komunikasi *Serial* (contoh: "maju", "putarKanan").
- Menggerakkan motor/roda robot.

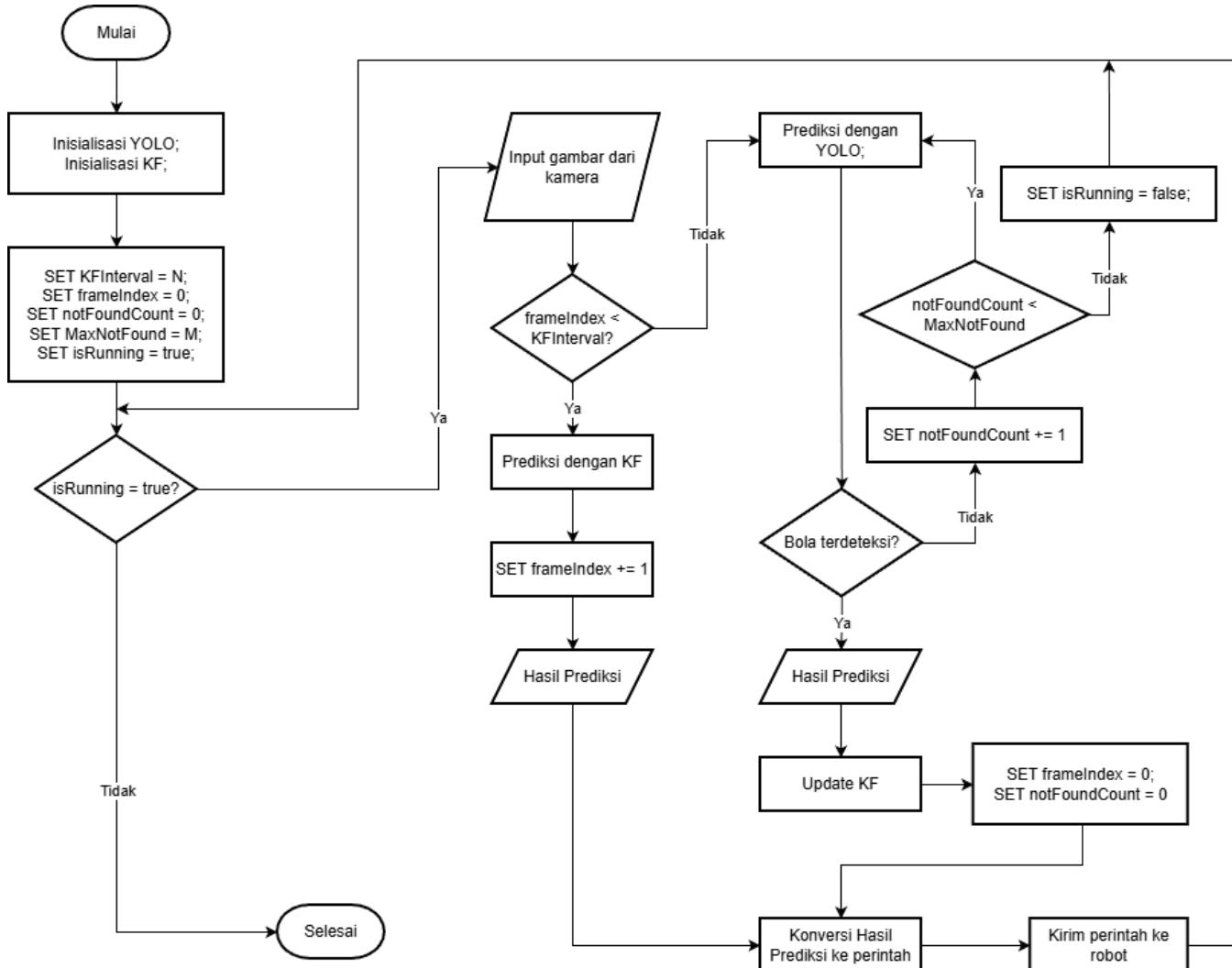
## Alur Data:

- Kamera → Deteksi & Prediksi → Perintah Gerak → Arduino

# Strategi Utama – *Frame Skipping*

- **Konsep:** Tidak menjalankan YOLO di setiap frame untuk menghemat resource.
- **Siklus Kerja:**
  - **Frame 1 (Measurement):** YOLO aktif mendeteksi posisi bola → *Update Kalman Filter*.
  - **Frame 2-5 (Prediction Only):** YOLO **dimatikan**. Posisi bola diprediksi murni oleh *Kalman Filter*. Beban komputasi turun drastis.
  - **Frame 6 (Correction):** YOLO aktif kembali untuk mengoreksi posisi jika prediksi melenceng.
- **Penanganan Loss:** Jika bola tertutup (*occlusion*), *Kalman Filter* tetap memprediksi jalur bola hingga batas toleransi 5-30 *frame* sebelum dianggap hilang (*lost track*).

# Strategi Utama – *Frame Skipping*



Gambar 16 Strategi *Frame Skipping*

# Skenario Pengujian & Evaluasi

- **Metode Komparatif:** Membandingkan kondisi *Baseline* (YOLO *Frame-by-Frame*) vs *Optimized* (YOLO + KF *Frame Skipping*).
- **Skenario Uji:**
  - **Uji Komputasi:** Mengukur peningkatan FPS dan stabilitas di laptop.
  - **Uji Akurasi & Occlusion:** Mengukur seberapa akurat prediksi posisi saat bola bergerak lurus dan saat bola ditutup penghalang.
  - **Uji Respons Robot:** Mengukur *jitter* (guncangan perintah) saat bola bergerak cepat.
- **Metrik:** FPS, RMSE, dan Frekuensi *Command Switching*.

Terima Kasih