# Analisis Kinerja Model Deteksi Kendaraan berbasis Yolo dengan Integrasi Kalman Filter untuk Pemantauan Lalu Lintas *Real-Time*

Muhammad Rizal Islami[1], Diah Priyawati[2]

[1]Universitas Muhammmadiyah Surakarta, Jl. A. Yani, Pabelan, Kartasura, Sukoharjo, Jawa Tengah, Indonesia

[2] Universitas Muhammmadiyah Surakarta, Jl. A. Yani, Pabelan, Kartasura, Sukoharjo, Jawa Tengah, Indonesia

Email korespondensi: l200210044@student.ums.ac.id

**Abstrak**. Pemantauan lalu lintas secara *real-time* adalah komponen kunci dalam sistem transportasi cerdas, namun akurasi deteksi dan pelacakan kendaraan dari rekaman CCTV masih menjadi tantangan, terutama pada kondisi padat atau pencahayaan rendah. Penelitian ini bertujuan mengembangkan model deteksi dan pelacakan yang andal dengan mengintegrasikan algoritma YOLOv8 dan Kalman Filter dalam kerangka kerja HybridTracker. Menggunakan pendekatan kuantitatif, studi ini memanfaatkan dataset gabungan dari video di Kota Surakarta dan tiga dataset publik (COCO, MIO-TCD, UA-DETRAC), yang menghasilkan 10.507 gambar teranotasi. Hasil evaluasi menunjukkan model deteksi YOLOv8 mencapai performa tinggi dengan mAP@0.5 sebesar 0.901. Kelas bus dan mobil menunjukkan F1-score di atas 0.904, meskipun sepeda motor masih menjadi tantangan dengan F1-score 0.848. Dalam tugas pelacakan, HybridTracker yang dioptimalkan dengan Kalman Filter berhasil mencatatkan *Multi-Object Tracking Accuracy* (MOTA) sebesar 66.36% dan IDF1 sebesar 0.88 pada skenario lalu lintas padat, dengan 94.8% objek berhasil dilacak secara konsisten. Model ini terbukti mampu mendeteksi dan melacak kendaraan secara akurat. Pengembangan selanjutnya akan difokuskan pada peningkatan deteksi objek kecil, efisiensi pemrosesan, dan integrasi fitur analitik seperti penghitungan kendaraan dan deteksi pelanggaran untuk mendukung manajemen lalu lintas berbasis data.

***Kata kunci:*** *YOLO; Kalman Filter; deteksi kendaraan; pelacakan objek; pemantauan lalu lintas*

## PENDAHULUAN

Pemantauan lalu lintas secara real-time merupakan kebutuhan penting dalam sistem transportasi cerdas modern. Teknologi ini memungkinkan pengumpulan informasi visual dari kamera CCTV untuk mendukung pengambilan keputusan dalam manajemen lalu lintas, perencanaan infrastruktur, dan peningkatan keselamatan berkendara[1], [2]. Dibandingkan dengan sistem berbasis sensor seperti radar atau LiDAR yang mahal dan kompleks, pendekatan berbasis visi komputer menggunakan kamera konvensional jauh lebih fleksibel dan ekonomis[3], [4] . Pendekatan ini terbukti efektif dalam berbagai aplikasi, termasuk perhitungan kendaraan [5]dan mengenali cuaca dengan menggunakan CCTV[4]. Namun, sistem pemantauan berbasis visi tetap menghadapi tantangan, terutama dalam kondisi lalu lintas padat, pencahayaan rendah, sudut pandang kamera yang terbatas, serta keberadaan oklusi antar kendaraan [6], [7]. Oleh karena itu, dibutuhkan solusi algoritmik yang tangguh dan adaptif. Salah satu algoritma deteksi objek yang paling populer adalah *You Only Look Once* (YOLO), yang mampu mendeteksi objek secara cepat dan efisien dari citra tanpa memerlukan tahap pemrosesan wilayah awal [8], [9].

YOLO telah terbukti unggul dalam aplikasi real-time, seperti deteksi pelanggaran lalu lintas [10] dan klasifikasi kendaraan dari video CCTV [11]. Namun, kemampuan YOLO dalam melakukan pelacakan antar-frame masih terbatas. Ketika objek mengalami oklusi atau berpindah cepat, identitas pelacak bisa hilang, menurunkan stabilitas sistem pelacakan [12]. Untuk mengatasi hal ini, dibutuhkan integrasi dengan algoritma pelacakan seperti *Kalman Filter*, yang mampu memprediksi posisi objek di frame berikutnya berdasarkan posisi sebelumnya[13], [14]. Salah satu pendekatan terkenal dalam hal ini adalah *DeepSORT*, yang menggabungkan deteksi YOLO dengan pelacakan berbasis Kalman Filter dan algoritma pencocokan Hungarian [15]. Kalman Filter terbukti mampu mempertahankan identitas objek dengan baik, terutama saat terjadi oklusi atau hilangnya deteksi sementara. Efektivitasnya bahkan diperkuat dengan pendekatan pembobotan kemunculan objek[16]. Kombinasi ini juga dapat dimanfaatkan untuk estimasi kecepatan kendaraan dan deteksi atribut lain seperti helm atau plat nomor[10], [17].

Selain digunakan dalam pelacakan kendaraan, pengembangan YOLO terbaru seperti YOLOv8 juga telah diterapkan untuk pendeteksian objek kecil, seperti pada citra pengawasan malam dan satelit[7], [16]. Dengan kemampuannya yang semakin baik dalam mendeteksi objek dengan ukuran dan bentuk yang kompleks, YOLOv8 menjadi kandidat kuat untuk digunakan dalam sistem pemantauan lalu lintas di Indonesia. Terlebih, penelitian lokal juga menunjukkan bahwa metode YOLO mampu mendeteksi kendaraan secara akurat di simpang jalan perkotaan Indonesia[2]. Berdasarkan studi

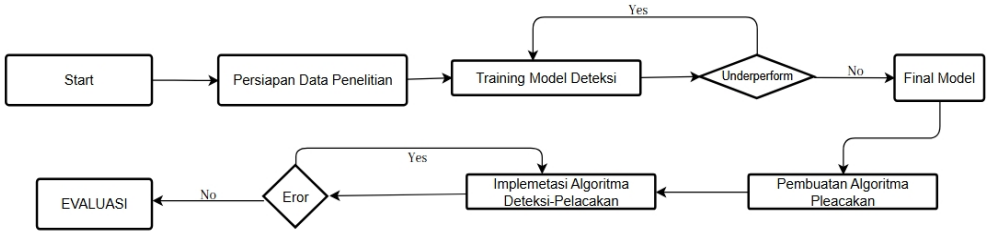
literatur tersebut, terlihat bahwa integrasi YOLO dan Kalman Filter merupakan solusi yang menjanjikan dalam membangun sistem deteksi dan pelacakan kendaraan yang andal.

Oleh karena itu, penelitian ini bertujuan untuk menganalisis kinerja sistem deteksi dan pelacakan kendaraan yang menggabungkan algoritma YOLOv8 dengan Kalman Filter, baik dalam skenario lalu lintas padat, pencahayaan rendah, maupun sudut pandang kamera yang dinamis. Evaluasi dilakukan menggunakan metrik objektif seperti precision, recall, F1-score, dan mean average precision (mAP) untuk deteksi, serta MOTA, IDF1, dan HOTA untuk pelacakan. Diharapkan, penelitian ini dapat memberikan kontribusi dalam pengembangan sistem transportasi cerdas yang adaptif, efisien, dan ekonomis, khususnya di lingkungan perkotaan Indonesia.

## METODE

Bagian Penelitian ini dirancang secara komprehensif untuk mengembangkan dan mengevaluasi Algoritma deteksi serta pelacakan kendaraan berbasis *computer vision*. Pendekatan eksperimen kuantitatif digunakan untuk memastikan bahwa setiap tahapan dilakukan secara terukur dan sistematis. Sistem yang dibangun memanfaatkan integrasi algoritma *You Only Look Once (YOLO)* dengan *Kalman Filter*. Melalui integrasi ini, penelitian diharapkan dapat menghasilkan solusi inovatif yang akurat, efisien, dan adaptif dalam mendukung pemantauan lalu lintas secara *real-time*.

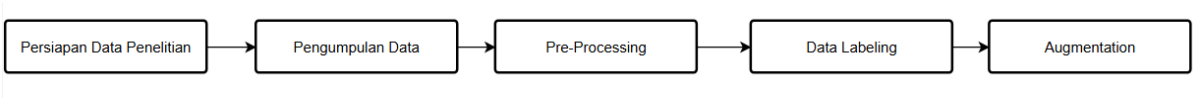
Ruang lingkup penelitian mencakup beberapa tahapan utama, mulai dari kajian pustaka untuk menelaah perkembangan terbaru dalam teknologi deteksi dan pelacakan kendaraan, penyusunan dan persiapan dataset, implementasi Algoritma, pengujian performa dalam berbagai skenario, hingga analisis data dan penyusunan laporan akhir. Evaluasi Algoritma dilakukan dengan membandingkan performa model YOLO yang telah diintergrasikan dengan kalman filter dengan model YOLO yang dintergrasikan dengan metode tracking lain. Guna memudahkan pemahaman terhadap alur kerja penelitian, tahapan-tahapan tersebut divisualisasikan dalam bentuk *flowchart* pada gambar 1.




**Gambar 1**. Flowchar Alur penelitian

Seluruh proses penelitian dilaksanakan dengan menjunjung tinggi prinsip-prinsip etika akademik. Data yang digunakan tidak mengungkap identitas individu, dan seluruh informasi yang diperoleh dipastikan hanya dimanfaatkan untuk kepentingan ilmiah. Dengan pendekatan ini, diharapkan hasil penelitian dapat memberikan kontribusi yang bermakna bagi pengembangan teknologi pemantauan lalu lintas cerdas berbasis kecerdasan buatan di masa depan.

**Persiapan Data Penelitian**




**Gambar 2**. Alur Perisapan Data Penelitian

Tahap pertama dalam alur penelitian ini adalah persiapan data. Proses persiapan data yang dilakukan, seperti yang diilustrasikan pada Gambar 2, mencakup:

**a. Pengumpulan Data**

Data penelitian diperoleh dari video rekaman cctv lalu lintas publik yang terpasang di daerah Surakarta. Lokasi pengambilan data dipilih secara strategis pada beberapa titik yang memiliki tingkat kepadatan lalu lintas, sudut pandang kamera, dan kondisi pencahayaan yang berbeda dengan tujuan untuk membangun dataset yang menggambarkan kondisi nyata lingkungan yang mungkin akan dihadapi oleh model. Data penelitian di dapatkan dari website pemantauan lalu lintas milik dinas perhubungan kota surakarta. Selain itu,untuk meningkatkan kemampuan generalisasi model, data cctv digabungkan dengan dataset publik MIO-TCD dan UA-DETRAC digunakan untuk melatih model dalam menghadapi tantangan spesifik lalu lintas (seperti kepadatan tinggi dan oklusi), sementara COCO digunakan untuk menambah variasi jenis kendaraan dan latar belakang. Kombinasi ini bertujuan untuk menghasilkan model yang tangguh dan akurat di berbagai situasi nyata.

***b. Pre-Processing***

Setelah data video diperoleh, setiap video diekstrak menjadi frame atau citra diam menggunakan perangkat lunak VLC Media Player. Untuk meningkatkan keberagaman data, proses ini juga menyertakan *frame* dari dataset publik yang secara spesifik mengandung objek yang jarang muncul di CCTV, seperti bus dan truk. Selanjutnya, seluruh frame yang dihasilkan dari setiap video dan dataset public akan melalui proses pembersihan untuk memisahkan frame yang memiliki kesamaan visual. Proses pembersihan ini dilakukan menggunakan metode perceptual hashing (pHash), yaitu

algoritma yang mampu mendeteksi kemiripan visual antar gambar secara otomatis. pHash bekerja dengan cara menyederhanakan struktur visual gambar untuk menghasilkan "sidik jari" digital yang unik, sehingga frame yang memiliki kemiripan visual dapat terdeteksi dan dihapus, proses dari cleaning data dapat di. Dengan demikian, hanya frame yang benar-benar unik yang digunakan sebagai bagian dari dataset penelitian.

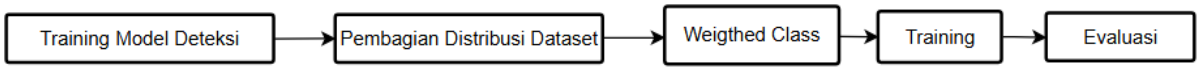
**c. Data Labeling**

Tahap selanjutnya adalah pelabelan data, di mana setiap objek kendaraan pada frame gambar diberikan bounding box (kotak pembatas) dan label kelas secara manual. Proses anotasi ini dilakukan menggunakan platform Roboflow dengan tujuan menyediakan ground truth (kunci jawaban) yang akurat bagi algoritma. Khusus untuk dataset publik, proses pelabelan digantikan dengan proses remapping untuk menyamakan nama label kelas yang digunakan, sehingga konsistensi antar dataset tetap terjaga. Dengan adanya ground truth yang presisi, model dapat belajar untuk mengenali posisi dan jenis kendaraan secara lebih tepat pada setiap gambar, sehingga diharapkan dapat meningkatkan kualitas hasil deteksi dan klasifikasi pada tahap pengujian model.

d. A*ugmentation*

Tahap berikutnya adalah augmentasi data, yaitu proses penambahan variasi pada dataset gambar untuk meningkatkan keragaman dan jumlah dataset. Pada tahap ini, frame yang telah dianotasi berbagai efek seperti rotasi, *cropping*, *scaling*, *flipping*, dan penyesuaian pencahayaan. Proses augmentasi dilakukan secara otomatis menggunakan fitur yang tersedia di platform Roboflow. Dengan adanya augmentasi data, model diharapkan mampu belajar dari beragam kondisi visual sehingga lebih robust dan akurat dalam mengenali objek kendaraan pada berbagai situasi nyata.

**Training Model Deteksi**




**Gambar 3.** Alur Proses Training Model

Setelah dataset diperoleh,langakah selanjutnya melatih model agar bisa mengenali objek, langkah pelatihan ditunjukkan pada Gambar 3.. Pada tahap ini digunakan YOLOv8 (You Only Look Once version 8), yaitu model deteksi objek terbaru yang dikembangkan oleh Ultralytics dan berbasis deep learning *convolutional neural network* (CNN). Model YOLOv8 dipilih karena mampu mengenali serta melokalisasi objek pada gambar secara akurat dan cepat, sehingga sangat mendukung aplikasi *real-time*.

**a. Pembagian Distribusi Dataset**

Sebelum proses pelatihan model dimulai, dataset yang telah diberi label dibagi menjadi tiga bagian, yaitu data latih (training set), data validasi (validation set), dan data uji (test set). Data latih digunakan sebagai sumber utama bagi model untuk belajar mengenali pola objek kendaraan. Data validasi berfungsi untuk memantau performa model secara berkala selama proses pelatihan, serta membantu mencegah terjadinya *overfitting.* Sementara itu, data uji disimpan dan tidak digunakan sama sekali selama pelatihan; data ini berperan sebagai tolok ukur akhir untuk mengevaluasi performa model pada data yang benar-benar baru dan belum pernah dilihat sebelumnya. Proses pembagian dataset dilakukan dengan membagi seluruh data CCTV ke dalam komposisi 70% untuk data latih, 20% untuk data validasi, dan 10% untuk data uji. Sementara itu, seluruh dataset publik sepenuhnya digunakan sebagai data latih agar model memperoleh keberagaman data yang lebih baik

b. Weigthed Class

```
1.  Kelas YOLOWeightedDataset:
2.  Fungsi __init__:
3.      Inisialisasi parent class
4.      Deteksi split dari file gambar
5.      Jika tidak terdeteksi, set split ke "train"
6.      Hitung jumlah instance tiap kelas dari label
7.      Hitung class_weights berdasarkan jumlah instance
8.      Clamp class_weights antara min_weight dan max_weight
9.      Untuk setiap label:
10.         Jika tidak ada kelas, bobot=1.0
11.         Jika ada, bobot = rata-rata class_weights dari kelas di label
12.     Hitung probabilities: bobot dibagi total bobot
13.     Cetak log info dataset
```

**Gambar 4.** Pseudocode Weighted Class

Untuk mengatasi masalah ketidakseimbangan jumlah data antar kelas objek (class imbalance) dalam data latih, diterapkan teknik weighted class. Pada metode ini, setiap kelas diberi bobot yang berbanding terbalik dengan jumlah datanya; kelas dengan sampel lebih sedikit memiliki bobot lebih tinggi. Hal ini memastikan model tidak cenderung bias terhadap kelas mayoritas dan dapat belajar mengenali kelas minoritas secara lebih optimal.

*c. Training*

Setelah Setelah proses pembobotan kelas, dilakukan pelatihan (*training*) akhir terhadap model dengan mempertimbangkan bobot kelas yang telah diterapkan. Model dilatih menggunakan data yang telah terdistribusi sesuai bobot kelas, sehingga diharapkan mampu mendeteksi dan mengklasifikasikan objek kendaraan secara lebih

presisi, terutama pada kelas-kelas dengan frekuensi kemunculan rendah. Proses pelatihan dilakukan selama 100 epoch untuk memaksimalkan pembelajaran, dengan pengaturan learning rate sebesar 0.01. Batch size ditetapkan sebanyak 16 dengan ukuran input gambar 640 piksel. Selain itu, early stopping diaktifkan dengan patience selama 50 epoch agar pelatihan berhenti secara otomatis apabila tidak terjadi peningkatan performa model.Pelatihan model dilaksanakan menggunakan unit pemrosesan grafis (GPU) berbasis CUDA untuk mempercepat proses komputasi. Kombinasi dari penyesuaian hyperparameter tersebut diharapkan dapat meningkatkan akurasi prediksi sekaligus menjaga efisiensi pelatihan dan kemampuan generalisasi model terhadap data yang belum pernah terlihat sebelumnya.

**Pembuatan Algoritma Pelacakan**

Pelacakan objek bertujuan untuk mengidentifikasi dan mengikuti posisi objek kendaraan secara berkelanjutan dalam video. Tantangan utama dalam proses pelacakan adalah menjaga konsistensi identitas setiap objek, meskipun terjadi pergerakan, tumpang tindih antar objek, ataupun hilangnya objek secara sementara dari frame. Pada penelitian ini, algoritma pelacakan dikembangkan dengan dua pendekatan, yaitu SORT dan Hybrid Tracker.

a. SORT (*Simple Online and Realtime Tracking*)

SORT merupakan algoritma pelacakan multi-objek yang menggabungkan Kalman Filter untuk memprediksi posisi objek, serta Hungarian Algorithm untuk mencocokkan objek hasil deteksi baru dengan objek yang sudah terlacak sebelumnya. SORT menggunakan IOU sebagai parameter utama dalam proses pencocokan, sehingga algoritma ini sangat efisien dan cocok untuk aplikasi real-time. Namun demikian, akurasi SORT cenderung menurun ketika objek saling berdekatan, tumpang tindih, atau mengalami perubahan penampilan, karena parameter pencocokan yang digunakan terbatas dan tidak mempertimbangkan bentuk maupun kelas objek.

*b. Hybird Tracking*

```
Fungsi update(deteksi_baru, kelas_baru, confidence_baru):
   Jika deteksi_baru kosong:
      Hapus semua objek yang sedang dilacak
      Selesai
   Jika belum ada objek yang sedang dilacak:
      Untuk setiap deteksi:
         Buat objek baru dan mulai lacak
      Selesai
   Prediksi posisi baru untuk semua objek yang sudah ada
   Untuk setiap kombinasi objek dan deteksi:
      Hitung jarak posisi antara objek dan deteksi
      Hitung perbedaan aspek rasio dan
      Jika kelas berbeda, tambahkan penalti ke biaya
      Jika mode hybrid, biaya_total = jarak + iou_cost + appearance_cost *
      bobot
      (iou_cost = iou_cost_scale * (1 - IOU antar bbox))
   Buat matriks biaya pencocokan: baris = objek, kolom = deteksi
   Lakukan pencocokan objek ke deteksi dengan biaya terkecil (Hungarian
   Algorithm)
   Untuk setiap pasangan objek dan deteksi hasil pencocokan:
      Jika biaya pencocokan di bawah batas:
          Update posisi dan kelas objek sesuai deteksi
          Reset penghitung hilang objek ke 0
          Simpan history kelas untuk stabilisasi
      Jika biaya terlalu besar:
          Tidak dipasangkan
   Untuk objek yang tidak mendapat pasangan:
      Tambah penghitung hilang (disappeared += 1)
      Jika penghitung hilang melebihi batas:
           Hapus objek dari pelacakan (deregister)
   Untuk deteksi baru yang tidak terpasang:
      Jika ada objek yang baru saja hilang dan posisinya dekat:
           Sambungkan kembali deteksi ke objek lama (re-ID)
      Jika tidak:
           Buat objek baru dari deteksi ini
```

**Gambar 5** . *Pseudocode Hybrid Trcaker*

*Hybrid Tracker* merupakan pengembangan dari algoritma pelacakan berbasis Kalman Filter dan Hungarian Algorithm dengan penambahan beberapa parameter pada proses pencocokan. Selain jarak centroid, Hybrid Tracker juga memanfaatkan *Intersection over Union* (IoU), penalti terhadap perubahan kelas objek, serta mekanisme *recovery* untuk objek yang sempat hilang dari frame. Dengan menggabungkan berbagai parameter ini, Hybrid Tracker mampu menghasilkan pelacakan yang lebih akurat dan fleksibel, terutama dalam menangani kasus objek yang mirip atau sering mengalami tumpang tindih. Meskipun demikian, kompleksitas komputasi yang dibutuhkan oleh Hybrid Tracker lebih tinggi dibandingkan dengan SORT

**Implementasi Algoritma Deteksi dan Pelacakan**

```
1.  Buka video (bisa dari file, RTSP, HTTP, atau YouTube)
2.  Inisialisasi detektor objek (YOLO) dan tracker (MultiObjectTracker)
3.  Untuk setiap frame:
4.     Baca frame berikutnya dari stream video
5.     Jalankan deteksi objek untuk mendapatkan kotak (bbox), class, dan
       confidence
6.     Jika tracker aktif:
7.        Update tracker dengan hasil deteksi
8.        Hapus track duplikat jika ada (pakai IOU)
9.        Untuk setiap objek yang sedang dilacak:
10.           Gambar bounding box, ID, class, dan confidence pada frame
11.           Update dan gambar garis pelacakan (jejak) jika fitur aktif
12.    Jika tracker tidak aktif:
13.        Tampilkan hasil deteksi saja, tanpa tracking dan jejak
14.        Reset semua history tracking
15.    Tampilkan FPS, status tracker, status garis pelacakan, dan debug di
       frame
16.    Tampilkan frame hasil ke layar
17.    Setelah selesai:
18.        Tutup stream video dan semua window
```

**Gambar 6.** Pseudocode Implementasi Deteksi dan pelacakan

Setelah semua siap, algoritma YOLO dan Kalman Filter diintegrasikan untuk mendeteksi dan melacak kendaraan pada video. YOLO digunakan untuk mendeteksi objek secara cepat dan akurat di setiap frame video, menghasilkan posisi bounding box dan label objek. Hasil deteksi YOLO kemudian menjadi input bagi Kalman Filter, yang memperkirakan posisi serta kecepatan objek secara berkelanjutan, bahkan ketika terjadi kehilangan deteksi sementara. Integrasi kedua algoritma dilakukan secara berurutan: YOLO mendeteksi objek dan Kalman Filter memperbarui serta memprediksi posisi objek pada frame berikutnya. Proses matching antara hasil deteksi dan prediksi dilakukan menggunakan pencocokan berbasis jarak minimum. Jika deteksi tidak tersedia, Kalman Filter tetap melakukan prediksi, dan jejak pelacakan akan dihapus jika objek tidak terdeteksi dalam beberapa frame berturut-turut. Dengan pendekatan ini, sistem mampu mendeteksi dan melacak kendaraan secara andal pada berbagai kondisi video.

**Pengujian Model Deteksi dan Algoritma Pelacakan**

Pengujian ini bertujuan untuk menilai efektivitas algoritma deteksi dan pelacakan kendaraan yang dikembangkan melalui integrasi algoritma YOLO dan Kalman Filter. Penilaian dilakukan dengan menggunakan serangkaian metrik evaluasi yang umum digunakan dalam bidang deteksi dan pelacakan objek, dengan fokus pada kemampuan sistem dalam mengenali, mempertahankan, dan mengikuti pergerakan kendaraan secara akurat dalam video.

**Pengujian Model Deteksi**

Pengujian deteksi kendaraan bertujuan untuk menilai seberapa baik algoritma YOLO dalam mengenali objek kendaraan pada data uji. Evaluasi dilakukan menggunakan beberapa metrik yaitu precision, recall, F1-score dan mAP. Precision (Presisi) berfungsi untuk mengetahui seberapa banyak dari prediksi suatu kelas yang benar-benar tepat, sebagaimana dirumuskan pada Persamaan (1)

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

Di mana (TP) (*True Positive*) adalah jumlah kendaraan yang terdeteksi dengan benar, dan (FP) (False Positive) adalah jumlah deteksi yang salah. *Recall* mengukur sejauh mana sistem mampu mendeteksi seluruh kendaraan yang sebenarnya ada recall dirumuskan dengan persamaan (2)

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

dengan (FN) (False Negative) sebagai jumlah kendaraan yang tidak terdeteksi. Recall yang tinggi menandakan model mampu menangkap hampir seluruh kendaraan yang ada pada frame. F1-Score, yaitu rata-rata dari precision dan recall, dihitung dengan persamaan (3)

$$F1\text{-}score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (3)$$

Nilai F1-score yang tinggi menunjukkan bahwa sistem tidak hanya mampu mendeteksi kendaraan secara akurat, tetapi juga tidak banyak melewatkan kendaraan yang seharusnya terdeteksi. *Mean Average Precision* (mAP) adalah rata-rata dari nilai Average Precision (AP) untuk setiap kelas objek, mAp dirumuskan dengan persamaan (4)

$$\text{mAP} = \frac{1}{n}\sum_{i=1}^{n} A\,P_i \quad (4)$$

dengan $n$ adalah jumlah kelas objek, *Average Precision* (AP) merupakan rata-rata nilai precision pada berbagai tingkat recall yang dihasilkan dari seluruh threshold confidence (0-1) untuk satu kelas objek. Nilai AP menggambarkan performa model dalam mendeteksi objek pada berbagai level keyakinan (*confidence*) dan *recall*. Semakin tinggi nilai mAP, semakin baik dan konsisten sebuah model dalam mendeteksi serta mengklasifikasikan objek pada seluruh kelas yang diuji.

**Pengujian Algoritma Pelacakan**

Pengujian pelacakan bertujuan untuk menilai performa algoritma tracker dalam mengikuti pergerakan kendaraan dan menjaga konsistensi identitas objek antar frame pada video. Pengujian pelacakan multi-objek pada penelitian ini dilakukan menggunakan TrackEval, sebuah toolkit evaluasi tracking yang telah digunakan secara luas sebagai standar di komunitas penelitian pelacakan objek. Data uji disusun dan diformat sesuai standar yang ditetapkan oleh TrackEval, sehingga hasil evaluasi dapat dibandingkan secara adil dan akurat dengan penelitian lain. Evaluasi pelacakan dilakukan dengan beberapa matriks yaitu *IDS, MT, ML, IDF1, MOTA*. *ID Switches* (IDS) mengukur jumlah pergantian identitas objek yang tidak semestinya; semakin rendah nilainya, semakin baik konsistensi pelacakan. *Mostly Tracked (MT*) menunjukkan persentase objek yang berhasil dilacak selama sebagian besar waktunya; nilai tinggi berarti pelacakan stabil. Sebaliknya, *Mostly Lost (ML*) mengukur objek yang hanya terlacak sebagian kecil waktunya, semakin tinggi nilai mencerminkan pelacakan yang buruk atau tidak konsisten. *Identity F1 Score (IDF1)* adalah metrik yang mengukur konsistensi pelacakan identitas objek sepanjang waktu dengan menyeimbangkan presisi dan recall identitas, IDF1 dirumuskan dengan persamaan (5)

$$IDF1 = \frac{2 \times IDTP}{2 \times IDTP + IDFP + IDFN} \times 100\% \qquad (5)$$

dengan IDTP (Identity True Positives) sebagai lintasan prediksi yang cocok, IDFP (Identity False Positives) sebagai lintasan prediksi yang salah, dan IDFN (Identity False Negatives) sebagai lintasan ground truth yang tidak terdeteksi. Nilai IDF1 yang tinggi menunjukkan kemampuan tracker dalam mempertahankan identitas objek secara konsisten. *Multiple Object Tracking Accuracy (MOTA)* mengevaluasi akurasi pelacakan dengan memperhitungkan kesalahan deteksi dan pelacakan seperti false negatives, false positives, dan ID switch. MOTA diruumuskan dengan persamaan (6)

$$\text{MOTA} = 1 - \frac{\sum_t (FN_t + FP_t + IDSW_t)}{\sum_t GT_t} \qquad (6)$$

dengan $FN_t$, $FP_t$, dan $IDSW_t$ masing-masing menyatakan jumlah false negatives, false positives, dan ID switches pada frame ke- $t$, serta $GT_t$ adalah jumlah ground truth. Nilai MOTA yang mendekati 1 menunjukkan performa pelacakan yang baik dengan sedikit kesalahan.

**Evaluasi**

Evaluasi dilakukan dengan membandingkan dua sistem, yaitu YOLO yang dipadukan dengan Hybrid Tracker dan YOLO dengan SORT sebagai tracker, berdasarkan hasil pengujian pada berbagai skenario lalu lintas dan lingkungan. Analisis difokuskan pada efektivitas masing-masing kombinasi dalam mendeteksi dan melacak kendaraan, serta kestabilan dan keandalan pelacakan identitas objek pada kondisi seperti oklusi, pencahayaan rendah, sudut pandang kamera yang berbeda, dan tingkat kepadatan kendaraan yang bervariasi. Hasil evaluasi memberikan gambaran menyeluruh mengenai performa kedua sistem dalam aplikasi nyata serta menunjukkan pengaruh pemilihan algoritma tracker terhadap peningkatan kinerja deteksi dan pelacakan kendaraan berbasis YOLO.

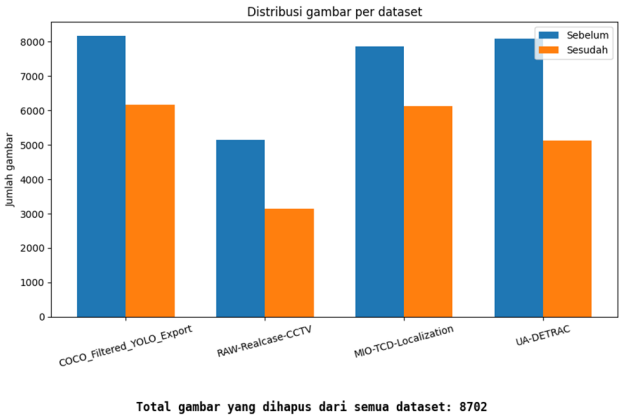
## HASIL

### PERSIAPAN DATA PENELITIAN


**Gambar 7**. Dataset yang Digunakan

*Dataset* dalam penelitian ini dibentuk dari kombinasi rekaman CCTV lalu lintas Kota Surakarta dan tiga *dataset* publik, yaitu COCO, MIO-TCD, dan UA-DETRAC contoh dari *dataset* yang digunakan dapat dilihat pada gambar 7. Setelah dilakukan ekstraksi *frame* dari seluruh video, jumlah gambar yang diperoleh untuk masing-masing *dataset*

adalah sebagai berikut: COCO sebanyak 8.200 gambar, *Realcase*-CCTV sebanyak 5.200 gambar, MIO-TCD-Localization sebanyak 7.900 gambar, dan UA-DETRAC sebanyak 8.100 gambar, sehingga total gambar yang didapat adalah 29.400 gambar. Namun, pada keseluruhan *dataset* masih terdapat frame-frame berurutan yang sangat mirip, sehingga berpotensi menyebabkan bias dan ketidakefisienan dalam proses pelatihan. Untuk mengatasi hal ini, diterapkan proses pembersihan data menggunakan algoritma *perceptual hashing* (pHash), yang mampu mengidentifikasi dan menghapus *frame-frame* serupa secara otomatis.




**Gambar 8.** Distribusi Gambar Setelah Proses Cleaning

Hasil dari proses ini, seperti ditunjukkan pada Gambar 8, COCO sebanyak 6.200 gambar, *Realcase*-CCTV sebanyak 3.200 gambar, MIO-TCD-Localization sebanyak 6.100 gambar, dan UA-DETRAC sebanyak 5.100 gambar, dengan total gambar setelah *cleaning* menjadi 20.600 gambar. Proses ini berhasil menunjukkan penurunan jumlah gambar yang signifikan pada setiap *dataset*, dengan total 8.800 gambar berhasil dihapus. Pengurangan terbesar terjadi pada *dataset* UA-DETRAC dan *Realcase*-CCTV, yang mengindikasikan tingkat kemiripan visual tertinggi di antara sumber data.

(a) (b)

**Gambar 9.** (a) Sebelum Labeling (B) Setelah Labeling

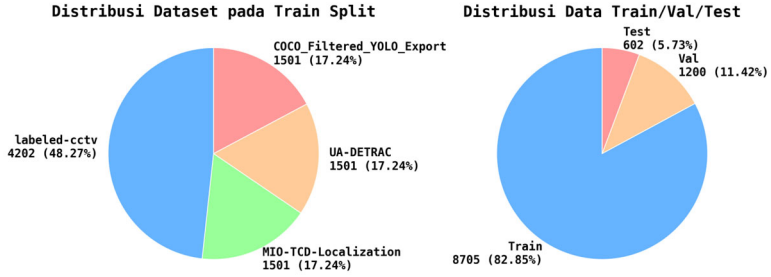
Setelah proses pembersihan, *frame* yang tersisa dianotasi secara manual menggunakan Roboflow, sementara *dataset* publik dilakukan *remapping* label untuk menjaga konsistensi kelas hasil *anotasi* dapat dilihat pada gambar 9. Berdasarkan hasil proses labeling dan *remapping* , diperoleh distribusi objek pada *dataset* sebagaimana ditunjukan pada tabel 1.

**Table 1**.Tabel Jumlah Object Per-Class Tiap Dataset

| | *Realcase*-CCTV | COCO | MIO-TCD | UA-DETRAC |
|---|---|---|---|---|
| Bus | 3094 | 3949 | 8417 | 22172 |
| *Car* | 5885 | 12245 | 69289 | 67669 |
| *Motorbike* | 5240 | 3501 | 1092 | 3954 |
| *Truck* | 4501 | 6123 | 29264 | 3423 |

Selanjutnya, dilakukan *augmentasi* data dengan teknik seperti *rotasi*, *flipping*, dan penyesuaian pencahayaan guna memperkaya variasi visual. Kombinasi dari proses pembersihan, pelabelan yang akurat, dan *augmentasi* sistematis menghasilkan *dataset* akhir yang bersih, variatif, dan representatif terhadap kondisi lalu lintas nyata, sehingga siap digunakan untuk pelatihan dan evaluasi model deteksi serta pelacakan kendaraan

## PEMBUATAN DAN PENGUJIAN MODEL DETEKSI




**Gambar 10**.Distribusi Spliting Dataset

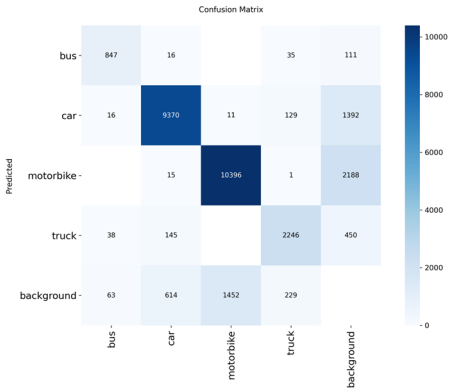
Proses pelatihan model deteksi objek diawali dengan melakukan pembagian *dataset* ke dalam tiga kelompok, yaitu data latih (*train*), data validasi (*val*), dan data uji (*test*). Pada tahap ini, *dataset Realcase*-CCTV digunakan sebagai *dataset* utama, sedangkan *dataset* lain seperti COCO, UA-DETRAC, dan MIO-TCD hanya berperan sebagai *dataset* tambahan untuk data latih. Penggunaan *dataset* tambahan dibatasi agar tidak melebihi jumlah *dataset* utama, dan hanya gambar dengan objek minoritas, yakni bus dan *truck* berdasarkan Tabel 1, yang diambil dari *dataset* tambahan. Seluruh data validasi dan data uji diambil secara eksklusif dari *dataset Realcase*-CCTV untuk memastikan evaluasi performa model mencerminkan kondisi nyata di lapangan. Distribusi *dataset* yang digunakan terdiri dari 10.507 gambar hasil pembersihan visual, yang selanjutnya dibagi menjadi 8075 ( 82,85%) gambar untuk data latih, 1200(11,42%) gambar untuk data validasi, dan 602(5,73%) gambar untuk data uji, sebagaimana ditunjukkan pada Gambar 10.

Data latih terdiri atas kombinasi *dataset realcase* (48,28%) dan data publik dari MIO-TCD, UA-DETRAC, serta COCO (masing-masing 17,24%) sedangkan untuk data validasi dan data tes diambil dari *dataset realcase*. Setelah itu model dilatih menggunakan arsitektur YOLOv8 selama 100 *epoch* dengan nilai learning rate sebesar 0,01, *batch size* 16, dan resolusi gambar 640×640 *piksel*, serta didukung oleh GPU berbasis CUDA untuk akselerasi komputasi. Untuk mengoptimalkan hasil pelatihan, diterapkan strategi *weighted class* guna mengatasi ketidakseimbangan distribusi antar kelas, serta *early stopping* dengan *patience* selama 50 *epoch* untuk mencegah *overfitting* ketika performa validasi tidak mengalami peningkatan. Dengan implementasi strategi pelatihan dan distribusi data yang terstruktur tersebut, model yang dihasilkan diharapkan memiliki performa yang *robust*, efisien, dan siap untuk diterapkan pada sistem pemantauan lalu lintas di dunia nyata

**Table 2.** Tabel Evaluasi Training Model

| Metrik | Bus | Car | Motorbike | Truck | mAP@0.5 | mAP@0.5:0.95 |
|---|---|---|---|---|---|---|
| Precision | 0.905 | 0.910 | 0.884 | 0.872 | - | - |
| Recall | 0.903 | 0.912 | 0.815 | 0.864 | - | - |
| F1-score | 0. 904 | 0. 911 | 0. 848 | 0. 868 | - | - |
| mAP per class | 0. 955 | 0. 962 | 0. 914 | 0. 931 | 0. 941 | 0. 823 |

Evaluasi kinerja model deteksi objek menunjukkan hasil yang bervariasi antar kelas, sebagaimana dirangkum dalam Tabel 2. Model menunjukkan performa sangat baik dalam mendeteksi kendaraan besar, dengan kelas bus dan *car* masing-masing mencapai nilai *Precision*, *recall*, dan *F1-score* diatas 0.90. Kelas *truck* juga menunjukkan performa baik, meski sedikit di bawah bus dan *car*. Sebaliknya, performa terlemah terlihat pada kelas *motorbike*, dengan nilai *recall* 0.815 dan *F1-score* 0.848. Hal ini mengindikasikan bahwa deteksi kendaraan kecil masih menjadi tantangan utama. Secara keseluruhan, akurasi deteksi model tergolong tinggi dengan nilai mAP@0.5 sebesar 0.941. Namun, nilai ini menurun menjadi 0.823 pada metrik mAP@0.5:0.95, yang menunjukkan bahwa ketepatan prediksi *bounding box* menjadi tantangan utama pada ambang *Intersection over Union* (IoU) yang lebih ketat




**Gambar 11.** Confussion Matrix

Analisis pola kesalahan pada model deteksi objek dilakukan menggunakan *confusion matrix* seperti pada Gambar 11. *Confusion matrix* berbeda dengan tabel evaluasi YOLO karena diambil pada satu nilai *confidence* tertentu, sehingga memberikan gambaran detail distribusi prediksi dan kesalahan model pada *threshold* tersebut. Berdasarkan *confusion matrix*, model menunjukkan performa sangat baik pada kelas *car* dan *motorbike*

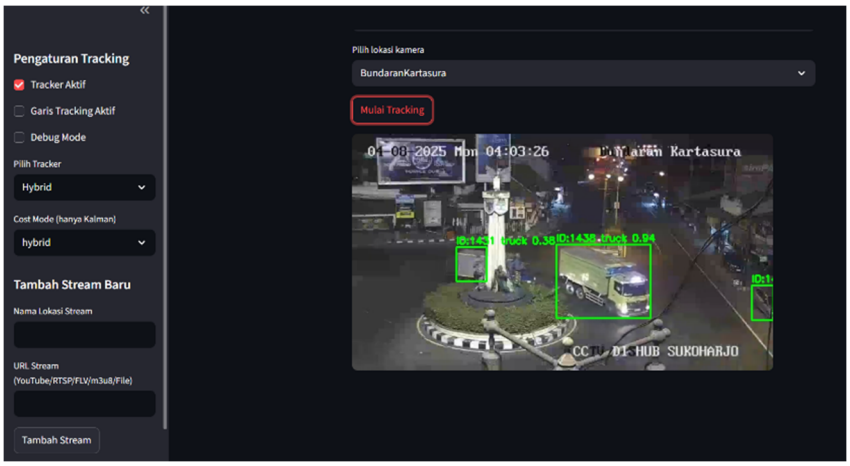
dengan jumlah prediksi benar masing-masing 9.370 dan 10.996. Kesalahan silang antar kelas ini tergolong rendah, misalnya *car* yang diprediksi sebagai bus hanya 16, dan *motorbike* sebagai *car* hanya 15. Namun, pada kelas *truck* dan *background*, kesalahan prediksi cukup signifikan. Sebanyak 145 *truck* terdeteksi sebagai *car*, 450 sebagai *background*, dan 229 sebagai *motorbike*, menandakan ambiguitas dalam membedakan *truck* dari kelas lain. Selain itu, *False Negative* pada *background* cukup tinggi, yaitu 614 *car* dan 1.452 *motorbike* terdeteksi sebagai *background*. Angka *False Positive* juga besar, seperti 2.188 *background* yang terdeteksi sebagai *motorbike* dan 1.392 sebagai *car*. Hal ini menunjukkan model masih kesulitan mendeteksi objek berukuran kecil atau dengan fitur visual mirip, sehingga berdampak pada penurunan recall dan *F1-score*. Meskipun demikian, pengujian pada video CCTV menunjukkan mayoritas kendaraan tetap terdeteksi dengan baik sabagaimana terlihat dalam gambar 12(a).


(a) (b)

**Gambar 12**. (a) Kondisi Citra Sebelum Pelabelan. (B) Kondisi Citra Setelah Pelabelan

## PEMBUATAN DAN IMPLEMENTASI ALGORITMA PELACAKAN

Algoritma pelacakan pada penelitian ini mengintegrasikan YOLOv8 untuk deteksi kendaraan dengan dua pendekatan pelacakan objek, yaitu SORT dan *Hybrid Tracker* berbasis *Kalman Filter* yang telah dimodifikasi. Setiap *frame* video diolah menggunakan YOLOv8 untuk mendeteksi objek, lalu hasil deteksi menjadi *input* bagi *tracker*. *Hybrid Tracker* menggabungkan prediksi *Kalman Filter*, algoritma pencocokan *Hungarian* berbasis *centroid*, serta tambahan pencocokan IoU, penalti perubahan kelas, dan mekanisme *recovery* jika objek hilang sementara. Pendekatan ini meningkatkan efisiensi pelacakan dalam aplikasi *real-time*, khususnya saat terjadi tumpang tindih atau perubahan penampilan objek.




**Gambar 13.** Aplikasi Deteksi dan Tracker

Seluruh proses deteksi dan pelacakan diimplementasikan secara *modular* dengan Python dan OpenCV, lalu dibangun dalam aplikasi berbasis Streamlit yang interaktif. Pengguna dapat memilih mode *tracker*, mengatur metode pencocokan biaya, serta menambah sumber stream video CCTV secara dinamis melalui sidebar aplikasi. Hasil pelacakan divisualisasikan secara *real-time* berupa *bounding box*, label kelas, *confidence score*, dan lintasan objek pada tampilan utama Streamlit. Dengan desain fleksibel dan kemudahan konfigurasi, aplikasi ini mendukung berbagai kebutuhan monitoring lalu lintas atau keamanan dari beragam sumber video, sekaligus memungkinkan troubleshooting langsung jika terjadi masalah pada stream.tampilan aplikasi dapat dilihat pada gambar 13.

## PENGUJIAN ALGORITMA PELACAKAN

Pengujian algoritma pelacakan objek dilakukan dengan membandingkan dua pendekatan, yaitu YOLOv8 + SORT dan YOLOv8 + Hybrid Tracker (berbasis Kalman Filter yang dimodifikasi), menggunakan toolkit evaluasi pelacakan TrackEval. Penilaian dilakukan berdasarkan sejumlah metrik standar, meliputi MOTA, IDSW, MT, ML, IDF1. Hasil evaluasi disajikan dalam Tabel 3.

**Table 3**. Tabel Evaluasi Pelacakan

| | MOTA | IDSW | MT | ML | IDF1 |
|---|---|---|---|---|---|
| Hybird | **66.36** | 71 | **5706** | **310** | **0.88** |
| Sort | 63.65 | **52** | 4836 | 1181 | 0.83 |

Seperti yang ditunjukkan dalam Tabel 2, Tracker Hybird secara konsisten unggul dalam aspek stabilitas pelacakan dan akurasi identitas objek. Hal ini terlihat dari nilai

MOTA sebesar 66,36, lebih tinggi dibandingkan dengan SORT yang hanya mencapai 63,65. Ini berarti sistem pelacakan Tracker Hybird mampu mengurangi false positive, false negative, dan ID switch lebih baik daripada SORT.Pada aspek konsistensi identitas, nilai ID Switches (IDS) pada Tracker Hybird tercatat sebanyak 71 kejadian, sedangkan SORT mencatat 52 kejadian. Meskipun SORT memiliki jumlah ID switch lebih rendah, Tracker Hybird tetap mampu menjaga konsistensi pelacakan objek dengan nilai Mostly Tracked (MT) sebesar 94,8% dan Mostly Lost (ML) hanya sebesar 5,2%. Nilai ini menunjukkan bahwa sebagian besar objek berhasil diikuti selama masa kemunculannya. Dari segi IDF1, Tracker Hybird juga mencatat skor yang lebih baik, yaitu 0,88 dibandingkan SORT yang hanya 0,83. Nilai IDF1 yang lebih tinggi menunjukkan pencocokan identitas yang lebih konsisten dan minim fragmentasi lintasan.

Lebih lanjut, Tracker Hybird mencatat jumlah fragmen lintasan (FRAG) sebanyak 0, sama dengan SORT (0), sehingga kedua tracker mampu menjaga lintasan objek tanpa terlalu sering kehilangan jejak,nilai MOTP SORT sedikit lebih tinggi daripada Hybrid, menunjukkan bahwa SORT memiliki akurasi posisi (bounding box overlap) yang lebih baik dalam tracking. Namun, jika dilihat dari metrik HOTA (*Higher-Order Tracking Accuracy*), kedua tracker mencatat nilai yang sama yaitu 6,74. Hal ini mengindikasikan bahwa dalam hal akurasi spasial dan temporal secara simultan, Tracker Hybird dan SORT memiliki performa yang setara pada data ini. Secara keseluruhan, Tracker Hybird menawarkan pelacakan yang lebih konsisten, akurat, dan andal pada lingkungan lalu lintas nyata seperti yang terdapat dalam data CCTV. Sementara SORT lebih unggul dalam efisiensi dan kecepatan komputasi, performanya menurun ketika menghadapi objek yang saling tumpang tindih, bergerak cepat, atau sering berubah orientasi. Oleh karena itu, untuk sistem pemantauan lalu lintas yang membutuhkan keandalan pelacakan jangka panjang, pendekatan Tracker Hybird menjadi pilihan yang lebih tepat sebagaimana disajikan dalam Tabel 2. Pengujian dilanjutkan dengan uji coba algoritma pelacakan pada video cctv secara langung, hasilnya obejct barhasil dilacak dengan baik ditunjukan pada garis *tracking* pada *object*, visualisasi dapat dilihat pada gambar 10.

## PEMBAHASAN

Berdasarkan seluruh tahapan perancangan, pelatihan, dan pengujian, sistem deteksi dan pelacakan kendaraan berbasis YOLOv8 yang dikembangkan menunjukkan kinerja yang sangat baik. Model deteksi YOLOv8 menunjukkan performa sangat baik berdasarkan hasil evaluasi, dengan nilai mAP@0.5 sebesar 0.941 yang menandakan tingkat akurasi deteksi tinggi secara keseluruhan. Pada kelas bus dan *car*, model mampu mencapai nilai *Precision* diatas 0.90 dan *F1-score* diatas 0.90, menandakan kemampuan sangat baik dalam mendeteksi kendaraan berukuran besar dan sedang. Kelas *truck* juga

menunjukkan performa baik, meskipun sedikit di bawah *bus* dan *car*. Sebaliknya, performa pada kelas *motorbike* masih menjadi tantangan, tercermin dari nilai *recall* sebesar 0.807 dan *F1-score* sebesar 0.852, yang menandakan masih adanya objek *motorbike* yang tidak terdeteksi atau salah klasifikasi.

Penetapan *confidence threshold* sebesar 0.28 dipilih untuk menyeimbangkan antara tingkat akurasi dan minimasi kesalahan deteksi, sehingga model dinilai layak untuk diterapkan pada sistem pemantauan lalu lintas. Pada tahap pelacakan. Kalman*Tracker* secara konsisten unggul dibandingkan SORT, dengan skor MOTA 66.36 dan jumlah *ID Switches* yang relatif rendah, membuktikan stabilitas dan keandalan pelacakan dalam kondisi lalu lintas padat. Kalman*Tracker* mencatat *Mostly Tracked* (MT) sebesar 94,8%, *Mostly Lost* (ML) hanya 5,2%, dan IDF1 sebesar 0,88, menandakan keberhasilan pelacakan objek secara konsisten. Sementara itu, SORT tetap efisien secara komputasi dan memiliki *ID Switch* lebih rendah, namun kurang akurat untuk skenario pemantauan jangka panjang, dengan MOTA dan IDF1 yang lebih rendah serta *Mostly Lost* yang lebih tinggi. Secara keseluruhan, kombinasi YOLOv8 dan *Hybrid Tracker* berhasil menyediakan solusi akurat dan andal untuk deteksi serta pelacakan kendaraan dari CCTV, dan dapat menjadi dasar pengembangan sistem pemantauan lalu lintas cerdas.

## KESIMPULAN

Penelitian ini telah berhasil mengembangkan sistem deteksi dan pelacakan kendaraan pada rekaman CCTV di Kota Surakarta berbasis model YOLOv8. Hasil evaluasi menunjukkan bahwa sistem mampu mendeteksi kendaraan dengan tingkat ketepatan yang sangat baik, tercermin dari capaian mAP sebesar 0.947. Performa deteksi sangat tinggi tercapai pada kendaraan berukuran besar seperti bus dan mobil, sementara untuk kendaraan berukuran kecil seperti sepeda motor masih terdapat ruang untuk perbaikan. Penetapan nilai ambang kepercayaan yang optimal memberikan keseimbangan antara keberhasilan deteksi dan minimnya kesalahan. Pada aspek pelacakan, integrasi sistem deteksi dengan metode pelacakan berbasis *Hybrid Tracker* terbukti menghasilkan akurasi pelacakan yang tinggi dan kestabilan identitas objek yang lebih baik jika dibandingkan dengan metode SORT. Kombinasi model deteksi dan mampu mempertahankan identitas kendaraan secara konsisten meskipun dalam kondisi lalu lintas yang padat dan dinamis, serta menunjukkan tingkat kehilangan identitas yang rendah dan sebagian besar objek dapat dipantau secara berkelanjutan. Secara keseluruhan, kombinasi model deteksi dan pelacakan yang diterapkan dalam penelitian ini menghasilkan pemantauan lalu lintas dengan kinerja yang baik dan dapat diaplikasikan dalam lingkungan nyata. Untuk pengembangan lebih lanjut, disarankan agar sistem meningkatkan kemampuannya dalam mengenali kendaraan berukuran kecil

melalui perbaikan kualitas data, penambahan variasi contoh objek dalam pelatihan, dan peningkatan efisiensi pemrosesan agar dapat dioperasikan pada perangkat dengan sumber daya terbatas. Selain itu, kemampuan model dapat diperluas dengan fitur-fitur analitik tambahan seperti penghitungan kendaraan, pemantauan kecepatan, dan deteksi pelanggaran lalu lintas guna mendukung pengelolaan lalu lintas yang lebih efektif dan berbasis data di masa depan.

## UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada instansi pemerintah terkait, khususnya Dinas Perhubungan (Dishub) kota solo, yang telah menyediakan fasilitas akses publik terhadap siaran langsung (live stream) CCTV. Ketersediaan data visual secara *real-time* ini telah memberikan kontribusi yang signifikan bagi pengumpulan data dalam penelitian ini. Selain itu penulis juga mengucapkan terimkasih kepada dosen,petugas laboratorium,dan seluruh teman yang telah mendukung sehingga penelitian ini bisa diselesaikan.

## DAFTAR PUSTAKA

[1] A. A. Tayeb, R. W. Aldhaheri, and M. S. Hanif, "Vehicle Speed Estimation Using Gaussian Mixture Model and Kalman Filter," *International Journal of Computers, Communications and Control*, vol. 16, no. 4, pp. 1–11, 2021, doi: 10.15837/ijccc.2021.4.4211.

[2] A. S. Kusuma, A. I. Pradana, and B. W. Pamekas, "Pengembangan Sistem Perhitungan Jumlah Kendaraan Berdasarkan ," Jul. 2024.

[3] S. Fulari, *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*. IEEE, 2020.

[4] M. Fiallos-Salguero, S.-T. Khu, J. Guan, T. Wang, and M. Wang, "Computer Vision-Based Method for Rainfall Estimation Using CCTV Cameras and Smartphone Videos," Research Publishing Services, Jun. 2024, pp. 320–322. doi: 10.3850/iahr-hic2483430201-284.

[5] F. T. Hidayat and A. K. Whardana, "Deteksi Pelanggaran Sepeda Motor Menggunakan Algoritma Yolo Dan Mean Average Precision," 2024.

[6] J. N. Subekti and A. D. Putra, "Deteksi Trafic Pada Jumlah Kendaraan Yang Lewat Di You See Indonesia Menggunakan Metode Yolov8," *Information System Journal*, vol. 7, no. 02, pp. 78–86, Feb. 2025, doi: 10.24076/infosjournal.2024v7i02.1736.

[7] N. Charibaldi, "Penerapan Object Detection Menggunakan Deep Learning Yolov8 untuk Mengidentifikasi Sampah Anorganik (Maksimal Sepuluh Objek) dalam Satu Citra," vol. 12, no. 1, pp. 195–202, 2025, doi: 10.25126/jtiik.2025129012.

[8] S. Chaudhari, N. Malkan, A. Momin, and M. Bonde, "Yolo Real Time Object Detection," *International Journal of Computer Trends and Technology*, vol. 68, no. 6, pp. 70–76, Jun. 2020, doi: 10.14445/22312803/ijctt-v68i6p112.

[9] H. Achmad, S. . M. Gumilang, A. Pramudwiatmoko, B. Al Karim, and H. Wiyono, "Analisis Kinerja Model Deteksi Objek Yolo, Ssd, dan Faster R-Cnn pada Citra Penglihatan Malam untuk Pengenalan Tindak Kejahatan," *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 12, no. 1, pp. 145–152, Feb. 2025, doi: 10.25126/jtiik.2025128409.

[10] M. P. Anugrah, B. Fatkhurrozi, and H. T. Setiawan, "Jurnal Vocational Teknik Elektronika dan Informatika Deteksi Helm Pengendara dan Plat Nomor Kendaraan pada CCTV Lampu Lalu Lnitas Menggunakan Algoritma YOLO," Mar. 2024, [Online]. Available: http://ejournal.unp.ac.id/index.php/voteknika/index

[11] M. Y. Kurniawan, M. E. Rosadi, and F. Hafidh, "Klasifikasi Tipe Kendaraan dari Video CCTV Menggunakan Metode Yolo," *Technologia : Jurnal Ilmiah*, vol. 15, no. 2, p. 356, Apr. 2024, doi: 10.31602/tji.v15i2.14498.

[12] T. Maulana and T. Harahap, "Deteksi Plat Nomor Kendaraan Menggunakan Algoritma YOLOv5 dengan Metode Convolutional Neural Network," *Jurnal Riset Matematika*, pp. 103–112, Dec. 2024, doi: 10.29313/jrm.v4i2.5060.

[13] M. Muzammul, X. Li, and X. Li, "Enhancing Tiny Object Detection Using Guided Object Inference Slicing (GOIS): An efficient dynamic adaptive framework for fine-tuned and non-fine-tuned deep learning models," *Neurocomputing*, vol. 640, p. 130327, Aug. 2025, doi: 10.1016/J.NEUCOM.2025.130327.

[14] H. Wang, W. Wang, and Y. Liu, "X-YOLO: A deep learning based toolset with multiple optimization strategies for contraband detection," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, May 2020, pp. 127–132. doi: 10.1145/3393527.3393549.

[15] T. N. Doan and M. T. Truong, "Real-time vehicle detection and counting based on YOLO and DeepSORT," in *Proceedings - 2020 12th International Conference on Knowledge and Systems Engineering, KSE 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020, pp. 67–72. doi: 10.1109/KSE50997.2020.9287483.

[16] H. Yi, B. Liu, B. Zhao, and E. Liu, "Small Object Detection Algorithm Based on Improved YOLOv8 for Remote Sensing," *IEEE J Sel Top Appl Earth Obs Remote Sens*, vol. 17, pp. 1734–1747, 2024, doi: 10.1109/JSTARS.2023.3339235.

[17] Tejashree K, P. Ashish, R. Gorai, R. Sah, and S. Kumar, "SPEED DETECTION OF MOVING VEHICLE USING YOLO," *International Research Journal of Modernization in Engineering Technology and Science*, Feb. 2025, doi: 10.56726/IRJMETS67043.