Journal of Information Technology and Computer Science (INTECOMS)
Volume 9 Nomor 2, Tahun 2026
e-ISSN : 2614-1574
p-ISSN : 2621-3249

SISTEM MONITORING DAN ESTIMASI LAMA WAKTU KUNJUNGAN PELANGGAN MENGGUNAKAN ALGORITMA YOLO BERBASIS DEEP LEARNING (STUDI KASUS: JARAK COFFEE & EATERY)

MONITORING SYSTEM AND ESTIMATION OF CUSTOMER VISIT DURATION USING DEEP LEARNING-BASED YOLO ALGORITHM (CASE STUDY: JARAK COFFEE & EATERY)

Melly Anggraini[1], Veronica Lusiana[2]
Program Studi Teknik Informatika, Fakultas Teknologi Informasi dan Industri, Universitas Stikubank Semarang[1, 2]
mellyanggraini0007@mhs.unisbank.ac.id[1], vero@edu.unisbank.ac.id[2]

ABSTRACT

Advances in Artificial Intelligence (AI) and Computer Vision have significantly improved monitoring systems. This study develops an automated system to estimate customer visit duration at Jarak Coffee & Eatery using a deep learning-based YOLO algorithm. Addressing the inefficiency of manual monitoring, the research implements YOLOv11 and BoT-SORT for real-time detection and tracking. The system was developed following the Waterfall model, comprising requirements analysis, system design, implementation, and testing. Results indicate that YOLOv11 achieved high performance with 96.5% mAP@0.5, 93.4% precision, and 93.4% recall. It effectively filters false positives using a 30-second duration threshold. The output provides real-time visualization and CSV logs for visit duration analysis and operational optimization, offering an objective solution for customer behavior monitoring.

Keywords*: YOLO Algorithm, Deep Learning, Computer Vision, Object Detection, Customer Visit Duration*

ABSTRAK

Perkembangan teknologi *Artificial Intelligence* (AI) dan *Computer Vision* telah memberikan dampak signifikan terhadap sistem *monitoring*. Penelitian ini mengembangkan sistem *monitoring* dan estimasi lama waktu kunjungan pelanggan menggunakan algoritma YOLO berbasis *deep learning* di Jarak Coffee & Eatery. Permasalahan utama adalah kurangnya efektivitas metode *monitoring* manual dalam mengelola data kunjungan pelanggan yang menyebabkan kesulitan mendapatkan informasi akurat mengenai durasi kunjungan. Tujuan penelitian ini adalah mengimplementasikan algoritma YOLO untuk mendeteksi dan melacak pelanggan secara *real-time* serta mengembangkan sistem yang mampu menghitung durasi kunjungan secara otomatis. Menggunakan metode *Waterfall* dengan tahapan analisis kebutuhan, perancangan sistem, implementasi menggunakan YOLOv11 dan BoT-SORT, serta pengujian. Hasil model YOLOv11 mencapai performa sangat baik dengan mAP@0.5 sebesar 96,5%, precision 93,4%, dan recall 93,4%, serta mampu menghitung durasi kunjungan dengan *threshold* minimal 30 detik untuk memfilter *false positive*. Sistem menghasilkan output visualisasi *real-time* dan *log CSV* yang dapat digunakan untuk analisis durasi kunjungan dan optimalisasi operasional.

Kata Kunci: Algoritma YOLO, Deep Learning, Computer Vision, Object Detection, Durasi Kunjungan Pelanggan

PENDAHULUAN

Perkembangan teknologi *Artificial Intelligence* (AI) dan *Computer Vision* telah memberikan dampak signifikan terhadap berbagai bidang, termasuk sistem *monitoring* berbasis video. Salah satu teknologi yang mengalami kemajuan pesat adalah *Computer Vision* berbasis *deep learning*, yang mampu melakukan deteksi dan *object tracking* [1]. Proses deteksi tersebut dapat dilakukan melalui berbagai metode yang umumnya melibatkan pembacaan fitur-fitur dari seluruh *object* dalam citra maupun video [2].

Informasi mengenai lama waktu kunjungan pelanggan memiliki peran penting dalam optimalisasi pelayanan, perencanaan kapasitas tempat duduk, serta peningkatan pengalaman pelanggan [3]. Namun, metode manual dalam monitoring

aktivitas pelanggan memerlukan sumber daya manusia yang besar dan rentan terhadap kesalahan. Oleh karena itu, diperlukan sistem otomatis yang mampu melakukan monitoring dengan tingkat ketepatan dan akurasi yang tinggi.

Algoritma YOLO (*You Only Look Once*) adalah suatu algoritma yang dikembangkan khusus untuk tugas-tugas dalam bidang *Computer Vision*, terutama dalam hal *object tracking* [4]. Model YOLO juga dapat diterapkan dalam sistem monitoring menggunakan kamera, sehingga mampu mendeteksi *object* yang diinginkan [5]. Algoritma YOLO dipilih dalam pengembangan sistem ini karena kemampuannya untuk mendeteksi *object* secara langsung dan *real-time* dengan tingkat akurasi yang baik, serta mampu melakukan perhitungan cepat untuk menghasilkan hasil deteksi yang akurat [6].

penelitian ini bertujuan mengembangkan sistem *monitoring* dan estimasi lama waktu kunjungan pelanggan menggunakan algoritma YOLO berbasis *deep learning* di Jarak Coffee & Eatery. Sistem ini diharapkan dapat menyediakan informasi akurat mengenai durasi kunjungan pelanggan secara otomatis, sehingga dapat mendukung pengambilan keputusan manajemen dalam meningkatkan kualitas layanan dan efisiensi operasional [3].

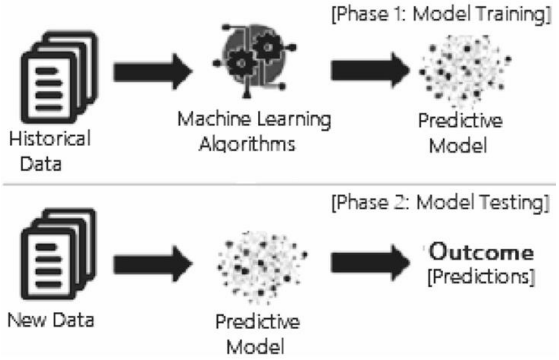
TINJAUAN PUSTAKA

2.1 Artificial Intelligence (AI)

AI adalah cabang ilmu komputer yang fokus pada cara membuat sistem komputer yang bisa berpikir dan bertindak seperti manusia [7]. AI meliputi berbagai teknik dan pendekatan, seperti pembelajaran mesin (*machine learning*), pembelajaran mendalam (*deep learning*), dan pemrosesan bahasa alami [8].

2.2 Machine Learning

Machine learning (Pembelajaran mesin) adalah proses di mana komputer memanfaatkan data untuk meningkatkan performanya dalam tugas tertentu. Proses ini melibatkan algoritma yang dapat mengenali pola dalam data dan kemudian membuat prediksi atau keputusan berdasarkan pola tersebut [9].

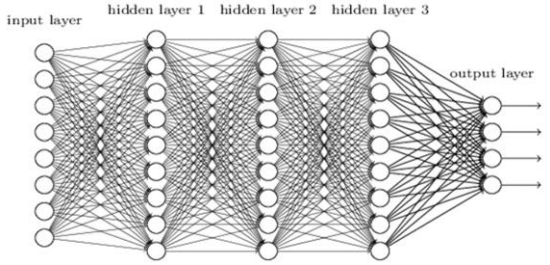


Gambar 1. ***A general structure of a machine learning based predictive*** **[8]**

2.3 Deep Learning

Deep Learning merupakan bagian dari kecerdasan buatan dan *machine learning* yang berkembang dari *neural network* dengan banyak lapisan untuk memberikan akurasi dalam tugas seperti *object detection* [10].

Perbedaan *deep learning* dengan *neural network* yaitu lebih banyaknya *hidden layer* pada *deep learning*. Jika lebih dari tiga layer (termasuk input dan output) maka memenuhi syarat sebagai "*deep*" *learning*. Jadi *deep learning* bisa didefinisikan secara teknis yaitu *machine learning* yang mempunyai lebih dari satu *hidden layer* [11].



Gambar 2. Ilustrasi *Deep Learning* [11]

2.4 Computer Vision

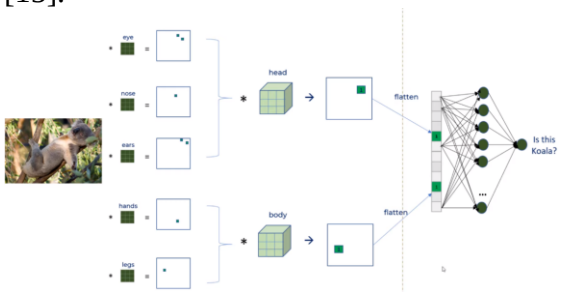
Computer Vision adalah bidang yang memungkinkan mesin untuk "melihat" dengan memanfaatkan kamera dan komputer sebagai alternatif mata manusia, guna mengidentifikasi, melacak, serta mengukur *object* sebelum dilakukan pemrosesan lebih lanjut secara digital [12].

2.5 Object Detection

Object detection adalah kemampuan komputer untuk mendeteksi dan mengidentifikasi *object* tertentu. Kemampuan ini merupakan bagian dari ilmu *computer vision*. Dalam *computer vision*, *object detection* bisa menganalisis video atau gambar untuk mengatasi suatu masalah [13].

2.6 Algoritma YOLO (You Only Look Once)

Algoritma YOLO (*You Only Look Once*) merupakan metode *object detection real-time* yang memanfaatkan *Convolutional Neural Network* [14]. Metode ini telah terbukti lebih unggul dalam hal kecepatan dan akurasi saat mengidentifikasi *object* pada gambar atau citra, sehingga sangat sesuai untuk aplikasi *object detection* secara *real-time* pada video [15].



Gambar 3. Cara Kerja Algoritma YOLO [16]

2.7 BoT-SORT

BoT-SORT (*Botorchvision Simple Online and Realtime Tracking*) merupakan algoritma pelacakan objek (*object tracking*) yang dikembangkan sebagai penyempurnaan dari algoritma SORT (*Simple Online and Realtime Tracking*). Algoritma ini dirancang khusus untuk melakukan pelacakan objek secara *real-time* dengan akurasi tinggi [17]. BoT-SORT menggabungkan beberapa teknik *advanced tracking* untuk meningkatkan performa, termasuk *motion prediction* menggunakan *Kalman Filter*, *appearance feature extraction* menggunakan *deep learning*, dan strategi *re-identification* untuk menangani *occlusion*.

2.8 Sistem Monitoring

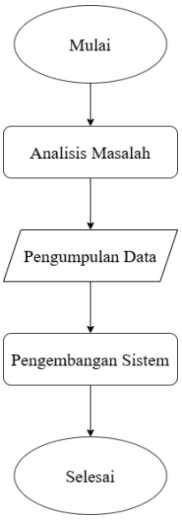
Sistem *monitoring* di sini merujuk pada pemanfaatan teknologi, khususnya *Computer Vision*, untuk mengamati, melacak, dan mencatat aktivitas di suatu area secara otomatis [2]. Berbeda dengan pemantauan manual oleh manusia yang rentan terhadap kelelahan dan bias, sistem *monitoring* otomatis bisa berjalan terus-menerus dengan konsistensi tinggi untuk mendeteksi dan menghitung objek [1].

2.9 Estimasi Lama Waktu Kunjungan

Estimasi lama waktu kunjungan pelanggan adalah proses menghitung durasi yang dihabiskan pelanggan di area tertentu, misalnya di restoran atau kafe. Data dari sistem estimasi durasi ini bisa digunakan untuk berbagai analisis, seperti durasi kunjungan rata-rata, dan pengoptimalan tata letak ruangan [3].

1. METODE

3.1 Flowchart Penelitian



Gambar 4. *Flowchart* Penelitian

3.1.1 Analisis Masalah

Permasalahan utama yang mendasari penelitian ini adalah kurangnya efektivitas metode monitoring manual dalam mengelola data kunjungan pelanggan di Jarak Coffee & Eatery. Metode ini memiliki keterbatasan, terutama terkait dengan faktor *human error* dan ketidak mampuan untuk melakukan analisis data secara langsung saat kondisi kafe sedang ramai. Keterbatasan ini menyebabkan manajemen kesulitan mendapatkan informasi yang

akurat mengenai durasi kunjungan pelanggan, yang padahal merupakan indikator penting untuk efisiensi operasional.

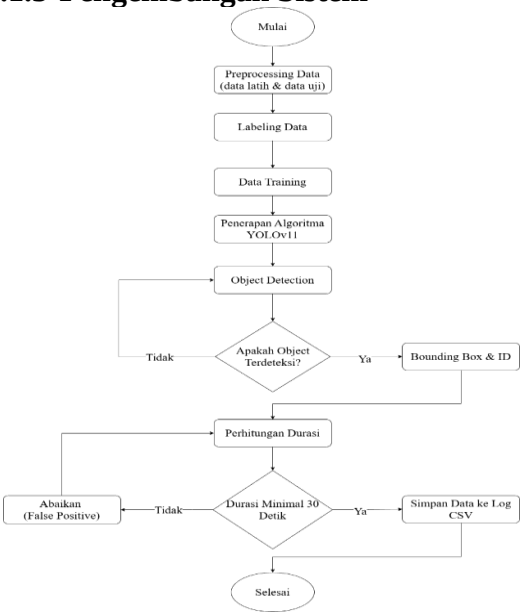
3.1.2 Pengumpulan Data

Tahap ini berfokus pada pengumpulan data yang akan digunakan sebagai input utama untuk sistem. Data dalam penelitian ini adalah rekaman video aktivitas pelanggan di Jarak Coffee & Eatery, yang diperoleh melalui rekaman CCTV langsung di lokasi penelitian.

Perekaman dilakukan pada berbagai kondisi pencahayaan dan tingkat keramaian pengunjung yang bervariasi (sepi, sedang, ramai). Variasi data ini penting untuk memastikan bahwa sistem yang dikembangkan dapat menangani berbagai kondisi operasional.

Gambar 5. Data Video Rekaman CCTV

3.1.3 Pengembangan Sistem



Gambar 6. Alur Kerja (*Workflow*)

a. Preprocessing Data (Data Latih & Data Uji)

Preprocessing dibagi menjadi dua yaitu *preprocessing* untuk data latih dan *preprocessing* untuk data uji.

Data latih berupa *frame-frame* gambar yang diekstrak dari video rekaman di Jarak Coffee & Eatery. Proses ekstraksi dilakukan dengan mengambil *frame* setiap 60 detik dari video untuk mendapatkan variasi. Setiap *frame* yang mengandung objek '*person*' harus dianotasi secara manual menggunakan *tools* Roboflow untuk membuat *bounding box* di sekitar objek.

Untuk data uji, *preprocessing* berfokus pada persiapan. Video uji dipotong (video *trimming*) menjadi klip dengan durasi 30-90 menit dari berbagai kondisi operasional.

b. Labeling Data

Setelah *preprocessing* selesai, tahap selanjutnya adalah labeling data. Proses ini dilakukan secara manual dimana setiap *object* '*person*' yang muncul dalam dataset gambar diberi label dengan menggambar *bounding box* di sekitar *object* tersebut. Proses labeling ini memastikan bahwa model dapat memahami dengan tepat bagaimana bentuk, ukuran, dan karakteristik visual dari *object* '*person*' yang harus dideteksi dalam berbagai kondisi dan posisi.

c. Data Training

Tahap data *training* adalah proses dimana model *machine learning* dilatih menggunakan dataset yang telah dilabeli. Pada tahap ini, model akan belajar mengenali pola-pola visual yang membedakan *object* '*person*' dari *background* dan *object* lainnya. Proses *training* dilakukan melalui beberapa *epoch* dimana model secara bertahap meningkatkan kemampuannya dalam mendeteksi *object*. Selama proses *training*, parameter-parameter model akan disesuaikan secara otomatis untuk meminimalkan kesalahan deteksi.

d. Penerapan Algoritma YOLOv11

Tahap penerapan algoritma YOLO merupakan implementasi dari model yang telah dilatih sebelumnya, file *weights* yang telah disimpan dari hasil *training* memungkinkan sistem untuk memproses video dan mendeteksi keberadaan *object* dengan tingkat akurasi yang tinggi, bahkan dalam kondisi pencahayaan yang bervariasi dan tingkat keramaian yang berbeda-beda.

e. Object Detection

Tahap *object detection* merupakan proses dimana sistem mulai menganalisis video *input*. Pada tahap ini, video yang akan dianalisis dibaca *frame by frame* oleh sistem. Setiap *frame* kemudian diumpankan ke model YOLO yang telah diterapkan sebelumnya. Model akan memproses setiap *frame* dan mencari keberadaan *object* '*person*' di dalamnya dengan menganalisis fitur-fitur yang telah dipelajari selama proses *training.*

f. Bounding Box & ID

Ketika objek berhasil terdeteksi, sistem akan memberikan dua informasi yaitu *bounding box* dan ID unik. *Bounding box* adalah kotak pembatas yang mengelilingi *object* '*person*' yang terdeteksi, dengan koordinat yang menunjukkan posisi *object* tersebut dalam *frame*. Selain *bounding box,* sistem juga memberikan ID unik untuk setiap pelanggan yang terdeteksi. ID ini sangat penting karena akan digunakan untuk melacak pergerakan pelanggan yang sama di *frame-frame* berikutnya. Pemberian ID ini dilakukan oleh algoritma *tracking* (BoT-SORT) yang dapat membedakan satu individu dengan individu lainnya.

g. Perhitungan Durasi

Tahap perhitungan durasi merupakan proses kalkulasi untuk menentukan berapa lama seorang pelanggan dengan ID tertentu terdeteksi di dalam *frame* video.

h. Decision: Durasi Minimal 30 Detik

Setelah durasi kunjungan dihitung, sistem melakukan validasi dengan memeriksa apakah durasi tersebut memenuhi *threshold* minimal yang telah ditetapkan yaitu 30 detik. Tahap validasi ini sangat penting untuk menjaga kualitas data dan menghindari *false positive* atau kesalahan deteksi yang dapat mengotori data analisis.

i. Simpan Data ke Log CSV

Setelah sistem berhasil memproses video, menghasilkan *output* (visualisasi dan log CSV), dan melewati tahap evaluasi. Data-data ini kemudian dapat digunakan oleh manajemen Jarak Coffee & Eatery untuk pengambilan keputusan strategis dalam meningkatkan kualitas layanan dan efisiensi operasional.

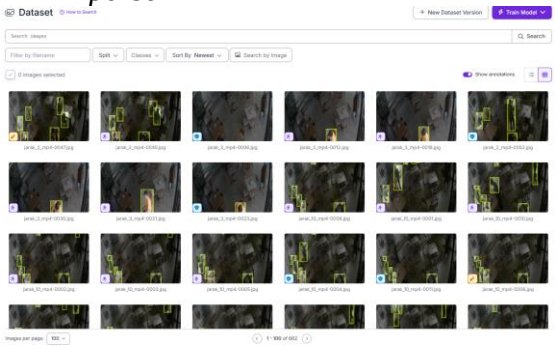
HASIL DAN PEMBAHASAN

4.1 Implementasi

Pada tahap implementasi merupakan tahap penerapan sistem yang telah dirancang. Pada penelitian ini, implementasi diterapkan pada sistem *monitoring* dan estimasi lama waktu kunjungan pelanggan menggunakan algoritma YOLOv11 dan BoT-SORT di Jarak Coffee & Eatery.

4.1.1 Labeling Object

Pada proses pelabelan, terdapat total 662 gambar yang dianotasi untuk dijadikan dataset. Setiap *object* '*person*' yang muncul dalam *frame* diberi *bounding box* dan label kelas "*person*".

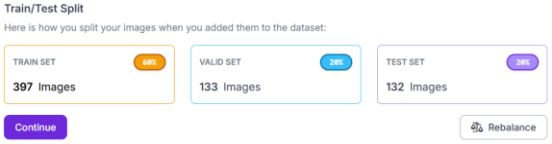


Gambar 7. Dataset Gambar

4.1.2 Pembagian Dataset

Dataset yang telah dilabeli dibagi menjadi tiga subset untuk keperluan pengembangan model, yaitu data *training*

sebanyak 60% (397 gambar), data *validation* sebanyak 20% (133 gambar), dan data *testing* sebanyak 20% (132 gambar).

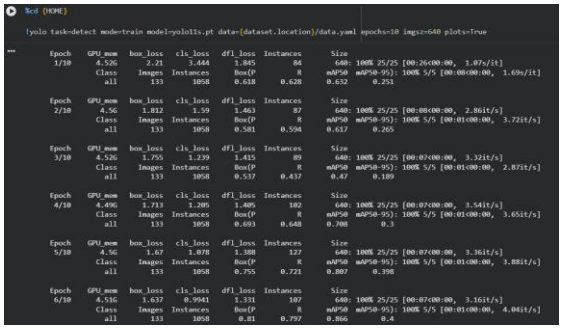


Gambar 8. Pembagian Dataset

4.1.3 Training Data

Selanjutnya ini adalah proses training menggunakan perintah *model.train*(). Proses *training* ditetapkan sebanyak 10 *epoch*, yang berarti model mempelajari keseluruhan dataset dalam 10 *cycles* dengan durasi rata-rata sekitar 15 detik per *epoch*.

Selama proses berjalan, model memperbarui bobotnya untuk meminimalkan nilai *loss*, dengan peningkatan nilai metrik *Precision*, *Recall*, dan mAP@0.5 seiring bertambahnya iterasi. Hal ini mengindikasikan bahwa model sedang belajar mengenali fitur *object* dengan semakin akurat dari waktu ke waktu. Akhir dari proses *training* ini menghasilkan file *weight* terbaik yaitu best.pt, yang nantinya akan digunakan sebagai model utama dalam sistem *object detection*.



Gambar 9. *Training* Model YOLOv11

4.1.4 Implementasi Kode

a. Load Model dan Inisialisasi Video

Muat *weight* model YOLO hasil *training* (best.pt) dan menginisialisasi video *input*. Pada tahap ini, sistem juga membaca properti FPS (*Frames Per Second*) dari video untuk menentukan *frame_time*. Variabel ini untuk memastikan perhitungan durasi kunjungan tetap akurat, terlepas dari kecepatan pemrosesan komputer.

```
# Load model YOLO
model = YOLO("best.pt")

# Buka video
video_path = "data/jarak_1.mp4"
cap = cv2.VideoCapture(video_path)

if not cap.isOpened():
    print(f"Error: Tidak dapat membuka video {video_path}")
    exit()

# FPS video untuk perhitungan waktu yang akurat
video_fps = cap.get(cv2.CAP_PROP_FPS)
if video_fps == 0:
    video_fps = 30  # Default FPS jika tidak terdeteksi
frame_time = 1.0 / video_fps  # Durasi per frame dalam detik
```

Gambar 10. Kode Program *Load* Model dan Video *Input*

b. Perhitungan Durasi per ID

Algoritma perhitungan durasi diimplementasikan. Logika program memeriksa apakah sebuah ID baru terdeteksi atau merupakan ID lama yang kembali muncul. Durasi kunjungan dihitung dengan menjumlahkan durasi sesi saat ini dengan durasi kumulatif sebelumnya (*cumulative_duration*). Jika total durasi (*visit_duration*) telah melebihi ambang batas *VISIT_THRESHOLD* (30 detik), sistem akan menandai status pelanggan tersebut sebagai kunjungan valid (*valid visit*) yang siap untuk dicatat.

```
# Proses setiap person yang terdeteksi
for box, obj_id, conf in persons:
    active_ids.add(obj_id)
    x1_orig, y1_orig, x2_orig, y2_orig = map(int, box)

    # Buat bounding box
    width = x2_orig - x1_orig
    height = y2_orig - y1_orig
    x1 = int(x1_orig + width * 0.05)
    x2 = int(x2_orig - width * 0.05)
    y1 = int(y1_orig + height * 0.05)
    y2 = int(y2_orig - height * 0.05)

    # Inisialisasi tracking time saat pertama kali terdeteksi atau re-deteksi
    if obj_id not in time_tracking:
        # Customer baru atau kembali setelah keluar frame
        time_tracking[obj_id] = current_time

        if obj_id not in cumulative_duration:
            # Benar-benar customer baru
            cumulative_duration[obj_id] = 0
            visit_logged[obj_id] = False

    # Hitung durasi sesi saat ini
    current_session_duration = current_time - time_tracking[obj_id]

    # Total durasi = durasi kumulatif + durasi sesi saat ini
    visit_duration = cumulative_duration[obj_id] + current_session_duration

    # Jika person terdeteksi >= 30 detik, mulai perhitungan (tanpa logging)
    is_tracking = False
    if visit_duration >= VISIT_THRESHOLD:
        is_tracking = True
        # Tandai bahwa customer ini sudah mencapai threshold
        if not visit_logged[obj_id]:
            visit_logged[obj_id] = True
```

Gambar 11. Kode Program Perhitungan Durasi

c. Loop Pemrosesan Frame

Sebagaimana diperlihatkan pada Gambar 12. Di dalam *loop* ini, metode *model. track* dijalankan dengan konfigurasi *persist=True* dan *tracking* botsort.yaml. Konfigurasi ini memungkinkan sistem untuk mempertahankan ID unik yang sama bagi setiap *object* antar *frame*. Parameter

classes=[0] memastikan deteksi hanya difokuskan pada objek manusia (*person*).

```
# Loop pemrosesan frame
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break

    frame_count += 1
    current_time = frame_count * frame_time  # Waktu berdasarkan FPS video

    # Deteksi person (class 0) dengan tracking BOTSort
    results = model.track(frame, persist=True, tracker="botsort.yaml", classes=[0], verbose=False, conf=0.3)
```

Gambar 12. Kode Program Loop Pemrosesan Frame

d. Visuualisasi Bounding Box dan Status

Untuk memberikan visualisasi pada antarmuka sistem, kode mengatur gambar *bounding box* dan teks informasi pada video. Kotak akan berwarna hijau jika durasi kunjungan telah memenuhi syarat (≥30 detik) dan berwarna oranye jika masih dalam tahap deteksi. Informasi durasi ditampilkan tepat di bawah kotak pembatas untuk memudahkan *monitoring*.

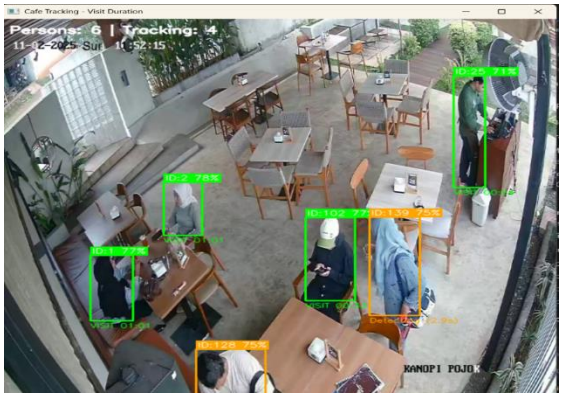
```
# Tentukan warna box dan status berdasarkan durasi kehadiran
if is_tracking:
    box_color = (0, 255, 0)  # Hijau untuk tracking (>= 30 detik)
    status = f"VISIT {format_time(visit_duration)}"
else:
    box_color = (0, 165, 255)  # Orange belum 30 detik
    status = f"Detecting ({visit_duration:.1f}s)"

# Gambar bounding
cv2.rectangle(frame, (x1, y1), (x2, y2), box_color, 2)

# Label ID dengan confidence di atas box
label = f"ID:{obj_id} {conf:.0%}"
(label_w, label_h), _ = cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX, 0.5, 1)
cv2.rectangle(frame, (x1, y1 - label_h - 6), (x1 + label_w + 4, y1), box_color, -1)
cv2.putText(frame, label, (x1 + 2, y1 - 4),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 1, cv2.LINE_AA)

# Status di bawah box
cv2.putText(frame, status, (x1, y2 + 15),
            cv2.FONT_HERSHEY_SIMPLEX, 0.45, box_color, 1, cv2.LINE_AA)
```

Gambar 13. Kode Program Visualisasi *Bounding Box*



Gambar 14. Hasil Visualisasi *Bounding Box*

e. Inisialisasi File CSV untuk Logging

Agar data hasil *monitoring* dapat dianalisis lebih lanjut, sistem menyimpan data otomatis ke dalam format CSV seperti. Sistem terlebih dahulu memeriksa ketersediaan folder "*logs*" dan membuatnya jika belum ada. Nama file CSV dibuat menggunakan *timestamp* saat program dijalankan untuk mencegah penumpukan data (*overwrite*). Header file mencakup kolom *Timestamp*, ID Pelanggan, dan Durasi Kunjungan.

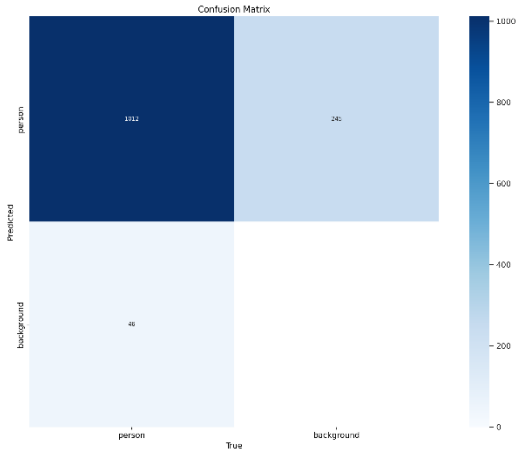
```
# Buat folder logs jika belum ada
if not os.path.exists("logs"):
    os.makedirs("logs")

# Inisialisasi CSV file dengan timestamp
csv_filename = f"logs/customer_duration_{datetime.now().strftime('%Y%m%d_%H%M%S')}.csv"
csv_file = open(csv_filename, 'w', newline='', encoding='utf-8')
csv_writer = csv.writer(csv_file)
csv_writer.writerow(['Timestamp', 'Customer_ID', 'Duration_Seconds', 'Duration_Formatted'])
```

Gambar 15. Kode Program Inisialisasi CSV Logging

4.2 Pengujian

Tahap pengujian bertujuan untuk mengevaluasi performa sistem *monitoring* yang telah dibangun. Kinerja sistem diukur menggunakan *confusion matrix*, yaitu *Precision*, *Recall*, *F1-Score*, dan *Mean Average Precision* (mAP). Komponen *confusion matrix* terdiri dari 4 kelas yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN) [3].



Gambar 16. *Confusion Matrix*

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

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

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3)$$

$$mAP = \frac{1}{n}\sum_{i=1}^{n} AP_i \quad (4)$$

Tabel 1. Performa Model YOLOv11

No	Metrik	Nilai
1	Precision	93.4%
2	Recall	93.4%

3	F1-Score	90.0%
4	mAP@0.5	96.5%

Hasil menunjukkan bahwa model mencapai performa yang sangat baik dengan nilai *precision* 93.4% dan *recall* 93.4%. Nilai mAP@0.5 sebesar 96.5% mengindikasikan bahwa model mampu mendeteksi objek '*person*' dengan akurasi tinggi. Nilai *F1-Score* 90.0% menunjukkan keseimbangan yang baik antara *precision* dan *recall*.

Tabel 2. Hasil Pengujian Sistem Pada Video Rekaman

Tanggal	Waktu	Person Terdeteksi	Customer Valid (≥30s)	Avg Confidence
10-29-2025	Siang	3	2	78%
10-25-2025	Malam	14	12	73%
11-01-2025	Siang	5	4	78%
	Malam	23	20	71%
11-02-2025	Siang	14	12	76%
	Malam	11	11	74%
11-03-2025	Siang	7	7	80%
	Malam	16	13	75%
11-04-2025	Siang	8	8	79%
	Malam	19	14	73%

Berdasarkan Tabel 2 mengenai Hasil Pengujian Sistem pada Video Rekaman, menunjukkan bahwa sistem mampu mendeteksi jumlah *person* sesuai dengan tingkat keramaian, baik pada kondisi sepi maupun ramai. Implementasi ambang batas (*threshold*) waktu selama 30 detik terbukti efektif dalam menyaring *false positive*, sehingga sistem hanya mengategorikan individu sebagai pengunjung valid jika berada di area dalam durasi tersebut. Tingkat akurasi deteksi skor *Avg Confidence* yang dihasilkan sistem berada pada rentang 71% hingga 80%, yang menunjukkan performa deteksi yang cukup konsisten di berbagai waktu pengujian.

Tabel 2. Hasil Perhitungan Durasi

Hari	Tanggal	Waktu	Rata-rata Durasi
Rabu	10-29-2025	Siang	53 menit
Sabtu	10-25-2025	Malam	43 menit
Sabtu	11-01-2025	Siang	1 jam
		Malam	58 menit
Minggu	11-02-2025	Siang	1 jam 15 menit
		Malam	50 menit
Senin	11-03-2025	Siang	1 jam 6 menit
		Malam	50 menit
Selasa	11-04-2025	Siang	46 menit
		Malam	50 menit

Seperti yang terlihat pada tabel 3, rata-rata durasi kunjungan pelanggan di Jarak Coffee & Eatery menunjukkan bahwa rata-rata durasi kunjungan berkisar antara 43 menit hingga 1 jam 15 menit. Durasi terlama tercatat pada hari Minggu, 2 November 2025 siang hari, dengan rata-rata 1 jam 15 menit. Data ini mengindikasikan bahwa akhir pekan cenderung lebih lama dibandingkan hari kerja. Informasi yang dihasilkan ini diharapkan dapat memberikan data objektif bagi pihak manajemen untuk mengatur strategi rotasi tempat duduk dan optimalisasi layanan pada jam-jam sibuk.

SIMPULAN

Penelitian ini berhasil mengembangkan sistem *monitoring* dan estimasi lama waktu kunjungan pelanggan menggunakan YOLOv11 dan BoT-SORT dengan performa mAP@0.5 96,5% serta *precision* dan *recall* 93,4%. Sistem dapat menjaga konsistensi ID dan menyaring *false positive* melalui *threshold*, terbukti dengan *customer valid* 103 dari 120 deteksi. *Output* berupa visualisasi dan log CSV. Data terstruktur ini dapat memberikan solusi bagi manajemen Jarak Coffee & Eatery untuk menganalisis pola kunjungan serta mengoptimalkan strategi operasional.

DAFTAR PUSTAKA

[1] I. N. Husna, M. Ulum, A. K. Saputro, Haryanto, D. T. Laksono, and D. N. Purnamasari, "Rancang Bangun Sistem Deteksi Dan Perhitungan Jumlah Orang Menggunakan Metode Convolutional Neural Network (CNN)," *SinarFe7*, vol. 5, no. 1, pp. 1–6, 2022.

[2] N. Wakhidah, P. T. Pungkasanti, and A. P. R. Pinem, "Deteksi Objek menggunakan Deep Learning untuk Mengetahui Tingkat Kerumunan Mahasiswa," *J. Edukasi dan Penelit. Inform.*, vol. 9, no. 3, p. 465, 2023, doi: 10.26418/jp.v9i3.70132.

[3] D. I. Mulyana, M. Rizki, S. Tinggi, I. Komputer, and C. Karya, "Deteksi Real-Time Durasi Kehadiran Pelanggan Di Meja Kafe Dengan Kamera Ip Dan Algoritma Deep Sort Real-Time Detection Of Customer

Duration At Cafe Tables Using Ip Cameras And Deep Sort Algorithms," *INTECOMS J. Inf. Technol. Comput. Sci.*, vol. 8, no. 5, pp. 1388–1393, 2025.

[4] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of Yolo Algorithm Developments," *Procedia Comput. Sci.*, vol. 199, pp. 1066–1073, 2021, doi: 10.1016/j.procs.2022.01.135.

[5] W. Wu and J. Lai, "Multi Camera Localization Handover Based on YOLO Object Detection Algorithm in Complex Environments," *IEEE Access*, vol. 12, no. November 2023, pp. 15236–15250, 2024, doi: 10.1109/ACCESS.2024.3357519.

[6] D. W. Permatasari and R. E. Putra, "Implementasi Algoritma YOLO11 dalam Mendeteksi Spesies Ikan Laut Komersial secara Real Time untuk Sistem Penyortiran Ikan," *J. Informatics Comput. Sci.*, vol. 6, no. 04, pp. 924–931, Feb. 2025, doi: 10.26740/jinacs.v6n04.p924-931.

[7] J. Jamil and S. Pulukadang, "Application of Deep Learning Method in Learning," *Formosa J. Sustain. Res.*, vol. 4, no. 6, pp. 1011–1028, Jun. 2025, doi: 10.55927/fjsr.v4i6.308.

[8] I. H. Sarker, "AI-Based Modeling: Techniques, Applications and Research Issues Towards Automation, Intelligent and Smart Systems," *SN Comput. Sci.*, vol. 3, no. 2, pp. 1–20, 2022, doi: 10.1007/s42979-022-01043-x.

[9] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, p. 53, Mar. 2021, doi: 10.1186/s40537-021-00444-8.

[10] A. Raup, W. Ridwan, Y. Khoeriyah, S. Supiana, and Q. Y. Zaqiah, "Deep Learning dan Penerapannya dalam Pembelajaran," *JIIP - J. Ilm. Ilmu Pendidik.*, vol. 5, no. 9, pp. 3258–3267, Sep. 2022, doi: 10.54371/jiip.v5i9.805.

[11] M. Rizki, S. Basuki, and Y. Azhar, "Implementasi Deep Learning Menggunakan Arsitektur Long Short Term Memory(LSTM) Untuk Prediksi Curah Hujan Kota Malang," *J. Repos.*, vol. 2, no. 3, Jan. 2024, doi: 10.22219/repositor.v2i3.30499.

[12] D. Pakiding, A. Selao, and W. Wahyuddin, "Implementasi Computer Vision dalam Mendeteksi Penyakit pada Tanaman Cabai dan Tomat Menggunakan Algoritma Convolutional Neural Networks," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 5, no. 3, pp. 841–850, Jun. 2025, doi: 10.57152/malcom.v5i3.1989.

[13] F. Agustina and M. Sukron, "Deteksi Kematangan Buah Pepaya Menggunakan Algoritma YOLO Berbasis Android," *J. INFOKAM*, vol. 18, no. 2, 2022.

[14] B. M. Basuki, "Deteksi klasifikasi dan menghitung kendaraan berbasis algoritma You Only Look Once (YOLO) menggunakan kamera CCTV," *Sci. Electro*, vol. 16, no. 3, 2023.

[15] D. I. Mulyana *et al.*, "Implementasi Deteksi Emosional Pada Wajah Menggunakan Deep Learning-Yolov5," *JUTECH J. Educ. Technol.*, vol. 4, no. 1, pp. 12–22, 2023.

[16] I. Andi, M. Muchtar, and J. Y. Sari, "Mask Detection Using the YOLO (You Only Look Once) Method," *J. Media Inf. Teknol.*, vol. 1, no. 1, 2024.

[17] Aharon, Nir, R. Orfaig, and B. Ben-Zion, "BoT-SORT: Robust associations multi-pedestrian tracking," *arXiv Prepr. arXiv2206.14651*, 2022.