LINK GITHUB UNTUK SOURCE PENELITIAN INI:
https://github.com/rizuabd/Tugas_Besar_Sysrem_Engineering_and_Modelling/tree/main/PeopleHeadDetectionYolov11

Pemodelan dan Simulasi Sistem Penghitung Penumpang Otomatis Pada Bus Rapid Transit (BRT) Berbasis YOLOv11 Sebagai *Cyber-Physical System*

Muhamad Rizqi Abdillah
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
23224071@mahasiswa.itb.ac.id

Idham Nurul Khaidir
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
23224317@mahasiswa.itb.ac.id

Anggera Bayuwindra
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
bayuwindra@itb.ac.id

***Abstrak*— Transportasi umum seperti Bus Rapid Transit (BRT) memiliki peran penting dalam mendukung mobilitas masyarakat perkotaan. Namun, masalah kelebihan kapasitas penumpang masih sering terjadi, terutama pada jam sibuk yang berpotensi membahayakan keselamatan. Untuk mengatasi hal tersebut, diperlukan sistem penghitung penumpang yang akurat dan efisien. Penelitian ini mengusulkan perancangan sistem penghitung penumpang otomatis berbasis algoritma object detection YOLO (*You Only Look Once*), yang mampu bekerja secara *real-time* dan dapat diimplementasikan pada perangkat *edge*. Sistem dirancang sebagai bagian dari *cyber-physical system* dan dimodelkan menggunakan pendekatan *discrete-event system* untuk menggambarkan aliran masuk dan keluar penumpang pad BRT. Implementasi dilakukan dalam bentuk simulasi berbasis Python, dengan integrasi model YOLO untuk deteksi objek serta logika penghitungan berdasarkan pergerakan penumpang dalam *Region of Interest*. Hasil dari penelitian ini merupakan model YOLOv11 yang bisa melakukan deteksi dan sistem yang bisa menghitung keluar dan masuknya penumpang pada BRT, dengan evaluasi model YOLOv11 didapatkan *mAP@0.5* sebesar 91,8%, *Precision* 91,4%, *Recall* 86,6%, dan *F1-score* 88,9%. Untuk penghitungan keluar dan masuk penumpang pada *early real-world* test didapatkan akurasi penghitungan sebesar 100% dan *inference time* pada rentang 15-45ms untuk setiap skenario yang diujikan.**

Kata Kunci—YOLO, penghitung penumpang, deteksi objek, discrete-event system.

I. PENDAHULUAN

Transportasi publik memainkan peran vital dalam sistem mobilitas perkotaan yang berkelanjutan. Salah satu moda transportasi yang banyak digunakan di berbagai kota besar adalah Bus Rapid Transit (BRT), yang menawarkan efisiensi tinggi dalam pengangkutan penumpang. Namun, sistem ini kerap menghadapi permasalahan kelebihan kapasitas (*overcrowding*), terutama pada jam sibuk.

Menurut data PT Jasa Raharja [1], kendaraan dengan dimensi dan muatan berlebih (*Over Dimension and Overload* - ODOL) menjadi penyebab kecelakaan tertinggi kedua, dengan 6.390 korban meninggal pada tahun 2024 dan 2.203 korban hingga Mei 2025. Sistem penghitungan penumpang yang ada, baik manual maupun dengan sensor tradisional, sering kali tidak efektif dan akurat saat kondisi ramai Click or tap here to enter text.. Hal ini mendorong perlunya pengembangan sistem penghitungan penumpang otomatis yang akurat dan efisien secara *real-time.* Kelebihan penumpang tidak hanya berdampak pada kenyamanan layanan, tetapi juga meningkatkan risiko keselamatan, seperti terganggunya evakuasi darurat atau cedera akibat pengereman mendadak [1].

Untuk memastikan keselamatan dan efisiensi operasional, pengelola transportasi membutuhkan sistem yang mampu melakukan pemantauan jumlah penumpang secara akurat dan *real-time*. Penghitungan manual oleh petugas tidak efisien dan rawan kesalahan, sedangkan solusi berbasis sensor tradisional (misalnya inframerah atau beban) memiliki keterbatasan dalam hal akurasi, terutama dalam situasi padat. Oleh karena itu, diperlukan pendekatan teknologi yang lebih adaptif dan cerdas.

Kemajuan dalam bidang *computer vision* dan *deep learning* telah menghasilkan berbagai algoritma deteksi objek yang andal. Salah satunya adalah YOLO [3], [4] (*You Only Look Once*) Click or tap here to enter text., yang dirancang untuk mendeteksi objek secara cepat dan efisien dalam satu

proses pemindaian gambar. model ini telah

mampu beroperasi secara *real-time* dan dapat dijalankan pada perangkat *edge* dengan daya komputasi terbatas seperti Jetson Nano atau Raspberry Pi [3], [4]. Hal ini membuka peluang besar untuk mengembangkan sistem penghitung penumpang otomatis yang bersifat mandiri, tanpa bergantung pada infrastruktur *cloud* [5].

Dalam penelitian ini, dikembangkan sistem penghitung penumpang berbasis YOLO yang dirancang sebagai bagian dari arsitektur *cyber-physical system*. Selain itu, sistem dimodelkan sebagai *discrete-event system* guna merepresentasikan dinamika aliran penumpang di dalam bus, khususnya pada titik masuk/keluar. Evaluasi dilakukan dalam bentuk simulasi menggunakan data video, dengan fokus pada tiga metrik performa utama: akurasi deteksi, kecepatan pemrosesan, dan akurasi penghitungan.

Penelitian ini mengusulkan pemodelan dan simulasi sistem penghitung penumpang otomatis berbasis algoritma YOLOv11[3]. Tantangan utama dalam penelitian ini adalah mencapai keseimbangan antara akurasi deteksi dan kecepatan pemrosesan (*inference time*). Algoritma YOLOv11 dipilih karena menawarkan *trade-off* terbaik antara akurasi dan efisiensi, serta memungkinkan implementasi secara *real-time* pada perangkat *edge* [6], [7].

Kontribusi utama dari penelitian ini meliputi:

- Perancangan arsitektur sistem penghitung penumpang otomatis berbasis YOLO.
- Pemodelan sistem sebagai discrete-event system untuk mendukung simulasi dan analisis performa.
- Evaluasi performa sistem dalam skenario simulasi, yang relevan dengan kondisi operasional BRT.

II. Tinjauan Pustaka

A. Penghitung Penumpang Konvensional

Sistem penghitung penumpang merupakan komponen penting dalam manajemen armada transportasi umum. Pendekatan tradisional banyak menggunakan perangkat keras seperti sensor inframerah, sensor tekanan, dan pintu otomatis dengan sensor penghitung. Meskipun terbukti efektif dalam beberapa kondisi, metode ini rentan terhadap kesalahan, terutama ketika penumpang bergerak secara berkelompok, membawa barang bawaan besar, atau terjadi gangguan lingkungan Click or tap here to enter text.. Selain itu, sistem ini tidak fleksibel terhadap konfigurasi kendaraan yang berbeda dan tidak menyediakan informasi visual yang dapat dimanfaatkan lebih lanjut.

Keterbatasan tersebut mendorong pengembangan sistem penghitung berbasis visi komputer yang memanfaatkan kamera dan algoritma pemrosesan citra/video.

B. Deteksi Penumpang Berbasis YOLOv11

Deteksi objek telah menjadi solusi dalam berbagai aplikasi visi komputer, termasuk penghitungan penumpang. Salah satu algoritma paling terkenal adalah YOLO (You Only Look okluOnce), yang dikembangkan pertama kali oleh Redmon et al. sebagai metode deteksi satu tahap yang efisien dan cepat [3]. YOLO memproses seluruh gambar dalam satu waktu inferensi dan langsung mengeluarkan prediksi kotak pembatas (bounding box) beserta klasifikasinya.

Perkembangan YOLO terus mengalami peningkatan signifikan, hingga versi terbaru yaitu YOLOv11, yang dirilis oleh komunitas Ultralytics dengan banyak perbaikan dari sisi arsitektur, efisiensi pemrosesan, dan ketepatan deteksi pada objek kecil maupun dalam kondisi pencahayaan rendah. YOLOv11 menawarkan:

- Arsitektur modular dan ringan, mendukung deployment di edge device seperti Jetson Nano, Raspberry Pi 5, hingga Coral TPU.
- Integrasi dengan Vision Transformer (ViT) dan mekanisme attention untuk meningkatkan kemampuan generalisasi pada latar kompleks seperti interior bus Click or tap here to enter text..
- Latency rendah dengan real-time inferensi hingga 60 FPS pada perangkat GPU ringan Click or tap here to enter text..

Keunggulan YOLOv11 menjadikannya sangat sesuai untuk implementasi sistem penghitung penumpang berbasis kamera, terutama pada lingkungan bergerak seperti bus kota atau Bus Rapid Transit (BRT).

C. Penghitung Penumpang Berbasis YOLOv11

Dalam implementasinya, YOLOv11 digunakan untuk mendeteksi individu dalam sebuah frame video. Deteksi ini kemudian dikombinasikan dengan algoritma pelacakan dalam pendekatan tracking-by-detection, di mana setiap objek yang terdeteksi dilacak lintas frame untuk mengetahui arah gerak dan identitas sementaranya. Zhang et al. Click or tap here to enter text. mengintegrasikan YOLO dengan Deep SORT (Simple Online Realtime Tracking) untuk menghitung jumlah penumpang berdasarkan arah lintasan yang melewati Region of Interest (ROI) di area spesifik pintu masuk atau keluar bus. Sistem ini mampu membedakan antara penumpang yang masuk dan keluar secara real-time dengan akurasi tinggi. Selain pelacakan berbasis Deep SORT[11], algoritma seperti ByteTrack juga semakin populer karena kemampuannya dalam mempertahankan identitas objek bahkan dalam kondisi deteksi yang tidak sempurna. Pelacakan ini sangat penting karena memungkinkan sistem untuk mengurangi false positive, misalnya ketika satu orang muncul dalam dua frame yang berbeda. Dengan memusatkan penghitungan pada ROI pintu masuk/keluar dan memanfaatkan pelacakan objek yang andal, sistem dapat meningkatkan efisiensi dan akurasi dalam proses pencacahan otomatis penumpang.

Formula penghitungan penumpang ditunjukan pada persamaan (1).

$$N(t) = N(t-1) + \sum_{i=1}^{n} \Delta_i(t) \quad (1)$$

dimana

$N(t)$: Jumlah kumulatif penumpang pada waktu t

$\Delta_i(t)$: Perubahan status masuk/keluar ke-I berdasarkan arah gerak pada *crossing line*

D. Simulasi dan Pemodelan

Untuk mendukung analisis performa sistem tanpa implementasi fisik, pendekatan Discrete-Event System (DES) digunakan. DES merupakan metode pemodelan sistem dinamis yang berubah berdasarkan event diskrit, seperti "penumpang masuk" atau "penumpang keluar". Setiap event memicu perubahan pada status sistem (state), yaitu jumlah penumpang di dalam bus. Model DES dinotasikan pada persamaan (2)

$$\mathrm{G} = (\mathrm{Q}, \Sigma, \delta, \mathrm{q}_0, \mathrm{Q}_\mathrm{m}) \quad (2)$$

dimana

Q : Himpunan state sistem (Kosong, Sebagian, Penuh)

Σ : Himpunan event diskrit (Masuk, Keluar)

δ : Fungsi transisi state

q_0 : State awal sistem

Q_m : Himpunan yang diamati

Pemodelan ini relevan untuk menguji skenario operasional seperti overkapasitas, lonjakan penumpang, atau kegagalan deteksi, melalui simulasi perangkat lunak. Pendekatan DES umum digunakan dalam bidang transportasi, manufaktur, dan sistem antrian Click or tap here to enter text.. Dalam konteks penelitian ini, DES digunakan untuk:

- Menyusun skenario alur masuk/keluar penumpang.
- Menyimulasikan pergerakan event dari hasil pelacakan YOLOv11.
- Mengevaluasi performa sistem berdasarkan akurasi dan throughput sistem penghitung.

E. Evaluasi Sistem Deteksi dan Pelacakan

Evaluasi sistem penghitung penumpang dilakukan berdasarkan tiga metrik utama:

1) Akurasi deteksi: Seberapa tepat model YOLOv11 dalam mendeteksi manusia dalam berbagai kondisi cahaya dan posisi.
2) Akurasi penghitungan: Dihitung berdasarkan selisih jumlah penumpang sebenarnya dan jumlah yang dilaporkan oleh sistem, dengan memperhatikan arah lintasan dalam ROI.
3) Kecepatan pemrosesan (frame rate): Ukuran waktu inferensi dan pemrosesan pelacakan, yang menentukan kelayakan untuk aplikasi real-time.

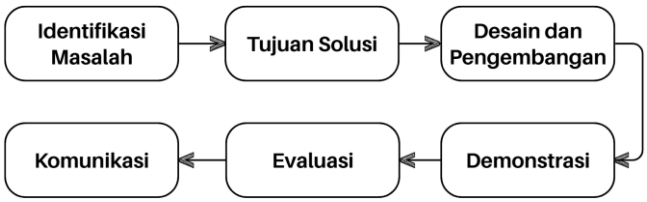
YOLOv11 telah diuji dalam berbagai benchmark dan menunjukkan performa kompetitif dengan model state-of-the-art lainnya seperti DETR dan DINO, namun dengan konsumsi daya dan sumber daya yang lebih rendah [13].

III. METODOLOGI PERANCANGAN

A. Alur Perancangan

Penelitian ini menggunakan pendekatan *Design Science Research Methodology* (DSRM).

DSRM adalah metodologi untuk menciptakan dan mengevaluasi artefak yang dirancang untuk memecahkan masalah di dunia nyata. Alur perancangan mengikuti tahapan-tahapan DSRM yang terdiri dari Identifikasi Masalah, Tujuan Solusi, Desain dan Pengembangan, Demonstrasi, Evaluasi, dan Komunikasi.

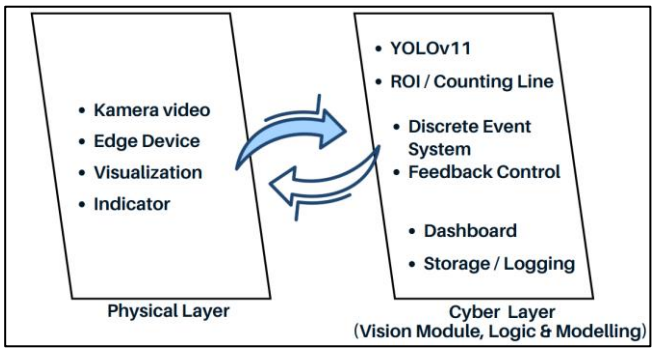


Gambar 1. Metodologi DSRM

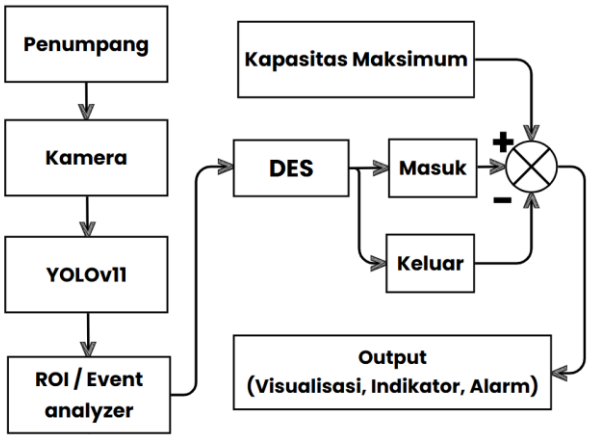
B. Desain Sistem

Arsitektur sistem dirancang secara modular dengan mengadopsi konsep *Cyber-Physical System* (CPS) [14]. Sistem ini terbagi menjadi dua lapisan utama:

1) *Physical Layer*: Terdiri dari komponen fisik seperti kamera video, perangkat pemrosesan (*Edge Device*), serta visualisasi dan indikator sebagai output.
2) *Cyber Layer*: Mencakup komponen logis dan perangkat lunak, termasuk model deteksi YOLOv11, modul *Region of Interest* (ROI) dan *Counting Line*, model *Discrete Event System* (DES), *Feedback Control*, dasbor pemantauan, serta modul penyimpanan data (*Storage/Logging*).



Gambar 2. Arsitektur Berbasis *Cyber-Physical System*

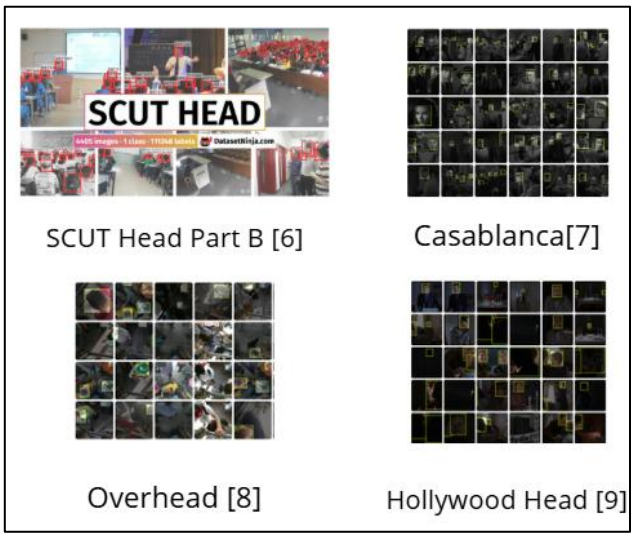


Gambar 3. Diagram Blok Sistem Penghitung Penumpang

Diagram blok sistem menggambarkan alur kerja sebagai berikut: Kamera menangkap citra, yang kemudian diproses oleh YOLOv11 untuk deteksi. *ROI/Event Analyzer* memproses hasil deteksi untuk mengidentifikasi kejadian masuk atau keluar, yang kemudian diperbarui oleh DES. Hasil akhir berupa jumlah penumpang ditampilkan melalui visualisasi, indikator, atau alarm.

C. Metode Pengujian

Pengujian sistem dilakukan dalam beberapa tahap. Pertama, model dilatih menggunakan gabungan beberapa dataset untuk merepresentasikan kondisi operasional yang beragam,



Gambar 4. Dataset yang Digunakan

Untuk *scene* yang mewakiliki atau menggambarkan kepadatan penumpang dan orientasi kepala. Dataset yang digunakan mencakup SCUT-HEAD (Part B) [10], Casablanca [8], [11], Hollywood Head [8], [12], dan Overhead People [13]. Dataset ini dibagi dengan proporsi 70% untuk data latih (*training*), 20% untuk validasi, dan 10% untuk pengujian (*testing*). Implementasi sistem ini di lapangan menghadapi tantangan khas visi komputer di lingkungan transportasi publik, seperti perubahan pencahayaan, getaran kamera, dan oklusi [7]

Pengujian performa model dari dataset, dievaluasi berdasarkan:

1) *Precision:*

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

2) *Recall:*

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

3) *F1-Score:*

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (5)$$

4) *mAP:*

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \quad ; \quad mAP \approx AP_{person} \quad (6)$$

Pengujian performa sistem di dunia nyata dievaluasi berdasarkan:

1) Akurasi Penghitungan:

$$Accuracy_{count} = 1 - \frac{|N - N_{true}|}{N_{true}} \quad (7)$$

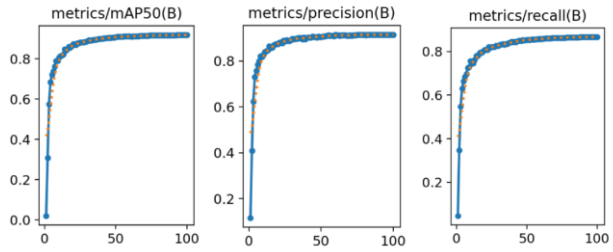
N : Jumlah prediksi
N_{true} : Jumlah aktual

2) *Frame Rate* (FPS): Mengukur jumlah *frame* yang dapat diproses per detik.

$$FPS = \frac{Jumlah\ Frame}{Total\ Waktu\ (detik)} \quad (8)$$

IV. Implementasi dan Hasil

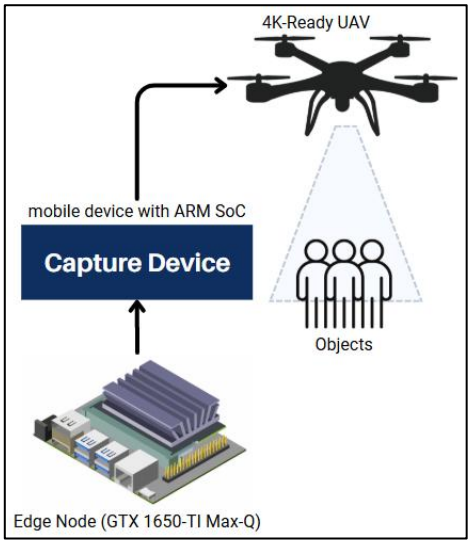
Hasil evaluasi model deteksi YOLOv11 menunjukkan performa yang sangat memuaskan.



Gambar 5. Grafik Evaluasi Model

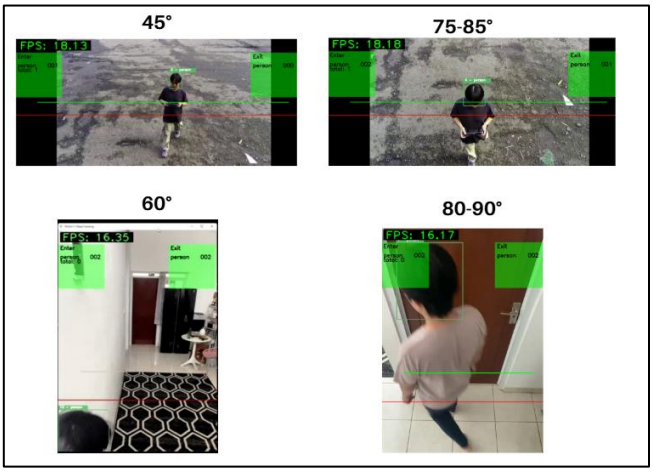
Berdasarkan pengujian, model mencapai mAP@0.5 sebesar 91,8%, Precision 91,4%, Recall 86,6%, dan F1-score 88,9%. Grafik evaluasi selama proses iterasi atau *epoch* pelatihan juga

menunjukkan bahwa nilai mAP, presisi, dan *recall* konsisten meningkat, menandakan model belajar dengan baik.



Gambar 6. Skema Pengujian *Early Real-World Test*

Pada tahap pengujian *early real-world test* yang disimulasikan, sistem diuji dengan berbagai sudut pandang kamera, dengan sudut elevasi bervariasi dari 45° hingga 90°.



Gambar 7. Pendeteksian Pada *Early Real-World Test*

Hasil dari *early real-world test*, sistem mampu mencapai akurasi penghitungan 100% di semua iterasi skenario pengujian.

Tabel 1. Hasil Pengujian *Early Real-World Test*

Iteration	**1**	**2**	**3**	**4**
POV/Elevation	60°	45°	75-85°	80-90°
Accuracy	100%	100%	100%	100%
Framerate	10-45	10-45	10-45	10-45
Inference Time	15-25ms	15-25ms	15-25ms	15-25ms

Dari segi performa komputasi pada perangkat *edge* (GTX 1650-TI Max-Q), sistem mampu berjalan pada *framerate* antara 10-45 FPS dengan *inference time* berkisar 15-25ms. Hasil ini menunjukkan bahwa sistem mampu beroperasi secara efisien dan *real-time*.

V. KESIMPULAN

Sistem penghitung penumpang otomatis berbasis YOLOv11 yang dirancang dalam penelitian ini telah berhasil didemonstrasikan melalui pemodelan dan simulasi. Sistem ini menunjukkan kemampuan yang baik dalam mendeteksi dan menghitung penumpang yang masuk dan keluar, dengan tingkat akurasi penghitungan mencapai 100% pada lingkungan pengujian yang disimulasikan.

Namun, perlu digarisbawahi bahwa pengujian sistem ini masih terbatas pada lingkungan simulasi dan belum diuji pada kondisi operasional Bus Rapid Transit (BRT) yang sebenarnya. Oleh karena itu, penelitian di masa depan perlu fokus pada uji coba di lingkungan nyata untuk mengevaluasi kinerja sistem terhadap berbagai tantangan seperti fluktuasi pencahayaan, kepadatan penumpang yang dinamis, dan getaran kamera. Secara keseluruhan, sistem ini memiliki potensi besar untuk diimplementasikan sebagai solusi teknologi guna meningkatkan keselamatan dan efisiensi operasional transportasi publik.

REFERENCES

[1] "Bersama Pemerintah dan Korlantas Polri, Jasa Raharja Siap Sukseskan Program Indonesia Menuju Zero Over Dimension and Over Load - PT Jasa Raharja." Accessed: Jun. 17, 2025. [Online]. Available: https://www.jasaraharja.co.id/id/article/69

[2] N. Cherrier, B. Rérolle, M. Graive, A. Dib, and E. Schmitt, "Context-Aware Automated Passenger Counting Data Denoising," Feb. 2024, doi: 10.1109/ITSC57777.2023.10422561.

[3] "Ultralytics YOLO11 - Ultralytics YOLO Docs." Accessed: Jun. 17, 2025. [Online]. Available: https://docs.ultralytics.com/models/yolo11/

[4] E. Humes, M. Navardi, and T. Mohsenin, "Squeezed Edge YOLO: Onboard Object Detection on Edge Devices," 2023. [Online]. Available: https://arxiv.org/abs/2312.11716

[5] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of Edge Computing and Deep Learning: A Comprehensive Survey," Jul. 2019, doi: 10.1109/COMST.2020.2970550.

[6] L. Jiao *et al.*, "A Survey of Deep Learning-based Object Detection," Jul. 2019, doi: 10.1109/ACCESS.2019.2939201.

[7] E. Dilek and M. Dener, "Computer Vision Applications in Intelligent Transportation Systems: A Survey," *Sensors*, vol. 23, no. 6, p. 2938, Mar. 2023, doi: 10.3390/s23062938.

[8] K. Sigman, "Introduction to Discrete Event Simulation." Accessed: Jun. 17, 2025. [Online]. Available: https://www.columbia.edu/~ks20/4404-16-Fall/Simulation-Discrete-Event.pdf

[9] X. Zhao, G. Lamperti, D. Ouyang, and X. Tong, "Minimal Diagnosis and Diagnosability of Discrete-Event Systems Modeled by Automata," *Complexity*, vol. 2020, no. 1, p. 4306261, 2020, doi: https://doi.org/10.1155/2020/4306261.

[10] J. Liu, Y. Zhang, J. Xie, Y. Wei, Z. Wang, and M. Niu, "Head Detection Based on DR Feature Extraction Network and Mixed Dilated Convolution Module," *Electronics (Basel)*, vol. 10, no. 13, p. 1565, Jun. 2021, doi: 10.3390/electronics10131565.

[11] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, "Multiple object tracking: A literature review," *Artif Intell*, vol. 293, p. 103448, Apr. 2021, doi: 10.1016/j.artint.2020.103448.

[12] T.-H. Vu, A. Osokin, and I. Laptev, "Context-aware CNNs for person head detection," Nov. 2015.

[13] A. Recasens, C. Vondrick, A. Khosla, and A. Torralba, "Following Gaze in Video," in *2017 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Oct. 2017, pp. 1444–1452. doi: 10.1109/ICCV.2017.160.

[14] E. A. Lee, "Cyber Physical Systems: Design Challenges," in *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, IEEE, May 2008, pp. 363–369. doi: 10.1109/ISORC.2008.25.

[15] "GitHub - HCIILAB/SCUT-HEAD-Dataset-Release: SCUT HEAD is a large-scale head detection dataset, including 4405 images labeld with 111251 heads." Accessed: Jun. 17, 2025. [Online]. Available: https://github.com/HCIILAB/SCUT-HEAD-Dataset-Release

[16] "GitHub - aosokin/cnn_head_detection: Code for Context-aware CNNs for person head detection." Accessed: Jun. 17, 2025. [Online]. Available: https://github.com/aosokin/cnn_head_detection

[17] "AI-For-Beginners/lessons/4-ComputerVision/11-ObjectDetection at main · microsoft/AI-For-Beginners · GitHub." Accessed: Jun. 17, 2025. [Online]. Available: https://github.com/microsoft/AI-For-Beginners/tree/main/lessons/4-ComputerVision/11-ObjectDetection

[18] "overhead Dataset > Overview." Accessed: Jun. 17, 2025. [Online]. Available: https://universe.roboflow.com/telkom-nyqjw/overhead-wmbfv