Pengembangan Aplikasi Deteksi Potensi Pindah Lajur Kendaraan Menggunakan YOLO V4

Franz Mikael

Informatika, Fakultas Ilmu Komputer dan Desain, Institut Teknologi dan Bisnis Kalbis
Jalan Pulomas Selatan Kav.22, Jakarta 13210
Email: franzmikael03@gmail.com

Abstract: *Driver factors need attention because they tend to cause traffic accidents. Driver factors plus the lack of safety features such as LDWS and ADAS inside vehicles in Indonesia are the reasons for this research. This study proposes a driver assistance application that can detect lane departure movement of a vehicle. If done with a camera, the application might help minimize driving negligence. YOLO v4 is selected to meet application needs such as high detection accuracy and fast computation time. The YOLO v4 model was trained using a dataset containing 400 images and obtained mAP value of 42,98% with FPS value range of 18,7 FPS to 40,5 FPS. Based on test results, the application can detect driving lanes well and give warnings as the departure movement reaches 20% or more.*
Keywords: *lane departure warning system, lane detection, object detection, YOLO v4*

Abstrak: *Faktor-faktor pengemudi perlu mendapatkan perhatian dikarenakan memiliki kecenderungan sebagai penyebab kecelakaan lalu lintas. Faktor-faktor pengemudi ditambah minimnya fitur keselamatan seperti LDWS dan ADAS dalam kendaraan bermotor di Indonesia inilah yang melatarbelakangi penelitian ini. Penelitian ini mengusulkan aplikasi bantuan pengemudi yang dapat mendeteksi potensi perpindahan lajur pada kendaraan yang apabila diintegrasikan bersama sebuah kamera dapat membantu meminimalkan kelalaian berkendara. YOLO v4 dipilih karena dapat memenuhi kebutuhan aplikasi seperti akurasi deteksi yang tinggi juga waktu komputasi yang cepat. Model YOLO v4 dilatih menggunakan dataset berisi 400 gambar dan memperoleh nilai mAP sebesar 42,98% dengan rentang nilai FPS sebesar 18,7 FPS hingga 40,5 FPS. Berdasarkan pengujian pada aplikasi, aplikasi dapat mendeteksi lajur dengan baik dan memberikan peringatan apabila perpindahan kendaraan terhadap pusat lajur mencapai 20% atau lebih.*
Kata kunci: *deteksi objek, pendeteksian jalur, sistem peringatan perpindahan jalur, YOLO v4*

I. PENDAHULUAN

Lane Departure Warning System (LDWS) adalah teknologi keselamatan bagian dari sistem bantu *Advance Driver-Assistance System* (ADAS) yang membantu pengemudi menjaga kendaraan tetap berada pada lajurnya [1]. LDWS bertujuan untuk meningkatkan keselamatan berkendara dengan memberikan peringatan kepada pengemudi apabila terjadi perpindahan lajur yang tidak diinginkan. *Lane detection* atau pendeteksian lajur merupakan bagian penting dari LDWS. *Lane detection* bertujuan untuk mendeteksi marka jalan pada lajur kendaraan sekaligus menentukan apakah kendaraan tetap berada pada lajurnya atau tidak [2]. *Lane detection* umumnya bergantung pada seperangkat sensor untuk memperoleh persepsi terhadap sekeliling kendaraan seperti lidar, radar, dan kamera di sekeliling kendaraan [3].

Berdasarkan sebuah penelitian terhadap tingkat kecelakaan lalu lintas di Indonesia pada tahun 2014, faktor-faktor pengemudi perlu mendapatkan perhatian dikarenakan memiliki kecenderungan sebagai penyebab

kecelakaan lalu lintas [4]. Faktor-faktor pengemudi ini meliputi penglihatan, pendengaran, pengalaman / kemahiran berkendara, emosi, kebiasaan, hingga rasa kelelahan. Di Indonesia, fitur keselamatan seperti LDWS dan ADAS masih jarang dimiliki oleh kendaraan bermotor seperti mobil penumpang. Fitur keselamatan ini masih tergolong baru dan jarang digunakan oleh masyarakat Indonesia. Untuk menjawab permasalahan di atas, maka dikembangkan aplikasi bantuan pengemudi yang dapat mendeteksi potensi perpindahan lajur pada kendaraan yang apabila diintegrasikan bersama sebuah kamera dapat membantu meminimalkan kelalaian berkendara.

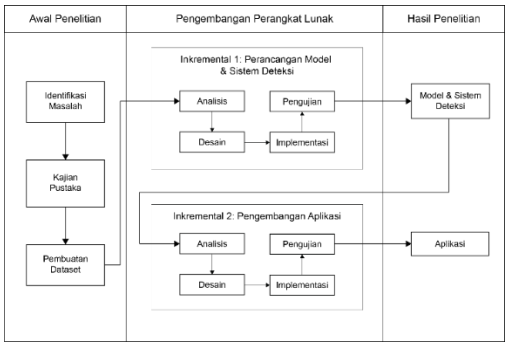
Deteksi objek dapat dilakukan menggunakan beberapa algoritma seperti *Convolutional Neural Network, Region-Based Convolutional Neural Network*, *Single Shot MultiBox Detector*, atau *You Look Only Once*. Dalam penelitian *A Fast Learning Method for Accurate and Robust Lane Detection Using Two-Stage Feature Extraction with YOLO v3* [5], Xiang, dkk mengembangkan sistem deteksi objek marka jalan menggunakan YOLO v3 dengan 2 tahap pelatihan pada 2 *dataset* berbeda. Hasilnya, YOLO v3 mampu menghasilkan *mean Average Precision* tertinggi dan waktu komputasi terendah dibandingkan YOLO v1/v2, *Faster* R-CNN, *Fast* R-CNN, *Context* + R-CNN, *Sliding Window* + CNN, dan SSD. Dalam penelitian *YOLO v4: Optimal Speed and Accuracy of Object Detection* [6], Alexey, dkk mengusulkan modifikasi pada arsitektur YOLO v3 yang kemudian disebut YOLO v4 sebagai peningkatan dari YOLO v3. Hasilnya, terdapat peningkatan mAP sebesar 10% dan *frames per second* sebesar 12% pada YOLO v4 dibandingkan YOLO v3. Dikarenakan pendeteksian lajur kendaraan digunakan secara waktu nyata maka mAP dan FPS yang dihasilkan harus sebaik mungkin.

Rumusan masalah pada penelitian ini adalah bagaimana mengembangkan aplikasi yang dapat mendeteksi potensi perpindahan lajur pada kendaraan menggunakan YOLO v4. Tujuan dari penelitian adalah untuk mengembangkan aplikasi yang dapat mendeteksi potensi perpindahan lajur pada kendaraan melalui rekaman kamera dasbor menggunakan YOLO v4.

II. METODE PENELITIAN

A. Proses Penelitian

Penelitian ini berfokus pada pengembangan aplikasi deteksi lajur dan LDWS menggunakan model *deep learning* YOLO v4. *You Only Look Once* adalah salah satu algoritma deteksi objek secara waktu nyata yang dibuat pada tahun 2015 [7]. YOLO v4 yang dikembangkan oleh Alexey Bochkovskiy, dkk [6] merupakan peningkatan dari YOLO v3 yang sebelumnya telah dikembangkan oleh Joseph Redmon, dkk [8]. Hal yang melatarbelakangi pengembangan aplikasi deteksi ini adalah diperlukannya langkah prefentif untuk meminimalkan faktor-faktor pengemudi yang berpotensi menyebabkan kecelakaan lalu lintas. Salah satu langkah prefentif ini adalah dengan mengembangkan aplikasi yang dapat mendeteksi potensi perpindahan lajur pada kendaraan dari lajur semestinya. Gambar 1 merupakan alur proses penelitian.

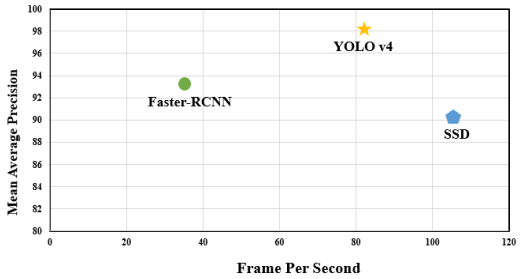


Gambar 1 Proses penelitian

Pengembangan aplikasi akan menggunakan model inkremental. Model inkremental adalah metode pengembangan perangkat lunak yang membagi pengembangan perangkat lunak menjadi beberapa bagian yang disebut inkremental [9]. Setiap tahapan inkremental memiliki input dan output [10]. Pada inkremental satu, sistem pendeteksi dikembangkan dengan menganalisis kebutuhan, perancangan desain, implementasi dan pengujian. Pada inkremental dua, *Graphical User Interface* (GUI) aplikasi dikembangkan dengan alur yang serupa dengan inkremental satu.

B. Identifikasi Masalah dan Kajian Pustaka

Faktor-faktor seperti kurangnya fokus dalam mengemudi (penglihatan, pendengaran, rasa lelah), gangguan eksternal, kebiasaan saat mengemudi, emosi hingga pengalaman dan kemahiran berkendara perlu mendapatkan perhatian dikarenakan memiliki kecenderungan sebagai penyebab kecelakaan lalu lintas [4], [11], [12]. Kebiasaan seperti penggunaan gawai saat mengemudi dinilai memperlambat waktu reaksi pengemudi, mempersulit pengemudi mempertahankan kendaraan pada lajurnya, dan menjaga jarak kendaraan dengan sekitarnya. Untuk itu, perlu dikembangkan aplikasi bantuan pengemudi yang dapat meminimalkan kelalaian dalam berkendara. Penelitian ini berfokus pada pengembangsan salah satu bentuk bantuan pengemudi, yaitu pengembangan aplikasi deteksi potensi perpindahan lajur pada kendaraan dari lajur semestinya yang umumnya disebut *Lane Departure Warning System*. Nilai mAP dan FPS akan menjadi pertimbangan utama dalam memilih algoritma deteksi objek dikarenakan LDWS nyatanya akan digunakan secara waktu nyata. *Mean average precision* (mAP) dipilih karena merupakan metrik yang populer untuk mengukur akurasi pendeteksi objek seperti YOLO [13]. Gambar 2 merupakan perbandingan performa YOLO v4 dengan beberapa algoritma lainnya.



Gambar 2 Perbandingan performa YOLO v4 dengan algoritma lain [14]

YOLO v4 memiliki akurasi paling tinggi dan nilai FPS yang cukup tinggi. Oleh karena itu, YOLO v4 dinilai paling optimal untuk digunakan dalam penelitian ini.

C. Pembuatan Dataset

Dataset training berupa kumpulan gambar jalan dengan perspektif kamera dasbor. Kumpulan gambar ini diperoleh dari potongan-potongan berbagai video kamera dasbor kendaraan di Youtube. *Dataset training* memiliki resolusi 640 x 480 berformat .jpg dengan jumlah 400 gambar. Masing-masing gambar kemudian perlu ditambahkan anotasi / label secara manual dalam format YOLO. Dalam penelitian ini, digunakan 2 kelas label, yakni “marka_kiri”

dan "marka_kanan". Proses pelabelan dilakukan dengan menambahkan *bounding box* beserta label kelas pada garis marka jalan di sisi kiri dan kanan kendaraan dalam gambar. *Bounding box* dibuat berurutan secara vertikal dengan ukuran kecil dalam jumlah banyak (Gambar 3). Pendekatan ini dipilih agar aplikasi dapat menggambarkan lekukan lajur dengan baik.

Gambar 3 Gambar dataset dengan label

Dataset testing yang akan digunakan berupa video rekaman kamera dasbor kendaraan yang didapat dari Youtube. Agar pendeteksian dapat berjalan dengan optimal maka video rekaman yang digunakan harus berlatar siang hari dan memiliki marka jalan.

D. Inkremental Satu

Tahap inkremental satu berfokus pada pengembangan model dan sistem deteksi lajur dengan menggunakan *dataset training* yang sudah dibuat. Dalam penelitian ini, YOLO v4 merupakan solusi optimal untuk pengembangan aplikasi deteksi lajur. YOLO v4 mampu memberikan akurasi deteksi yang sangat baik dalam waktu pemrosesan yang lebih singkat dibanding algoritma sejenis lainnya. Untuk mendukung proses pelatihan, digunakan platform Google Colaboratory Pro dengan akselerasi GPU. Tabel 1 merupakan perangkat keras dan perangkat lunak yang digunakan dalam penelitian ini.

Tabel 1 Instrumen Penelitian

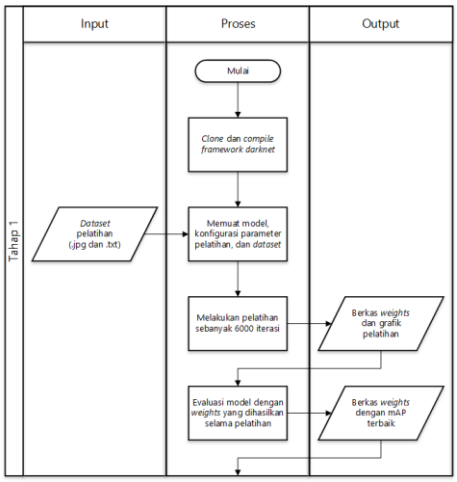
Perangkat keras	Perangkat Lunak
• Intel ® Core™ i5-8250U • NVIDIA MX130 • RAM 16 GB	• OS Windows 10 Home 64 bit • CUDA 10.2 • CUDNN 8.2.2 • Google Colaboratory Pro dengan akselerasi perangkat keras GPU NVIDIA Tesla P100 • Darknet Framework • Python 3.8.6 • Numpy 1.22.3 • OpenCV 4.5.5 • PyQt5 5.15.4

Pembuatan model pada penelitian akan menggunakan YOLO v4 dengan berkas konfigurasi *yolov4-custom.cfg* yang dipasangkan dengan *weight yolov4.conv.137* yang merupakan *pre-trained weight dataset* COCO. Proses pelatihan akan menghasilkan 1 model dengan 9 berkas *weight* berbeda. Beberapa parameter konfigurasi pelatihan yang diubah, yaitu jumlah *classes*, *batch*, *subdivisions*, *width*, *height*, *flip*, *max_batches*, *steps*, dan *filters*. Parameter *classes* dan *batch* ditetapkan seperti berikut: $classes = 2$; $batch = 64$. Parameter resolusi konvolusi (*width* dan *height*) bisa diisi dengan nilai kelipatan 32 [15]. Menyesuaikan dengan perangkat keras yang digunakan, maka resolusi konvolusi diperbesar menjadi 736 (sebelumnya 416). Nilai *subdivisions* diperbesar menjadi 64 (sebelumnya 16). Agar model dapat membedakan marka di kiri dan kanan sebagai kelas terpisah, perlu ditambahkan parameter *flip*. Parameter *width*, *height*, dan *flip* ditetapkan seperti berikut: $width = 736$; $height = 736$; $flip = 0$. Nilai *subdivisions*, *max_batches*, *steps*, dan *filters* akan

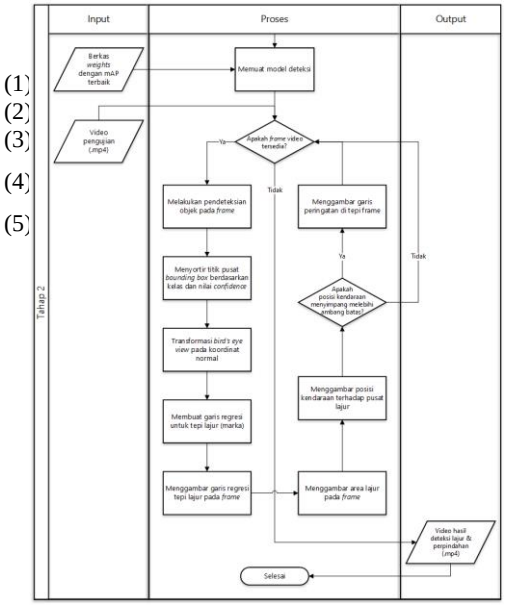
ditentukan menggunakan persamaan berikut [15]:

$$subdivisions \leq batch \quad (1)$$
$$max\_batches = classes * 2000 \quad (2)$$
$$max\_batches \geq 6000 \quad (3)$$
$$steps = 80\% * max\_batches, \quad (4)$$
$$90\% * max\_batches$$
$$filters = (classes + 5) * 3 \quad (5)$$

Parameter *subdivisions*, *max_batches*, *steps*, dan *filters* ditetapkan seperti berikut: $subdivisions = 64$; max $\_batches = 6000$; $steps = 4800{,}5400$; $filters = 21$. Gambar 4 dan 5 merupakan alur pengembangan model dan sistem.



Gambar 4 Diagram alir pengembangan sistem deteksi tahap 1



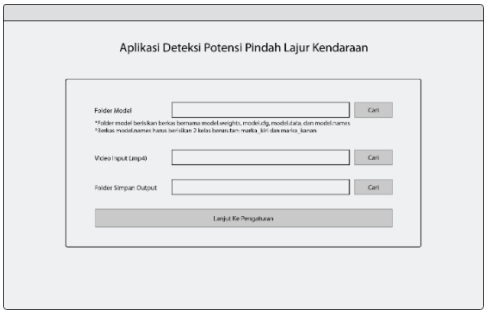
Gambar 5 Diagram alir pengembangan sistem deteksi tahap 2

Pengujian pada inkremental satu adalah pengujian terhadap kemampuan model dalam mendeteksi marka lajur. Proses pengujian dilakukan dengan mengevaluasi nilai mAP dan FPS dari kesembilan berkas *weight* hasil pelatihan. Pengujian FPS akan melibatkan beberapa resolusi konvolusi berbeda: 416, 512, 608, 736, 832.

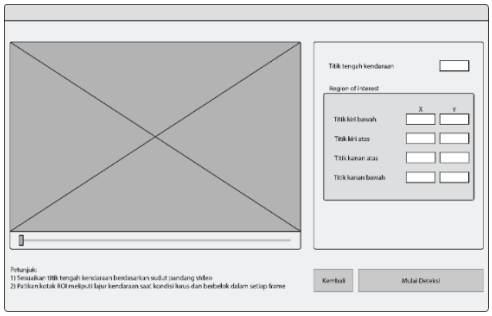
E. Inkremental Dua

Tahap inkremental dua berfokus pada pengembangan *Graphical User Interface* (GUI) aplikasi. Pembuatan antarmuka aplikasi akan menggunakan pustaka PyQt5 yang merupakan sebuah pustaka Python untuk menjalankan *framework* aplikasi Qt versi 5 [16]. Aplikasi akan terdiri dari 2 buah halaman. Halaman pertama akan berisikan *form* input untuk memilih *folder* model, video input, dan *folder* penyimpanan video output. Halaman kedua akan berisikan *form* input untuk mengatur sistem pendeteksian. Perancangan aplikasi dilakukan secara langsung pada aplikasi

QT5 Designer. *Mockup* desain antarmuka aplikasi dapat dilihat pada Gambar 5 dan 6.



Gambar 5 Mockup antarmuka aplikasi (1)



Gambar 6 Mockup antarmuka aplikasi (2)

Aplikasi pendeteksian yang telah dibangun kemudian akan diuji dengan metode *black box*. Pengujian *Black Box* adalah metode pengujian perangkat lunak yang didasarkan pada tampilan aplikasi, fungsi aplikasi, dan alur dari aplikasi [17]. Pengujian dilakukan terhadap antarmuka semua tombol dan kolom input untuk mengetahui fungsionalitas komponen tersebut tanpa melihat kode program.

III. HASIL DAN PEMBAHASAN

A. Hasil Inkremental Satu

1. Model Deteksi

Model deteksi YOLO v4 hasil pelatihan mendapatkan nilai mAP antara 33,29% sampai 42,98%.

Tabel 2 Nilai mAP Model YOLO v4

Jumlah Iterasi	*mAP@0.5*
1000	*33,29%*
2000	*39,29%*
3000	*37,23%*
4000	*33,82%*
5000	*33,36%*
6000	*35,75%*
Best	*42,98%*
Last	*35,75%*
Final	*35,75%*

Tabel 2 merupakan nilai mAP dari setiap berkas *weights* yang dihasilkan selama pelatihan. Berkas *best*.weights dengan akurasi 42,98% dipilih untuk diimplementasikan ke dalam aplikasi.

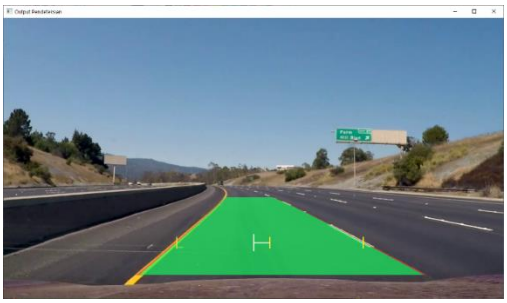
Tabel 3 Nilai FPS Model YOLO v4

Resolusi Konvolusi	*FPS*
416x416	*40,5*
512x512	*34,8*
608x608	*28,3*
736x736	*22,9*
832x832	*18,7*

Tabel 3 merupakan nilai FPS dari berkas *best*.weights yang diujikan pada beberapa resolusi konvolusi pada sebuah video. Pengujian dilakukan di platform Google Colaboratory Pro dengan akselerasi GPU NVIDIA Tesla P100. Hasilnya, resolusi 416x416 mampu menghasilkan FPS tertinggi, yakni 40,5 *frames per second* dan 832x832 menghasilkan FPS terendah, yakni 18,7 *frames per second*.

2. Sistem Deteksi

Sistem deteksi yang dihasilkan pada inkremental satu akan menampilkan komponen visual pada video yang diproses, yaitu garis tepi (berwarna merah) dan area lajur (berwarna hijau); *bar* posisi kendaraan (berwarna putih) dan lajur (berwarna kuning); dan peringatan perpindahan lajur (berwarna kuning atau merah). Komponen visual tersebut dapat dilihat pada Gambar 6, 7, dan 8.



Gambar 6 Hasil pemrosesan video oleh sistem deteksi

Gambar 7 Peringatan pertama perpindahan lajur

Gambar 8 Peringatan kedua perpindahan lajur

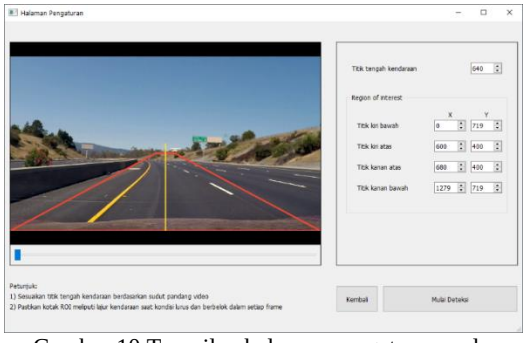
Garis kedua tepi lajur dihasilkan dengan membuat garis regresi polinomial derajat 2 pada marka kiri dan kanan lajur yang terdeteksi oleh model. Area lajur digambarkan sebagai poligon berwarna hijau yang menghubungkan garis kiri dengan garis kanan lajur. Besaran perpindahan diukur dari jarak antara *bar* posisi kendaraan terhadap *bar* pusat lajur. Apabila perpindahan mencapai 20% maka peringatan pertama berupa bingkai berwarna kuning akan digambarkan. Apabila perpindahan mencapai 30% maka peringatan kedua berupa bingkai berwarna merah akan digambarkan.

B. Hasil Inkremental Dua

Hasil dari inkremental dua adalah aplikasi deteksi lajur. Gambar 9 merupakan tampilan awal aplikasi.

Gambar 9 Tampilan halaman awal pada aplikasi

Di halaman awal, dibuat 3 buah kolom input beserta tombol “Cari” untuk memilih lokasi *folder* model, video input, dan *folder* penyimpanan output. Apabila tombol “Cari” ditekan, *file dialog* akan muncul untuk memilih *folder* atau berkas terkait.



Gambar 10 Tampilan halaman pengaturan pada aplikasi

Gambar 10 merupakan tampilan halaman pengaturan aplikasi. Pengaturan meliputi pengaturan posisi kendaraan dan area yang akan dideteksi. Keduanya memiliki nilai *default* dan dapat diatur lebih lanjut untuk mendapatkan hasil pendeteksian yang optimal. Komponen

slider di bawah bingkai video berguna membantu pengguna menyesuaikan ROI pada setiap *frame* video input.

IV. SIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan, maka diperoleh kesimpulan sebagai berikut:

1. Pada penelitian ini, penerapan YOLO v4 untuk mengembangkan aplikasi deteksi potensi perpindahan lajur pada kendaraan berhasil dilakukan. Model YOLO v4 memperoleh nilai mAP sebesar 42,98% dengan rentang nilai FPS sebesar 18,7 FPS hingga 40,5 FPS. Model YOLO v4 mampu membedakan marka kiri dan marka kanan sebagai 2 kelas objek yang berbeda.
2. Kemampuan perangkat keras menjadi aspek penting untuk mendukung akurasi pendeteksian YOLO v4 serta waktu pemrosesannya. Semakin besar resolusi konvolusi yang digunakan, semakin kecil FPS yang diperoleh selama proses pendeteksian. Begitu juga sebaliknya.
3. Dalam sistem LDWS yang dibangun, terdapat 2 buah peringatan: peringatan pertama (berwarna kuning), menandakan perpindahan yang terjadi berskala ringan dan peringatan kedua (berwarna merah), menandakan perpindahan yang terjadi berskala berat.
4. Peringatan pertama akan ditampilkan apabila kendaraan bergeser sebesar 20% atau lebih dari pusat lajur. Sementara peringatan kedua akan ditampilkan apabila kendaraan bergeser sebesar 30% atau lebih dari pusat lajur.

DAFTAR RUJUKAN

[1] M. Papis, T. Dziewo, and M. Matyjewski, "Preliminary Assessment of the Advanced Driver Assistance," *Autobusy. Tech. Eksploat. Syst. Transp.*, vol. 6, no. June, pp. 371–375, 2017.

[2] W. Li, F. Qu, Y. Wang, L. Wang, and Y. Chen, "A robust lane detection method based on hyperbolic model," *Soft Comput.*, vol. 23, no. 19, pp. 9161–9174, 2019, doi: 10.1007/s00500-018-3607-x.

[3] I. Y. Kim, K. S. Yang, J. J. Baek, and S. H. Hwang, "Development of intelligent electric vehicle for study of unmanned autonomous driving algorithm," *World Electr. Veh. J.*, vol. 6, no. 1, pp. 135–140, 2013, doi: 10.3390/wevj6010135.

[4] H. Herawati, "Karakteristik Dan Penyebab Kecelakaan Lalu Lintas Di Indonesia Tahun 2012," *War. Penelit. Perhub.*, vol. 26, no. 3, p. 133, 2014, doi: 10.25104/warlit.v26i3.875.

[5] X. Zhang, W. Yang, X. Tang, and J. Liu, "A fast learning method for accurate and robust lane detection using two-stage feature extraction with YOLO v3," *Sensors (Switzerland)*, vol. 18, no. 12, 2018, doi: 10.3390/s18124308.

[6] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," 2020, [Online]. Available: http://arxiv.org/abs/2004.10934.

[7] H. Ampadu, "Yolov3 and Yolov4 in Object Detection," *Https://Ai-Pool.Com/a/S/Yolov3-and-Yolov4-in-Object-Detection*, 2021. https://ai-pool.com/a/s/yolov3-and-yolov4-in-object-detection (accessed Nov. 13, 2021).

[8] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018, [Online]. Available: http://arxiv.org/abs/1804.02767.

[9] Roger S. Pressman; Bruce R. Maxim, *Software engineering : a practitioner's approach / Roger S. Pressman, Ph.D. — Eighth edition.* 2014.

[10] M. Syarif and W. Nugraha, "Metode Incremental Dalam Membangun Aplikasi Identifikasi Gaya Belajar Untuk Meningkatkan Hasil Belajar Siswa," *Jusikom J. Sist. Komput. Musirawas*, vol. 4, no. 1, pp. 42–49, 2019, doi: 10.32767/jusikom.v4i1.441.

[11] National Highway Traffic Safety Administration, "Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey," 2015.

[12] World Health Organization, "Road Traffic Injuries," 2021. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries (accessed Mar. 03, 2022).

[13] J. Hui, "mAP (mean Average Precision) for

Object Detection," 2018. https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173 (accessed Aug. 03, 2022).

[14] J. A. Kim, J. Y. Sung, and S. H. Park, "Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition," *2020 IEEE Int. Conf. Consum. Electron. - Asia, ICCE-Asia 2020*, pp. 8–11, 2020, doi: 10.1109/ICCE-Asia49877.2020.9277040.

[15] A. Bochkovskiy, "YOLOv4 / Scaled-YOLOv4 / YOLO - Neural Networks for Object Detection (Windows and Linux version of Darknet)." https://github.com/AlexeyAB/darknet.

[16] "What is PyQt?" https://riverbankcomputing.com/software/pyqt (accessed May 27, 2022).

[17] A. O. Nurshanty, A. Saputra, F. R. Hardjanto, M. B. Franklyn, and D. Yudanegara, "Teknik Dalam White-box dan Black-box Testing." https://socs.binus.ac.id/2020/07/02/teknik-dalam-white-box-dan-black-box-testing/ (accessed May 27, 2022).