IMPLEMENTASI YOLO (*YOU ONLY LOOK ONCE*) *ALGORITHM* UNTUK PENDETEKSIAN JARAK KENDARAAN PADA *SPEED LIMITER* SEPEDA LISTRIK SULISTIO

TUGAS AKHIR

Karya Tulis sebagai Salah Satu Syarat untuk Memperoleh Gelar Sarjana Teknik dari Universitas Singaperbangsa Karawang

oleh:

Resi Sujiwo Bijokangko

NPM: 1810631160119

PROGRAM STUDI TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS SINGAPERBANGSA KARAWANG

2022

IMPLEMENTASI *YOLO (YOU ONLY LOOK ONCE) ALGORITHM* UNTUK PENDETEKSIAN JARAK KENDARAAN SEBAGAI *SPEED LIMITER* SEPEDA LISTRIK SULISTIO

Resi Sujiwo Bijokangko*

Program Studi Teknik Elektro

Fakultas Teknik Universitas Singaperbangsa Karawang 2022

Pembimbin; Dian Budhi Santoso, S.T., M.Eng. dan Ibrahim, S.T., M.T.**

ABSTRAK

Sebagai upaya dalam mengurangi resiko kecelakaan yang dapat dialami pengendara maka keselamatan berkendara merupakan hal esensial yang memerlukan perhatian oleh setiap pengendara. Pada penelitian ini penulis melakukan perancangan dan implementasi sistem *speed limiter* sebagai fungsi pembatas kecepatan sepeda listrik SULISTIO dengan prediksi jarak terhadap objek mobil, sepeda motor, serta pejalan kaki di area depan sepeda untuk mengurangi resiko kecelakaan oleh pengemudi. Citra hasil tangkapan modul kamera kemudian diolah pada Raspberry Pi dengan program *speed limiter* untuk mendeteksi objek di depan sepeda listrik dan memprediksi jaraknya terhadap sepeda. Ketika objek yang dideteksi memiliki jarak kurang dari 3 meter maka sistem akan mengaktifkan Relay yang terhubung pada pin GPIO 4 Raspberry Pi untuk memutus jakur tegangan pada *throtle*. Berdasarkan hasil pengujian sistem, sistem ini dapat mendeteksi objek dengan kelas manusia, motor, dan mobil dengan baik dengan tingkat F1-Score sebesar 0,914926, IoU sebesar 66,14%, dan nilai mAP pada treshold 0,5 sebesar 97,58%. Hasil model prediksi jarak pada sistem *speed limiter* sangatlah baik dengan nilai akurasi 97,31% pada kelas Manusia, 94,92% pada kelas Motor, serta 97,97% pada kelas Mobil dengan akurasi keseluruhan kelas sebesar 97%.

Kata Kunci: resiko kecelakaan, speed limiter, deteksi objek, prediksi jarak

Alamat Sekarang:

* Program Studi Teknik Elektro, Fakultas Teknik Universitas Singaperbangsa Karawang, Jawa Barat. resi.sujiwo18119@student.unsika.ac.id

** Program Studi Teknik Elektro, Fakultas Teknik Universitas Singaperbangsa Karawang, Jawa Barat. dian.budhi@ft.unsika.ac.id

IMPLEMENTASI *YOLO (YOU ONLY LOOK ONCE) ALGORITHM* UNTUK PENDETEKSIAN JARAK KENDARAAN SEBAGAI *SPEED LIMITER* SEPEDA LISTRIK SULISTIO

Resi Sujiwo Bijokangko*

Program Studi Teknik Elektro

Fakultas Teknik Universitas Singaperbangsa Karawang 2022

Pembimbin; Dian Budhi Santoso, S.T., M.Eng. dan Ibrahim Lammada, S.T., M.T.**

ABSTRACT

As an effort to reduce the risk of traffic accidents that can be experienced by the driver, driving safety is an essential thing that requires attention by every driver. In this study, the authors design and implement a speed limiter system as a function of speed limiting SULISTIO electric bicycles by predicting the distance to objects from cars, motorcycles, and pedestrians in the front area of the bicycle to reduce the risk of accidents by drivers. The captured image of the camera module is then processed on the Raspberry Pi with a speed limiter program to detect objects in front of the electric bicycle and their distance from the bicycle. When the detected object is less than 3 meters away, the system will activate the Relay which is connected to the Raspberry Pi's GPIO 4 pin to disconnect the voltage on the throttle. Based on the results of system testing, this system can detect objects with human, motorcycle, and car classes well with an F1-Score level of 0.914926, an IoU of 66.14%, and a mAP value at a threshold of 0.5 of 97.58%. . The results of the distance prediction model on the speed limiter system are very good with an accuracy value of 97.31% in the Human class, 94.92% in the Motorcycle class, and 97.97% in the Car class with an overall class accuracy of 97%..

Keywords: traffic accident risk, speed limiter, object detection, distance prediction

Current Address: * *Electrical of Engineering Department, Faculty of Engineering, University of Singaperbangsa Karawang, West Java. resi.sujiwo18119@student.unsika.ac.id*

** *Electrical of Engineering Department, Faculty of Engineering, University of Singaperbangsa Karawang, West Java. dian.budhi@ft.unsika.ac.id*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR ISTILAH DAN SINGKATAN

STS	*Smart Transportation System*
YOLO	*You Only Look Once*
NC	*Normally Close*
NO	*Normally Open*
IoU	*Intersection Over Union*
mAP	*mean Average Precision*
CNN	*Convolutional Neural Network*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kemajuan teknologi ikut mendorong perubahan pola transportasi manusia yang dahulunya bergantung pada moda transportasi tradisional seperti berjalan kaki atau dengan berkendara menaiki hewan tertentu menjadi bentuk transportasi yang lebih modern menggunakan mesin dengan bahan bakar fosil atau listrik. Seiring dengan berkembangnya teknologi transportasi membuat kendaraan yang digunakan dalam kegiatan tersebut menjadi semakin tinggi kecepatannya. Kecepatan yang tinggi dapat meningkatkan resiko kecelakaan dan resiko dampak kecelakaan yang dialami oleh pengendara.

Sebagai upaya mengurangi resiko kecelakaan yang dapat dialami pengendara maka keselamatan berkendara merupakan hal esensial yang memerlukan perhatian lebih dari setiap pengendara kendaraan bermotor. Kurangnya kehati-hatian dari pengguna kendaraan bermotor dan buruknya medan menjadi dua penyebab utama terjadinya kecelakaan lalu lintas. Dalam UU No. 22 Tahun 2009 tentang Lalu Lintas dan Angkutan Jalan, disebutkan bahwa adalah Kecelakaan Lalu Lintas adalah suatu peristiwa di Jalan yang tidak diduga dan tidak disengaja melibatkan Kendaraan dengan atau tanpa Pengguna Jalan lain yang mengakibatkan korban manusia dan/atau kerugian harta benda [1].

Kecelakaan lalu lintas tidak hanya menimbulkan korban jiwa namun juga kerugian materil bagi negara dan keluarga korban kecelakaan. Menurut data Korlantas Polri pada tahun 2020, sejak Januari hingga Oktober 2020 terdapat 83.715 kejadian kematian atau cedera di jalan akibat kecelakaan lalu lintas yang berdampak pada kerugian materil hingga Rp 163,3 miliar [2]. Kecelakaan yang paling sering terjadi khususnya pada kendaraan roda dua diakibatkan oleh kelalaian manusia seperti kecepatan berlebih dan kurangnya fokus pengendara.

Sebagai upaya mengurangi resiko kecelakaan dalam berkendara pada penelitian ini penulis melakukan perancangan dan implementasi sistem *speed limiter* sebagai fungsi pembatas kecepatan sepeda listrik SULISTIO dengan pengukuran jarak dengan kendaraan mobil, sepeda motor, serta pejalan kaki di area depan sepeda. Sepeda SULISTIO memiliki motor penggerak berupa Motor DC 36V dengan daya 500W. Sepeda SULISTIO memiliki kecepatan maksimum hingga 30 km/jam pada beban 60 Kg. Dengan rangka bodi yang ringan, ketika terjadi kecelakaan momentum yang dimiliki oleh pengendara lebih besar dibanding sepeda SULISTIO sendiri yang dapat berakibat pada sulitnya pengendara untuk mengendalikan sepeda dan kerusakan pada sepeda akan lebih besar.

Pengukuran jarak kendaraan mobil, sepeda, sepeda motor, serta pejalan kaki di area depan sepeda dilakukan dengan menggunakan metode pengolahan citra

dengan metode *Convolutional Neural Network (CNN)* menggunakan Python dan library OpenCV. Klasifikasi objek dilakukan dengan menggunakan algoritma YOLO sebagai penyedia *pretreined model*. Citra yang diolah merupakan citra hasil pembacaan oleh modul kamera Raspberry Pi 4B yang terletak pada box alat *Smart Transportation System* di bagian atas stang sepeda listrik SULISTIO.

Pada sepeda listrik SULISTIO sistem *speed limiter* dapat diakses dengan mudah pada aplikasi GUI *Smart Transportation System Dashboard*. Aplikasi tersebut merupakan antarmuka sistem *Smart Transportation System* dengan pengguna sepeda listrik SULISTIO. Pada layanan ini sistem *speed limiter* dapat diintegrasikan dengan layanan livestream sehingga pengguna dapat menjalankan siaaran video langsung seiringan dengan jalannya sistem *speed limiter* sepeda listrik SULISTIO.

1.2 Identifikasi Masalah

Berdasarkan latar belakang yang sudah penulis paparkan dapat diidentifikasi masalahnya yaitu:

1. Pembuatan layanan pembatas kecepatan berbasis deteksi objek untuk mengurangi resiko kecelakaan lalu lintas
2. Antarmuka sistem *Speed limiter* berbasis deteksi objek dengan sistem *Smart Transpotation System* sepeda listrik SULISTIO

1.3 Rumusan Masalah

Berdasarkan uraian yang telah dipaparkan penelitian ini memiliki beberapa perumusan masalah sebagai berikut:

1. Bagaimana cara mengetahui jarak kendaraan melalui pengolahan citra kamera Raspberry menggunakan Algoritma YOLO?
2. Bagaimana cara melakukan antarmuka sistem pengukuran jarak dengan sepeda listrik SULISTIO dalam sistem *speed limiter*?
3. Bagaimana performa sistem deteksi objek dan prediksi jarak pada sistem *speed limiter* sepeda listrik SULISTIO?
4. Bagaimana performa sistem kontrol relay sebagai pembatas kecepatan sepeda listrik?

1.4 Batasan Masalah

Diantara batasan masalah yang terdapat pada Tugas Akhir ini adalah sebagai berikut:

1. Objek yang terbaca pada sistem deteksi dibatasi hanya pada kelas objek mobil, sepeda motor, serta pejalan kaki.
2. Sistem pembatas kecepatan mengukur jarak objek dengan sepeda melalui nilai keliling *bounding box* objek.

1.5 Tujuan Penelitian

Dalam menyelesaikan penelitian dengan judul "Implementasi *YOLO (You Only Look Once) Algorithm* untuk pendeteksian jarak kendaraan sebagai *speed limiter* sepeda listrik SULISTIO" penulis telah menetapkan tujuan penelitian yang daintaranya adalah:

1. Merancang dan membuat sistem deteksi objek dan prediksi jarak objek menggunakan pengolahan citra kamera Raspberry dengan algoritma YOLO secara *real time.*
2. Melakukan antarmuka sistem pada Raspberry Pi dengan sepeda listrik sebagai pembatas kecepatan laju motor.
3. Mengetahui performa kerja sistem deteksi objek dan prediksi jarak pada sistem *speed limiter* sepeda listrik SULISTIO.
4. Mengetahui performa kerja kontrol relay sebagai pembatas kecepatan sepeda listrik.

1.6 Manfaat Penelitian

Manfaat dari penelitian ini adalah sebagai berikut:

1. Terciptanya sistem sepeda listrik dengan sistem keamanan yang lebih baik dibanding sistem kemanan pada sepeda listrik selanjutnya melalui fitur *speed limiter* pada sepeda listrik.
2. Pengembangan sistem keamanan pada sepeda listrik dengan teknologi yang lebih modern menggunakan Kecerdasan Buatan.

1.7 Sistematika Penulisan

Laporan Tugas Akhir ini disusun dengan sistematika penulisan sebagai berikut:

Bab I Pendahuluan

Pendahuluan berisi tentang latar belakang penelitian, identifikasi masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian serta sistematika penulisan laporan Tugas Akhir.

Bab II Dasar Teori

Dasar teori berisi tentang kajian pustaka dari teori penunjang yang membahas tentang *Convolutional Neural Netwok (CNN)*, *YOLO Algorithm*, kajian ringkas Raspberry Pi 4, serta performa model *Computer Vision*, serta State of The Art Penelitian.

Bab III Metode Penelitian

Dalam metode penelitian dibahas mengenai metode yang digunakan dalam penyelesaian penelitian, perancangan perangkat keras dan perancangan perangkat lunak sistem, serta pendalaman mengenai implementasi algoritma YOLO dalam deteksi objek.

Bab IV Hasil Pengujian dan Analisis

Hasil pengujian dan analisis berisi tentang hasil yang didapat dari kalibrasi sistem prediksi jarak, performa sistem prediksi jarak, performa sistem deteksi objek serta performa kerja sistem kontrol relay sebagai pembatas kecepatan sepeda listrik.

Bab V Kesimpulan dan Saran

Kesimpulan dan saran berisi tentang kesimpulan yang didapat dalam hasil pengujian dan analisis pada alat penelitian dan saran yang diperlukan dalam pengembangan alat *speed limiter* bagi penelitian yang selanjutnya.

BAB II
TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

2.1.1 Kendaraan

Menurut Undang-Undang Nomor 22 Tahun 2009 tentang Lalu Lintas dan Angkutan Jalan kendaraan didefinisikan sebagai suatu sarana angkut di jalan yang terdiri atas kendaraan bermotor dan kendaraan tidak bermotor. Menurut Undang Undang tersebut kendaraan bermotoradalah setiap kendaraan yang digerakkan oleh peralatan mekanik berupa mesin selain kendaraan yang berjalan di atas rel, sedangkan kendaraan tidak bermotor adalah setiap kendaraan yang digerakkan oleh tenaga manusia dan/atau hewan [3].

2.1.2 Sepeda Listrik SULISTIO

Sepeda listrik adalah merupakan sepeda yang dapat dijalankan dengan 2 cara yaitu dengan mengayuh pedalnya atau dengan menggunakan motor listrik sebagai tenaga penggeraknya. Belakangan ini popularitas sepeda listrik mulai meningkat yang berdampak pada penjualan sepeda yang semakin meningkat tiap tahunnya. Pada tahun 2020 saja, Dalam situs resmi Melotronic (2020) dijelaskan bahwa kenaikan pembelian sepeda listrik pada tahun 2020 naik hingga 3 kali lipat dibandingkan tahun sebelumya [4].

Sepeda listrik SULISTIO merupakan sepeda listrik buatan tim penulis yang dibuat sebagai produk dari Tugas Akhir. Spesifikasi dari sepeda listrik SULISTIO secara detail dapat dilihat pada Tabel 1. Penelitian ini menggunakan sepeda listrik SULISTIO sebagai objek pemasangan fitur *speed limiter* yang dihasilkan pada penelitian ini.

Tabel 1. Spesifikasi Sepeda Listrik SULISTIO

No	Spesifikasi	Keterangan
1	Tegangan Motor DC	36V
2	Arus Maksimum Motor DC	18,7 Amphere
3	Daya Motor DC	500W
4	Tipe Motor Driver	36V 500W
5	Input Charger	220VAC 0,82A
6	Output Charger	36VDC 5A
7	Jarak Tempuh	±20km
8	Kecepatan Maksimum (Beban 80 kg)	30 km/h
9	Beban Maksimum	120 kg
10	Suhu Operasi	-20°C – 80°C
11	Kontroler *Smart System*	Arduino Nano, Arduino UNO, Raspberry Pi 3B

12	Standar Produk	SNI 8613:2018 ISO 13063:2012

2.1.3 Raspberry Pi

Raspberry Pi adalah komputer mini *single board* yang dikembangkan oleh Raspberry Pi Foundation. Pada awalnya Raspberry Pi dikembangkan oleh Pete Lomas, Jack Lang, dan Alam Mycrof di Laboratorium Komputer *Cambridge Univeristy*. Raspberry Pi sudah dilengkapi dengan *CPU, GPU, RAM, Port USB, Audio Jack, Ethernet, HDMI,* dan *GPIO* untuk masukan sensornya. Sedangkan untuk sumber dayanya Raspberry Pi membutuhkan sumber tegangan sebesar 5V dengan arus minimal 700mA.

Gambar 2. 1 Raspberry Pi 4B

(Sumber: Digiwarestore.com)

Raspbeery Pi memiliki dua buah model utama pada tiap generasinya yaitu model A dan model B. Raspberry Pi 4 model A memiliki memori hingga 1024 MB, sedangkan untuk Raspbery Pi model B memiliki memori sebesar 1024-8192 MB. Pada penelitian ini model Raspberry yang digunakan adalah model 4B. Spesifikasi lengkap pada Raspberry Pi 4B dapat dilihat pada Tabel 2.

Tabel 2. Spesifikasi Raspberry Pi 4B

No	Spesifikasi	Keterangan

1	Prosesor	Quad core Cortex-A72 (ARM v8) 64-bit SoC
2	Memori	4GB RAM
3	Wireless	2.4GHz and 5.0GHz IEEE 802.11b/g/n/ac wireless LAN, Bluetooth 5.0, BLE.
4	Jumlah Pin GPIO	40 pin
5	Port USB	2 port USB2, 2 port USB3
6	Audio	Output stereo 4 kutub dan port video komposit
7	Koneksi Display	2x Micro HDMI ports supporting up to 4K 60Hz video resolution, 2-lane MIPI DSI/CSI ports for camera and display
8	Penyimpanan Data	Port Micro SD sampai 256 GB
9	Daya Input	USB Tipe-C 5,1 V 3A

Sumber: raspberrypi.com

2.1.4 Relay

Relay adalah komponen elektronika yang berupa saklar atau switch elektrik yang dioperasikan menggunakan listrik [5]. Relay dikendalikan menggunakan coil untuk mengubah koneksi terminal COM dari terminal NC ke NO dan sebaliknya.

Gambar 2. 2 Relay

(Sumber: shopee.com)

2.1.5 *Convolutional Neural Network*

2.1.5.1 Deteksi Objek

Deteksi objek merupakan sebuah pendekatan komputasional yang ditujukan untuk mendateksi sebuah objek dalah suatu citra gambar. Deteksi objek sudah sangat berkembang dewasa ini seiring dengan meningkatnya tren kecerdasan buatan dan *Machine Learning*. Pada atikelnya, Jalled dan Voronkov (2016) membagi pengertian deteksi objek menjadi dua jenis yaitu *soft detection* yang mana hanya mendeteksi sebuah objek dan menetakan keberadaannya dan juga *hard detection* yang mampu mendeteksi keberdaan dan lokasi objek sekaligus [6].

Pada penelitian ini citra yang dijadikan bahan untuk deteksi objek merupakan citra video. Pada video deteksi objek dilakukan dengan melakukan pembacaan pada masing-masing frame video sehingga pada dasarnya tetap melakukan proses deteksi objek pada citra gambar. Proses deteksi objek ini ditujukan untuk mencari jarak objek di depan sepeda listrik SULISTIO untuk kemudian melakukan aktivasi relay dan mematikan sistem *throtle* sepeda listrik jika objek di depan sepeda listrik terlalu dekat.

2.1.5.2 OpenCV

OpenCV adalah library untuk keperluan computer vision dan machine learning yang tersedia secara *open-source*. OpenCV memiliki lebih dari 2500 algotima. OpenCV dapat digunakan dengan antarmuka program C++, Python, Java dan MATLAB dan dapat digunakan pada sistem operasi Windows, Linux, Android dan Mac OS. OpenCV dibangun untuk menyediakan infras tur umum untuk aplikasi visi komputer dan untuk mempercepat penggunaan persepsi mesin dalam produk komersial. Menjadi produk berlisensi BSD, OpenCV memudahkan bisnis untuk memanfaatkan dan memodifikasi kode [7].

2.1.5.3 You Only Look Once

You Only Look Once (YOLO) merupakan sistem deteksi objek yang pertama kali dikembangkan pada tahun 2015. Pada awal pengembangannya YOLO pertama kali muncul pada sebuah paper tentang computer vision dengan judul "*You Only Look Once: Unified, Real-Time Object Detection*" oleh Joseph Redmon [8]. Dalam papernya, Redmon (2015) menjelaskan bahwa YOLO bekerja dengan membingkai ulang deteksi objek sebagai masalah regresi tunggal, langsung dari piksel gambar ke koordinat kotak pembatas dan probabilitas kelas. Model terpadu ini memprediksi secara bersamaan beberapa kotak pembatas dan probabilitas kelas untuk objek yang dicakup oleh kotak. Pada saat diluncurkan, algoritma YOLO telah menghasilkan spesifikasi yang mengesankan yang melampaui algoritme utama dalam hal kecepatan dan akurasi untuk mendeteksi dan menentukan koordinat objek [9].

Pada website resminya, YOLO dikatakan dapat bekeerja dengan mendeteksi objek 1000 kali lebih cepat dibandingkan R-CNN dan 100 kali lebih cepat jika dibandingkan dengan Fast R-CNN. Hingga saat ini YOLO sudah berkembang sampai generasi kelimanya yaitu YOLOv5. Semenjak awal kehadirannya pada YOLOv1 algoritma YOLO telah berkembang jauh dengan benyaknya penambahan-penambahan kelas dan efisiensi algoritma. Sebagai perbandingan, evolusi YOLO sejak awal kehadirannya sampai ke generasi ketiganya dapat dilihat pada Tabel 3.

Tabel 3 Perkembangan YOLOv1 Hingga YOLOv4

Generasi	Kelebihan dari generasi sebelumnya	Jenis evolusi dari generasi sebelumnya	Keterangan
YOLOv1	Generasi Pertama		
YOLOv2	Waktu training model lebih cepat dan generalisasi jaringan meningkat; Resolusi gambar yang diolah lebih tinggi; Prediksi lebih akurat;	Penambahan Normalisasi Batch	Bertujuan untuk melakukan normalisasi luaran tiap layer deteksi ke **zero-mean** dengan standar deviasi 1.
		Klasifikasi Resolusi Tinggi	Resolusi gambar yang dapat di proses hingga 448x448 pixel dengan penambahan 4%mAP
		Konvolusi dengan *Anchor Box*	Penggunaan metode Intersection Over Union antar *bounding box* and pusatnya dalam melakukan deteksi objek dan mennetukan *anchor box* terbaiknya
YOLOV3	Penggunaan skala deteksi yang dapat diatur oleh pengguna	Jaringan konvolusi yang lebih besar dengan ResNet	Penggunaan jaringan konvolusi hybryd dengan arsitektur YOLOv2, Darknet-53 (53 layer konvolusi) dan Residual network (ResNet)
		Deteksi Multi-scale	Pendeteksian dengan 3 skala yang berbeda pada layer ke 82, 94, dan 106.
YOLOV4	peningkatan presisi *mean Average Prediction (mAP)* sebanyak 10% dan jumlah frame per detik sebesar 12%	Arsitektur Yolov4 memiliki 4 blok berbeda	*The backbone* adalah arsitektur ekstraksi fitur yang merupakan CSPDarknet53, Bagian *neck* membantu menambahkan lapisan

			antara *The backbone* dan *dense prediction block(head), The head(Dense prediction)* digunakan untuk menemukan kotak pembatas dan untuk klasifikasi.

Pada penelitian ini YOLO digunakan sebagai algoritma CNN yang diolah dengan bantuan OpenCV untuk melakukan ekstraksi algoritma dan penggunaan modelnya. Objek yang akan dideteksi menggunakan *pre-trained* model yang dimiliki oleh YOLO dapat dilihat pada Lampiran A. Penelitian ini menggunakan model YOLOv4-tiny dengan file weight dan config yang digunakan adalah file YOLOv4-tiny. YOLOV4-tiny merupakan hasil kompres dari YOLOv4 yang memberikan waktiu proses yang lebih cepat dibandingkan YOLOv4 aslinya. Pemilihan model ini bertujuan untuk mendapatkan nilai proses yang lebih rendah dibandingnkan versi YOLOV4 yang aslinya agar dapat menjalankan sistem secara *realtime*.

2.1.6 *Microsoft Excel*

Microsoft Excel merupakan program aplikasi lembar kerja *spreadsheet* yang dibuat dan didistribusikan oleh Microsoft Corporation untuk sistem operasi Microsoft Windows dan Mac OS [10]. Hingga saat ini *Microsoft Excel* telah memiliki banyak versi pada produknya untuk sistem operasi Wndows dan Mac OS. Jenis-jenis ekstensi file dari *Microsoft Excel* beserta versi *Microsoft Excel* yang terkait dapat dilihat pada Tabel 4.

Tabel 4. Ekstensi file dan versi *Microsoft Excel*-nya

Ekstensi File	Versi *Microsoft Excel*
.xls	Format dasar sebelum *Excel* 12
.xlt	Format *template worksheet Microsoft Excel* sebelum *Excel* 12
.XML	Format *XML Spreadsheet*
.xla	Format *Excel Add-in* sebelum Excel 12
.xlsx	Format dasar *worksheet Microsoft Excel* 12
.xlsm	Format *worksheet Microsoft Excel* 12
.xslb	Format *worksheet Microsoft Excel* 12
.xltm	Format *template worksheet Microsoft Excel* 12

.xlam	Format untuk *Excel Add-in* untuk menambah kemampuan *Excel* 12.

Pada penelitian ini *Microsoft Excel* digunakan sebagai aplikasi pengolahan data kalibrasi jarak dan data pengujian. Data kalibrasi jarak diolah dengan menggunakan tool Trendline pada excel. Trendline merupakan tool pada excel yang digunakan untuk melakukan analisis kurva pada suatu grafik di *Microsoft Excel*. Trendline digunakan untuk mencari persamaan garis dan nilai R^2 dari plot kurva keliling *bounding box* terhadap jarak sebenarnya pada hasil kalibrasi jarak.

2.1.7 Recall

Recall merupakan rasio dari total keseluruhan sampel positif yang diklasifikasikan dengan benar kemudian dibagi dengan total sampel positif. Kondisi *High recall* menunjukan bahwa kelas objek dideteksi dengan benar (*False Negative* sedikit). Perhitungan nilai Recall dapat dilakukan menggunkaan persamaan 2.1.

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \qquad (2.1)$$

Dimana :

TP : *True Positive*

FN : *False Negative*

2.1.8 Precision

Nilai presisi merupakan tingkat presisi dari suatu algoritma deteksi objek terhadap hasil deteksi objeknya. Nilai presisi dihitung dengan cara membagi total sampel positif yang diklasifikasikan dengan benar dengan total sampel positif yang diprediksi seperti pada persamaan. Pada bukunya Umar (2020) menjelaskan bahwa jika nilai Presisi tinggi menunjukan contoh berkelas positif memang positif (False Positive sedikit) [11]. Nilai presisi dapat dikalkulasi dengan menggunakan persamaan 2.2.

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \qquad (2.2)$$

Dimana :

TP : *True Positive*

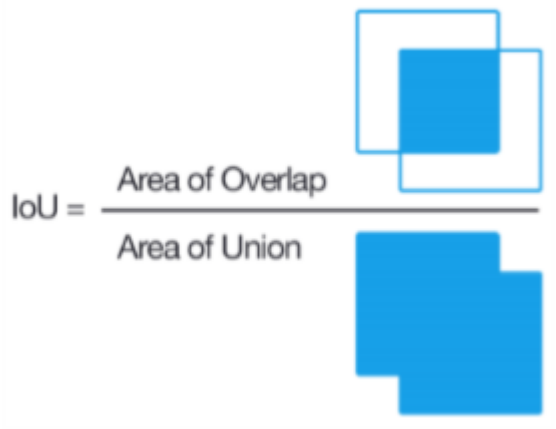
FN : *False Negative*

2.1.9 *Intersection Over Union* (IoU)

Intersection over Union (IoU) adalah metric evaluasi untuk mengukur keakuratan detektor objek pada dataset tertentu [12]. Intersection over Union (IoU)

merupakan perbandingan antara ground-truth *bounding box* dengan prediksi *bounding box* seperti pada persamaan (2.3) ilustrasi persamaan IoU dapat dilihat pada Gambar 2.3 [11].

$$IoU = \frac{A \cap B}{A \cup B} \qquad (2.3)$$



Gambar 2. 3. ***Intersection Over UnionF-Measure***

F-*Measure* bertujuan untuk menghitung kombinasi dari presisi dan recall. F-Measure akan menggunakan harmonic mean dari presisi dan recall. Nilai F-Measure dihitung menggunakan persamaan 2.4 [11].

$$F - Measure = \frac{2 \cdot recall \cdot precision}{recall + precision} \qquad (2.4)$$

2.1.10 Mean Average Precision (mAP)

Nilai *mean Average Precision (mAP)* merupakan nilai rata-rata dari *average precision*. *Average precision* merupakan nilai yang didapatkan dari setiap nilai precision item relevan yang dihasilkan dan menggunakan nilai 0 untuk item relevan yang tidak dihasilkan oleh sistem. Nilai precision untuk *average precision* dihitung dengan memperhatikan urutan item yang diberikan oleh sistem, sehingga nilai precision diberikan untuk setiap item yang dihasilkan oleh sistem [13]. Kalkulasi mAP dapat dilakukan menggunakan persamaan (2.5).

$$mAP = \frac{1}{|Q|}\sum_{j=1}^{|Q|}\frac{1}{m_j}\sum_{k=1}^{m_j} Precision(R_{jk}) \qquad (2.5)$$

Dimana :

Q : *jumlah query* pengujian atau jumlah objek pada citra pengujian
R : *Relevance Item* atau objek yang benar pada deteksi dan prediksi
m : Jumlah *Relevance Item*

2.2 *State of The Art*

Penelitian dengan menggunakan algoritma YOLO sebagai media deteksi objek telah beberapa kali dilakukan sebelumnya dengan tujuan dan implementasi yang beragam. Untuk memberikan gambaran terkait penggunaan algoritma YOLO pada penelitian sebelumnya penulis telah merangkumnya pada tabel *State of The Art* pada Tabel 5.

Tabel 5. *State of The Art*

No	Nama Penulis	Judul Penelitian	Metode	Ringkasan
1	A. Asni B, Amin, Mayda Waruni K (2021)	Penerapan Metode Yolo Object Detection V1 Terhadap Proses Pendeteksian Jenis Kendaraan Di Parkiran	YOLO Object Detection dengan YOLOV4	Model memiliki akurasi hingga 98,66% . Dengan sampel sebanyak 15 buah didapati total jumlah kendaraan terdeteksi sebanyak 64 objek dari total 66 objek. rata-rata waktu komputasi antar frame adalah 3,067 detik.
2	Jupiyandi, Fadhil Rizqullah Saniputra, Yoga Pratama, Muhammad Robby Dharmawan, Imam Cholissodin (2019)	Pengembangan Deteksi Citra Mobil Untuk Mengetahui Jumlah Tempat Parkir Menggunakan Cuda Dan Modified Yolo	Menggunakan pemprograman GPU dengan *Modified Yolo (M-Yolo)*	Penelitian bertujuan untuk mengetahui jumlah slot parkir yang masih tersedia. Hasil pengujian sistem pada 15 sampel memiliki waktu komputasi yang cepat yaitu 0,179 detik dengan rata-rata akurasi sebesar 100%.
3	waddah Harahap, Juni Elfrida, Pasrah Agusman, Mario Rafael, Rahul Abram, Kiki Andrianto (2019)	Sistem Cerdas Pemantauan Arus Lalu Lintas Dengan YOLO (You Only Look Once v3)	Pengolahan citra CCTV ATS Dishub Kota Medan dengan metode YOLOV4	Model yang digunakan mampu menlakukan klasifikasi kendaraan dengan *mAP (mean Average Precision)* pada CCTV ATS Fix dengan skor tertinggii hingga 97% dan skor tertinggi sebesar 99% untuk CCTV PTZ.
4	Calvin Geraldy, Chairisni Lubis (2020)	Pendeteksian Dan Pengenalan Jenis Mobil	Deteksi mobil dengan YOLOv2	Pengujian sistem menghasilkan tingkat keberhasilan sistem sebesar 88,1% dengan

		Menggunakan Algoritma You Only Look Once Dan Convolutional Neural Network		learning rate 0.000005 dan epoch 100.
5	Aldhiyatika Amwin (2021)	Deteksi Dan Klasifikasi Kendaraan Berbasis Algoritma You Only Look Once (Yolo)	Deteksi dan klasifikasi kendaraan dengan YOLOv2	Dari dataset sebanyak 513 gambar yang terbagi enjadi 5 kelas yaitu mobil, bus, sepeda motor, dan becak didapati model yang dpat mengenali objek pada hasil rekaman video CCTV di Simpang Air Mancur-Imanuel Kota Medan dengan *pre-trained YOLO wights* didapati nilai *mean Average Precision (mAP)* sebesar 99,35%.
6	Resi Sujiwo Bijokangko (2022)	Implementasi Yolo (You Only Look Once) Algorithm Untuk Pendeteksian Jarak Kendaraan Pada *Speed limiter* Sepeda Listrik SULISTIO	Penerapan algoritma YOLOV4 pada sistem *speed limiter* sepeda listrik SULISTIO menggunakan Raspberry Pi 4	Sistem dapat mendeteksi objek dengan kelas manusia, motor, dan mobil dengan baik dengan tingkat F1-Score sebesar 0,914926, IoU sebesar 66,14%, dan nilai mAP pada treshold 0,5 sebesar 97,58%. Hasil model prediksi jarak pada sistem *speed limiter* sangatlah baik dengan nilai akurasi 97,31% pada kelas Manusia, 94,92% pada kelas Motor, serta 97,97% pada kelas Mobil dengan akurasi keseluruhan kelas sebesar 97%.

BAB III
METODE PENELITIAN

3.1 Langkah-langkah Penelitian

Pada bagian ini dilakukan langkah-langkah penelitian digambarkan pada Gambar 3.1 yang terdiri dari studi literatur, perencanaan, perancangan, pengumpulan data, implementasi, dan pengujian.

Gambar 3. 1 Langkah-langkah Penelitian

3.2 Uraian Penelitian

3.2.1 Studi Literatur

Tahap studi literatur merupakan tahap awal dalama penelitian ini. Stuti literatur dilakukan dengan terlebih dahulu melakukan pengamatan terhadap masalah yang akan diteliti untuk kemudian dilakukan tinjauan literatur yang berkaitan dengan masalah tersebut. Pada penelitian ini studi literatur dilakukan dengan mencari literatur kepustakaan terkait pendeteksian objek melalui citra gambar atau video dengan algoritma YOLO. Pencarian literatur pada penelitian ini dilakukan dengan mengunjungi situs Google Scholar, Portal Jurnal Nasional dan Internasional, serta pada buku-buku yang ditemui oleh penulis.

Untuk mempermudah dan mempersempit pencarian sumber kajian literatur maka penulis menetapkan kriteria yang perlu diperhatikan dalam memilih literatur, diantaranya adalah sebagai berikut:

a Literatur merupakan artikel atau kajian publikasi ilmiah.
b Literatur disusun dengan menggunakan bahasa Indonesia dan Inggris.
c Literatur memiliki bahasan tentang deteksi dan klasifikasi mobil, motor, sepeda, sepeda motor, dan orang.
d Literatur menggunakan metode *Convolutional Neural Network (CNN)* dalam pengolahan citranya.
e Literatur menggunakan gambar atau video sebagai bahan deteksinya.
f Literatur yang menggunakan algoritma YOLO untuk pengolahan citranya sangatlah diutamakan.

3.2.2 Perencanaan

Perencanaan berfokus pada pemilihan software dan hardware yang digunakan untuk implementasi program deteksi objek pada layanan *speed limiter* di sepeda listrik SULISTIO. Dalam melakukan pengembangan dan implementasi program *speed limiter*, layanan tersebut dibangun pada aplikasi Visual Studio Code dengan interpreter Python 3.10.2 pada laptop pengguna, sedangkan pada

implementasi program kedalam sistem sepeda listrik SULISTIO perangkat yang digunakan adalah Raspberry Pi 4 Model B sebagai kontroler utama sistem cerdas SULISTIO. Spesifikasi perangkat keras dan perangkat lunak yang digunakan terdapat pada Tabel 6.

Tabel 6. Spesifikasi Perangkat Keras dan Perangkat Lunak

Keterangan	Spesifikasi	Pengembangan Program (Laptop ASUS X455YA)	Implementasi Program (Raspberry Pi 4 Model B)
Perangkat Keras	Prosesor	AMD A8 Quad Core	Quad core Cortex-A72 (ARM v8) 64-bit SoC
	Clock Speed	2.2 GHz	1.5GHz
	RAM	6 GB	4 GB
	VGA	Radeon Graphics	-
	Daya Input	19V	5V
Perangkat Lunak	Sistem Operasi (OS)	Windows 10 64 bit	Raspbian
	IDE Program	Visual Studio Code	Nano
	Python Interpreter	Python 3.10.0	Python 3.10.0

3.2.3 Perancangan

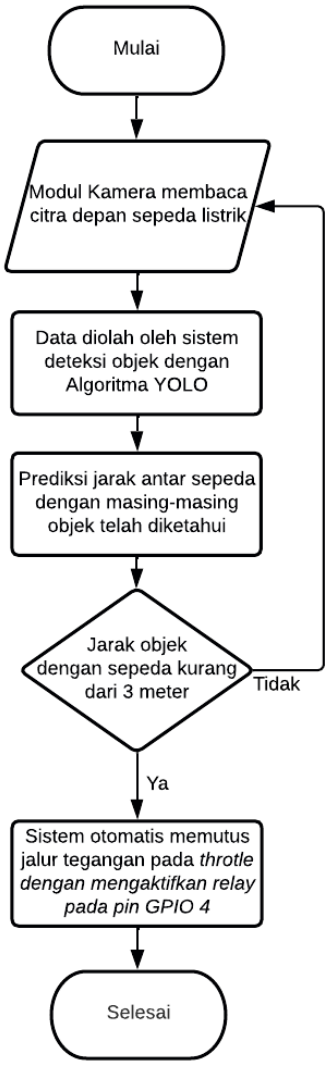
3.2.3.1 Perancangan *Hardware*

Antarmuka *hardware* sistem *Speed limiter* pada sepeda listrik SULISTIO adalah dengan menambahkan komponen Relay yangdigunakan sebagai penutus tegangan pada *throtle*. Pendekatan seperti ini dilakukan agar keseluruhan sistem tetap dapat aktif tanpa saling mengganggu sistem lainnya meskipun pengguna tidak dapat lagi menambah kecepatan sepedanya.

Gambar 3. 2 Diagram Blok Perancangan *Hardware* Sistem

3.2.3.2 Perancangan *Software*

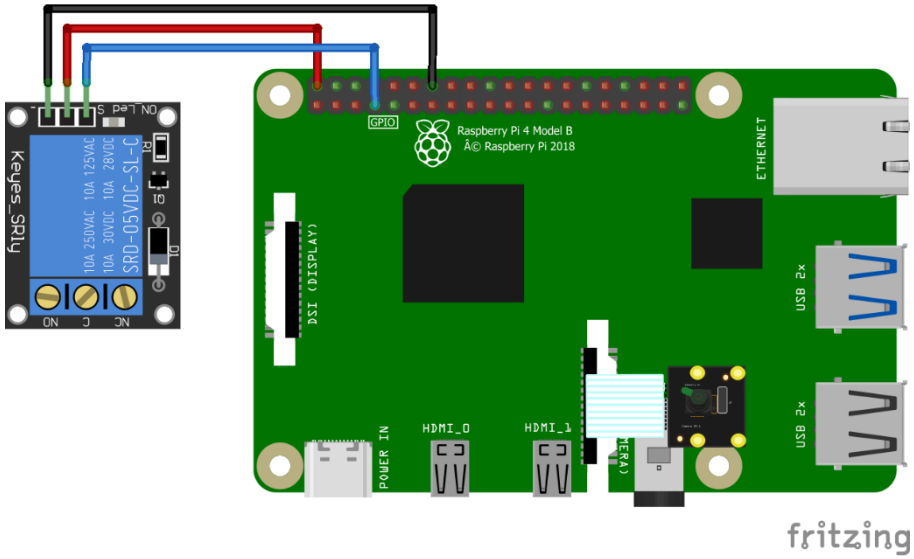
Perancangan *software* pada penelitian disajikan dalam bentuk diagram alir. Diagram alir perancangan sodtware menjelaskan tahapan kerja dari implementasi sistem *speed limiter* pada sepeda lisrik SULISTIO.



Gambar 3. 3 Diagram Alir Perancangan *Software* Sistem

Sistem dimulai dengan aktivasi sistem dan modul kamera mulai membaca citra depan sepeda listrik. Citra hasil tangkapan modul kamera kemudian diolah pada Raspberry Pi dengan program *speed limiter* untuk mendeteksi objek depan sepeda listrik dan jaraknya terhadap sepeda. Ketika objek yag dideteksi memiliki jarak kurang dari 3 meter maka sistem akan mengaktifkan Relay yang terhubung pada pin GPIO 4 Raspberry Pi untuk memutus jakur tegangan pada *throtle*.

3.2.3.3 Perancangan Elektronika



Gambar 3. 4 Perancangan Elektronika

Skematik rangkaian interkoneksi antar Raspberry Pi 4B, Relay dan Modul kamera Raspberry Pi dapat dilihat pada gambar 3.3. Pin IN Relay terhubung langsung pada pin GPIO 4 Raspberry Pi dengan sumber tegangan 5V dan GND terhubung langsung pada pin output 5V dan GND Raspberry Pi. Sedangkan modul kamera dipasang pada port kamera Raspberry Pi.

3.2.4 Pengumpulan Data

Data yang digunakan dalam penelitian ini merupakan data yang penulis dapatkan dari video hasil tangkapan kamera Raspberry Pi pada sepeda listrik SULISTIO. Pemilihan data ini beralasan bahwa program deteksi objek ini nantinya akan diimplementasikan pada layanan *speed limiter* SULISTIO dengan citra hasil tangkapan kamera Raspberry Pi yang terletak di bagian depan sepeda listrik. Data citra yang diolah tidaklah dilakukan perubahan baik pada ukuran maupun domain warna citra. Hal in ditujukan agar proses deteksi dapat berjalan dengan lebih baik dan akurat. Ukuran citra hasil tangkapan kamera Raspberry Pi adalah 1920 x 1080.

Data yang digunakan pada penelitian ini dibagi menjadi dua jenis denmgan tujuan yang berbeda. Data yang pertama merupakan data yang digunakan untuk melakukan kalibrasi jarak. Data ini merupakan data gambar dari individual objek sesuai kelas yang ingin dilakukan kalibrasi jarak pada tiap kelas objek. Data yang kedua adalah data pengujian yang mana merupakan data video hasil rekaman video kamera Raspberry Pi sepeda listrik SULISTIO yang di uji jalan pada jalan di Kelurahan Telukjambe Timur, Kabupaten Karawang untuk kemudian dipecah menjadi antar frame dengan jeda antar frame sebesar 1 detik. Jalan yang digunakan dalam pengambilan data adalah Jalan Adiarsa, Jalan Babakan Sananga, dan Jalan Galuh Mas.

a. Data Kalibrasi Jarak

Data kalibrasi jarak dikelompokkan berdasarkan kelas objeknya. Data yang diambil memiliki jumlah 10 citra pada tiap setengah meternya dengan ketentuan penggunaan 70% data akan digunakan sebagai pemodelan model matematis kalibrasi prediksi jarak dan 30% data akan digunakan sebagai citra test untuk mengetahui akurasi model. Data yang diambil sesuai kelasnya secara detail dibagi menjadi seperti berikut:

1. Mobil

Citra hanya berisikan mobil dengan jarak yang divariasi mulai dari 1,5 meter hingga 5 meter. Contoh gambar kalibrasi jarak pada mobil dapat dilihat pada gambar 3.5.

(a) (b)

Gambar 3. 5 kalibrasi jarak pada objek mobil dengan jarak (a) 5 meter, (b) 3 meter

2. Sepeda Motor

Pada kalibrasi jarak dengan objek sepeda motor terdapat dua buah sepeda motor yang ditempatkan berdekatan dengan variasi jarak 1,5 meter hgingga 5 meter pada masing-masing sepeda motor. Contoh gambar kalibrasi jarak pada objek sepeda motor dapat dilihat pada gambar 3.5.

(a) (b)

Gambar 3. 6 kalibrasi jarak pada objek sepeda dengan dengan jarak kedua objek sama pada (a) 3 meter, (b) 1,5 meter

3. Manusia

Kalibrasi orang menggunakan objek satu orang dalam satu foto dengan variasi jarak antar objek mulai dari 1,5 meter hingga 5 meter. Contoh gambar kalibrasi jarak pada objek orang dapat dilihat pada gambar 3.7.

(a) (b)

Gambar 3. 7 Kalibrasi jarak pada objek orang dengan dengan jarak masing-masing objek berbeda (a) 3 meter, (b) 5 meter

b. Data Pengujian Sistem Deteksi Objek

Data yang digunakan untuk pengujian didapatkan dengan melakukan pemecahan frame video menjadi frame baru dengan selisih antar frame adalah 0,5 detik. Video yang digunakan dibagi menjadi tigas sampel pada jalan yang berbeda yaitu:

1. Jalan Galuh Mas

Pengambilan video dilakukan mulai dari jalan depan pintu masuk Mall Karawang Central Park melawati bundaran perumnas hingga kembali lagi ke arah Masjid Puri Teluk Jambe. Reakaman video berdurasi 50 detik dengan total frame sebanyak 50 frame. Contoh frame yang didapat pada jalan Galuh Mas dapat dilihat pada gambar 3.8.

Gambar 3. 8 Sampel pengujian pada Jalan Galuh Mas

2. Jalan Babakan sananga

Pengambilan sampel di Jalan adiarsa dilakuan dari depan Gang Pelangi sampai jembatan Adiarsa dengan durasi rekaman 50 detik dan total frame sebanyak 50 frame. Contoh frame yang didapat pada Jalan Babakan Sananga dapat dilihat pada gambar 3.9.

Gambar 3. 9 Sampel pengujian pada Jalan Babakan Sananga

3. Jalan Adiarsa

Pengambilan sampel dilakukan pada jalan depan Alfamart Adiarsa hingga Bundar Adiarsa. Total frame yang dihasilkan adalah sebanyak 50 frame. Contoh frame yang didapat pada jalan Jalan Adiarsa dapat dilihat pada gambar 3.10.

Gambar 3. 10 Sampel pengujian pada Jalan Adiarsa

3.2.5 Pengolahan Data

Pengolahan data dilakukan sesuai dengan jenis data yang akan diolah. Data kalibrasi jarak dolah dengan menggunakan tool *Trendline* pada aplikasi *Microsoft Excel* untuk menentukan persamaan garis pada kurva hubungan antara keliling *bounding box* objek deteksi terhadap jarak sebenarnya. Persamaan dengan nilai R squared yang paling mendekati nilai 1 merupakan persamaan yang akan dipilih sebagai model matematis kalkulasi jarak pada sistem deteksi objek. Data pengujian akan diolah dan dijalankan langsung pada program *speed limiter* untuk mencari nilai *IoU* dan *mAP*-nya.

3.2.6 Analisis Data

Citra hasil pengujian kemudian akan dianalisis untuk menentukan nilai *Intersection over Union (IoU)* dan *mean Average Prediction (mAP)* serta nilai *confidence*-nya untuk kemudian mengetahui performa sistem deteksi objeknya. Pada sistem prediksi jarak analisis hasil pengujian akan berfokus pada akurasi hasil prediksi jarak dengan mencari nilai kesalahan pengukuran jarak oleh sistem dengan jarak yang sebenarnya.

BAB IV
HASIL DAN PEMBAHASAN

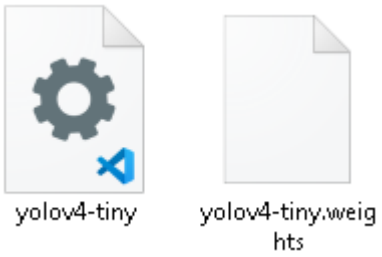
4.1 Implementasi

4.1.1 Instalasi Software

Software IDE yang digunakan dalam pembangunan sistem *speed limiter* adalah Visual Studio Code dengan Interpreter Python 3.10.9. Pengembangan sistem *speed limiter* membuatuhkan library dan file weight serta file configuration sebagai model Convolutional Neural Network (CNN).

a. Instalasi Python 3.10.9
b. Instalasi *Library*
c. Pengunduhan *file weight* dan *file configuration* yolov4-tiny

Sebagai model algoritma dan pretrained model pada operasi *Convolutional Neural Network (CNN)* digunakan file weight dan configuration pada YOLOv4 versi Tiny-nya. Versi ini dipilih dengan alasan bahwa penggunaan versi tiny dari algoritma YOLO dapat meringankan proses operasi pada deteksi citra dengan hasil deteksi objek yang cukup baik.



Gambar 4 1 File Weight dan Configuration YOLOv4 -tiny

4.1.2 Deteksi objek dengan Algoritma YOLO dan *speed limiter*

a. Import Library

```
import cv2
import numpy as np
import time
import RPi.GPIO as GPIO
```

Pada proses import Library sistem membutuhkan tiga buah library yang masing-masingnya berfungsi sebagai beikut :

1. Cv2, merupakan library OpenCV yang digunakan sebagai library pengolahan citra dan operasi *Convolutional Neural Network (CNN)*.
2. Numpy, merupakan library yang digunakan untuk
3. Time, merupakan library yang digunakan sebagai penghitung waktu operasi antar citra dan menentukan nilai FPS.
4. RPi.GPIO, merupakan library yang digunakan sebagai antarmuka penggunaan pin GPIO pada Raspberry Pi

b. Memuat Network YOLOv.4 Tiny

```
net = cv2.dnn.readNet("yolov4-tiny.weights",
                      "yolov4-tiny.cfg")
classes = []
with open("coco.names", "r") as f:
     classes = [line.strip() for line in
                f.readlines()]

layer_names = net.getLayerNames()
outputlayers = [layer_names[i-1] for i in
                net.getUnconnectedOutLayers()]

colors = np.random.uniform(0, 255,
                       size=(len(classes), 3))
```

Network yang digunakan pada penelitian ini diimport dengan menggunakan method cv2.dnn.readNet() menggunakan library OpenCV.

c. Membaca Masukan Video Pada Kamera Raspberry Pi

```
cap = cv2.VideoCapture(0)
```

Pembacaan citra video dari modul kamera Raspberry Pi dilakukan dengan menggunakaan method VideoCapture() dengan memberikan input "0" sebagai alamat kamera pada Raspberry Pi.

d. Mengambil Fungsi Blob Pada Frame

```
blob = cv2.dnn.blobFromImage( frame, 0.00392,
(210, 210), (0, 0, 0), True, crop=False)

net.setInput(blob)
```

Mengambil properti blob pada citra dengan menggunakan method blobFromImage() dan memasukkan hasilnya menjadi masukan pada network.

e. Menentukan *Confident Score* Pada Algoritma Dari Hasil Deteksi Objek

```
for detection in out:
      scores = detection[5:]
      class_id = np.argmax(scores)
      confidence = scores[class_id]
```

Pengambilan nilai confidence didapat dengan melakukan itersi deteksi pada hasil deteksi objek dan mengambil nilai *confidence*-nya pada masing-masing kelas.

f. Mengambil Nilai Koordinat Pusat dan Atas Kiri *Bounding box* Deteksi

```
center_x = int(detection[0]*width)
center_y = int(detection[1]*height)
```

```
w = int(detection[2]*width)
h = int(detection[3]*height)
```

Nilai koordinbat pusat didapat dengan mengambil indeks array ke-0 dan ke-1 pada hasil deteksi, sedangkan untuk tinggi dan lebar *bounding box* didapat dari indkes array ke-2 dan ke-3 dari hasil deteksi.

g. Menggambar *Bounding box* Objek

```
cv2.rectangle(frame, (x, y), (x+w, y+h), color, 2)
```

Penggambaran *bounding box* dilakukan dengan menggunakan method rectangel() dengan mengiusi nilai koordinat atas kiri kotak pada koordinat (x,y) dan koordinat pojok kiri bawah kotak pada koordinat(x+w, y+h) dengan tebal border 2.

h. Prediksi Jarak Masing-Masing Objek

```
kel = (2*w+2*h)/1000
if label == 'person':
    dis = 24.789*(kel*kel*kel*kel) - 251.72 *
          (kel * kel*kel) + 974.15*(kel*kel)-1815.6
          * kel + 1635.9
elif label == "car":
    dis = 6.2407*(kel*kel*kel*kel) -84.866*
          (kel*kel*kel) + 458.18 * (kel*kel)-1250.1
          * kel + 1667.6
elif label == "motorbike":
    dis = -32.051*(kel*kel*kel*kel) + 196.72 *
          (kel * kel*kel) - 313.26*(kel*kel)-147.13
          * kel + 672.53
dis = round(dis/10)
# Ubah satuan jarak menjadi Meter
dis = dis/10
```

Prediksi jarak dilakukan dengan memasukkan nilai keliling *bounding box* ke persamaan yang didapat dari hasil kalibrasi jarak pada Bab 4. Nilai jarak kemudian akan diubah ke ddalam satuan meter dengan ketelitiasn 0,1 meter.

i. Aktivasi Relay untuk *Speed limiter*

```
if dis < 3:
    GPIO.output(Relay, GPIO.HIGH)
else:
    GPIO.output(Relay, GPIO.LOW)
```

Ketika sistem prediksi jarak mendeteksi nilai jarak kurang dari 3 meter maka relay akan diaktifkan dan memutus jalur *throtle* dengan *motor driver*. Namun jika pengukuran jarak lebih dari 3 meter jalur tersebut dihubungkan kembali dengan menonaktifkan relay agar tetap dalama kondisi NC.

j. Kalkulasi Frame Per Second (FPS)

```
elapsed_time = time.time() - starting_time
        fps = frame_id/elapsed_time
        cv2.putText(frame,"FPS:"+str(round(fps,
                2)),(10,50),font,2,(0, 0, 0),1)
```

Kalkulasi fps dilakukan dengan membagi jumlah frame yang telah diproses dengan waktu yang telah dilalui sejak pertama kali program dijalankan.

k. Penampilan Citra Hasil Deteksi

```
cv2.imshow("Image", frame)
key = cv2.waitKey(1)
        if key == 27:
           break
```

Citra hasil pengolahan deteksi objek dan yang telah diberi *bounding box* dirtammpilkan dengan menggunakan fungsi cv2.imshow(). Untuk menghentikan program berjalan terus menerus maka iterasi proses deteksi objek akan dihentikan ketika pengguna menekan tombol “Esc” pada keyboard.

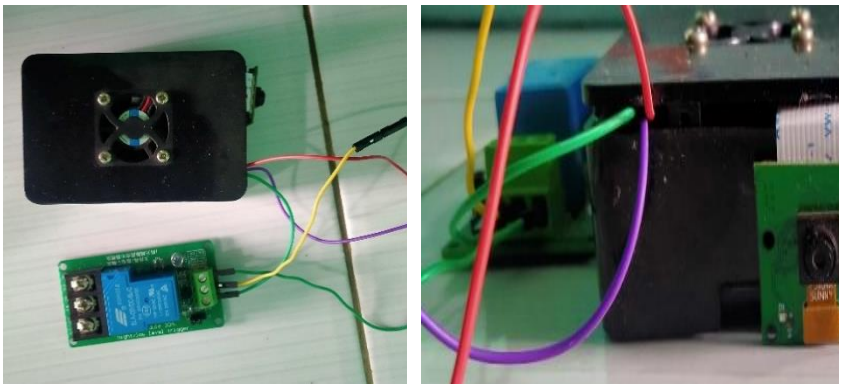
l. Akhiri Proses

```
cap.release()
cv2.destroyAllWindows()
```

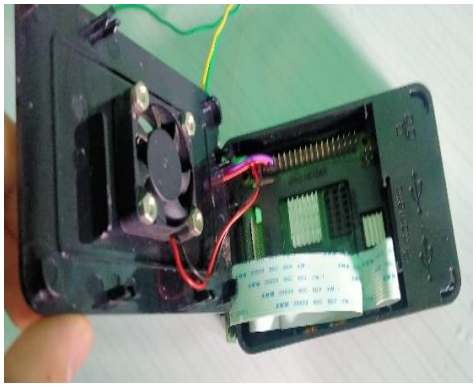
Ketika iterasi proses deteksi objek dihentikan maka untuk mengakhiri jalannya program dijalankan fungsi cap.release() untuk menghentikan pengambilan citra dari mnodul kamera Raspberry Pi dan cv2.destroyAllWindows() untuk menghentikan jendela OpenCV yang sedang berjalan.

4.1.3 Antarmuka *Hardware*

Antarmuka *hardware* yang dilakukan adalah dengan melakukan penyambungan relay ke pin GPIO 4 pada Raspbery Pi serta pemasangan modul kamera dan *casing* Raspberry Pi. Hasil antarmuka *hardware* sistem *speed limiter* dapat dilihat pada Gambar 4.2.



(A) (B)

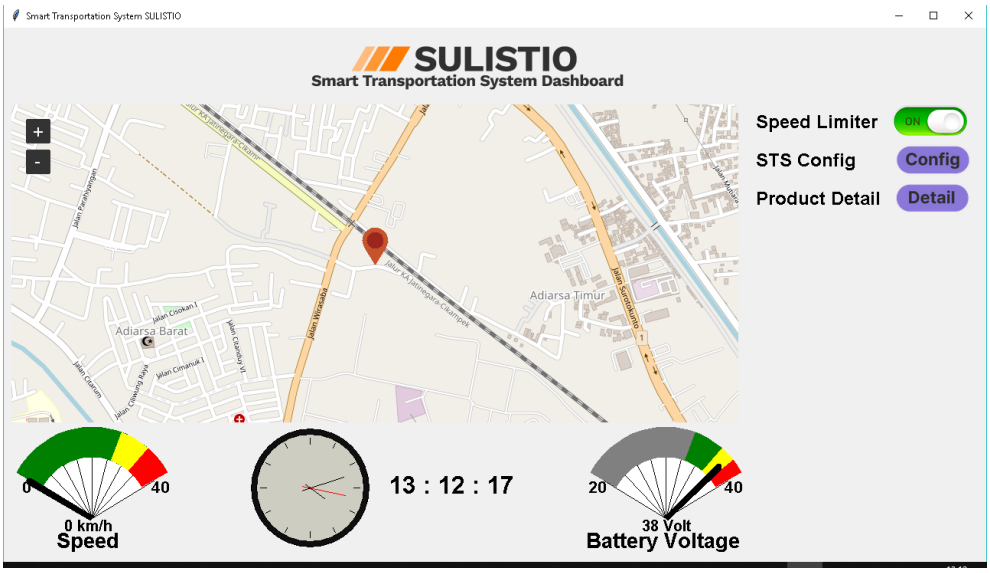


(C)

Gambar 4 2 Antarmuka hardware sistem *speed limiter* (A)Tampilan atas; (B) Tampilan Depan; (C) Tampilan Dalam Raspberry Pi

4.1.4 Antarmuka Sistem STS

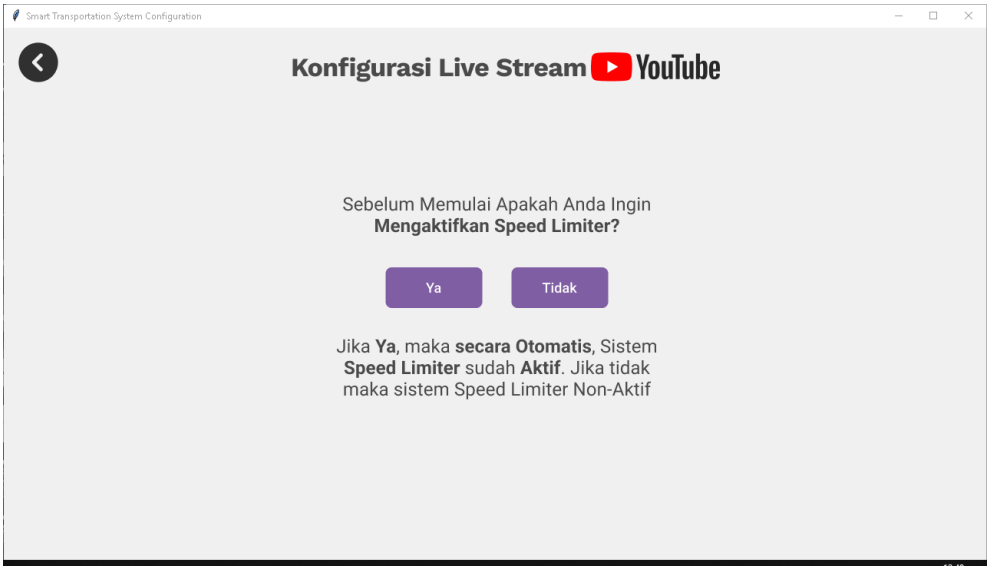
Program sistem *speed limiter* dihubungkan ke sistem STS secara modular. Fungsi deteksi objek dan *speed limiter* dijalankan secara *asynchronous* agar dapat berjalan bersamaan dengan sistem STS Dashboard. Penggunaan fitur *speed limiter* dapat dengan mudah diakses oleh pengguna dengan menekan *switch* "*Speed limiter*" pada Gambar 4.3 hingga berada pada posisi ON.



Gambar 4 3 Antarmuka sistem *speed limiter* pada Aplikasi Dashboard STS

Penggunaan bersama fitur *speed limiter* dengan layanan *livestream* STS dapat dilakukan dengan memilih kondisi "Ya" pada saat melakukan konfigurasi

awal livestream pada tombol STS Config. Langkah ini dapat dilihat pada Gambar 4.4.



Gambar 4 4 Halaman awaal konfigurasi *livestream* pada Aplikasi Dashboard STS

4.2 Pengujian

4.2.1. Kalibrasi Jarak

Kalibrasi jarak dibagi pada 4 objek deteksi yaitu pada objek manusia, mobil, motor, dan sepeda. Kalibrasi jarak dilakukan dengan tujuan mencari pewrsamaan yang dapat digunakan pada program *speed limiter* untuk mengalkulasi jarak pada masing-masing objek sesuai dengan masukan berupa keliling *bounding box* objek yang terdeteksi.

1. Kalibrasi Jarak pada Objek Manusia

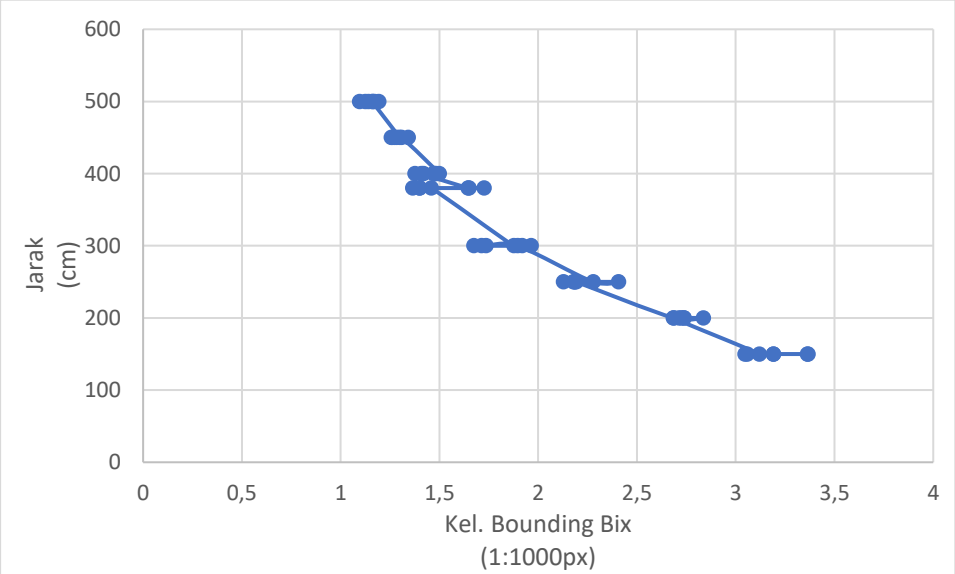
Kalibrasi jarak pada objek manusia menggunakan 56 sampel foto dengan masing-masing foto terdapat satu objek manusia. Kalibrasi jarak yang dilakukan hanya menggunakan satu objek manusia sebagai dataset kalibrasi. Hubungan antara *bounding box* pada masing-masing dataset terhadap jarak sebenarnya dapat dilihat pada Tabel 7.

Tabel 7. Hubungan antara keliling *bounding box* objek deteksi dengan jarak sebenarnya

Nomor Sampel	Keliling *Bounding box* (x1000px)	Jarak Sebenarnya (cm)
1	3,366	150
2	3,362	150
3	3,19	150
4	3,19	150
5	3,046	150
6	3,058	150

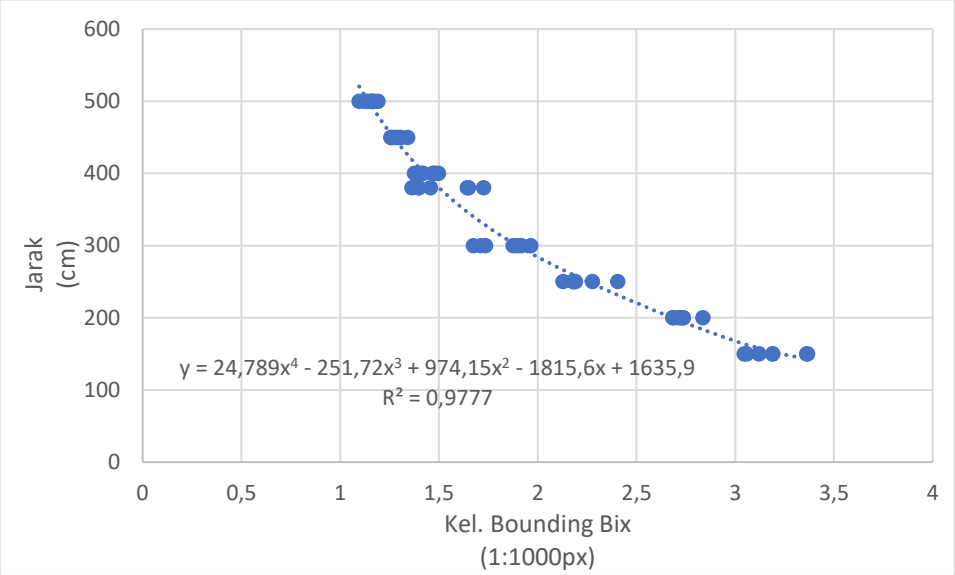
7	3,12	150
8	2,684	200
9	2,736	200
10	2,738	200
11	2,724	200
12	2,712	200
13	2,836	200
14	2,686	200
15	2,192	250
16	2,184	250
17	2,184	250
18	2,178	250
19	2,128	250
20	2,406	250
21	2,278	250
22	1,896	300
23	1,736	300
24	1,712	300
25	1,674	300
26	1,964	300
27	1,918	300
28	1,876	300
29	1,458	350
30	1,4	350
31	1,364	350
32	1,398	350
33	1,726	350
34	1,65	350
35	1,644	350
36	1,408	400
37	1,42	400
38	1,482	400
39	1,376	400
40	1,474	400
41	1,47	400
42	1,498	400
43	1,308	450
44	1,26	450
45	1,272	450
46	1,256	450
47	1,342	450

48	1,284	450
49	1,298	450
50	1,16	500
51	1,172	500
52	1,162	500
53	1,192	500
54	1,124	500
55	1,096	500
56	1,14	500



Gambar 4. 5 Grafik hubungan antara keliling *bounding box* objek deteksi dengan jarak sebenarnya

Penentuan persamaan garis pada grafik dilakukan dengan menggunakan tool Trendline pada *Microsoft Excel*. Hasil proses oleh tool *Trendline* pada *Microsoft Excel* dapat dilihat pada Gambar 4.6.



Gambar 4. 6 Hasil pengolahan tool Trendline pada *Microsoft Excel* terhadap grafik hubungan antara keliling *bounding box* objek deteksi dengan jarak sebenarnya

Berdasarkan hasil pada gambar 3 didapati model matematis yang mendekati nilai aktualnya sesuai pada persamaan (4.1).

$$y = 24{,}789x^4 - 251{,}72x^3 + 974{,}15x^2 - 1816{,}6x + 1635{,}9 \qquad (4.1)$$

Dimana:

y = Jarak kendaraan

x = Keliling *Bounding box* skala 1:1000px

Dengan persamaan tersebut didapati nilai $R^2 = 0{,}9777$

Menggunakan model matematis pada persamaan (1) didapati kalkulasi jarak kendaraan pada sistem *speed limiter* dapat dilakukan menggunakan sintaks berikut:

```
//Keliling bounding box
kel = (2*h+2*w)/1000
//Kalkulasi Jarak
dis = 24.789*(kel*kel*kel*kel) - 251.72 * (kel * kel*kel)
      + 974.15*(kel*kel) - 1815.6 * kel + 1635.9
dis = round(dis/10)
dis = dis/10 //Menghasilkan nilai jarak dengan satuan Meter
```

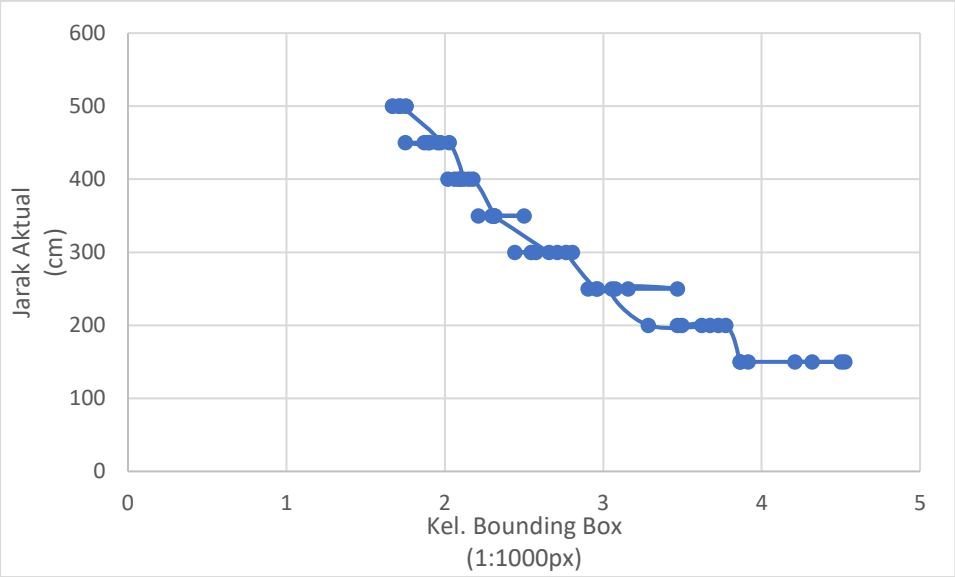
2. Kalibrasi Jarak pada Objek Mobil

Kalibrasi jarak pada objek mobil menggunakan 56 sampel foto dengan masing-masing foto terdapat satu objek mobil. Kalibrasi jarak yang dilakukan hanya menggunakan satu objek mobil sebagai dataset kalibrasi. Hubungan antara *bounding box* pada masing-masing dataset terhadap jarak sebenarnya dapat dilihat pada Tabel 8.

Tabel 8. Hubungan antara keliling *bounding box* objek deteksi dengan jarak sebenarnya

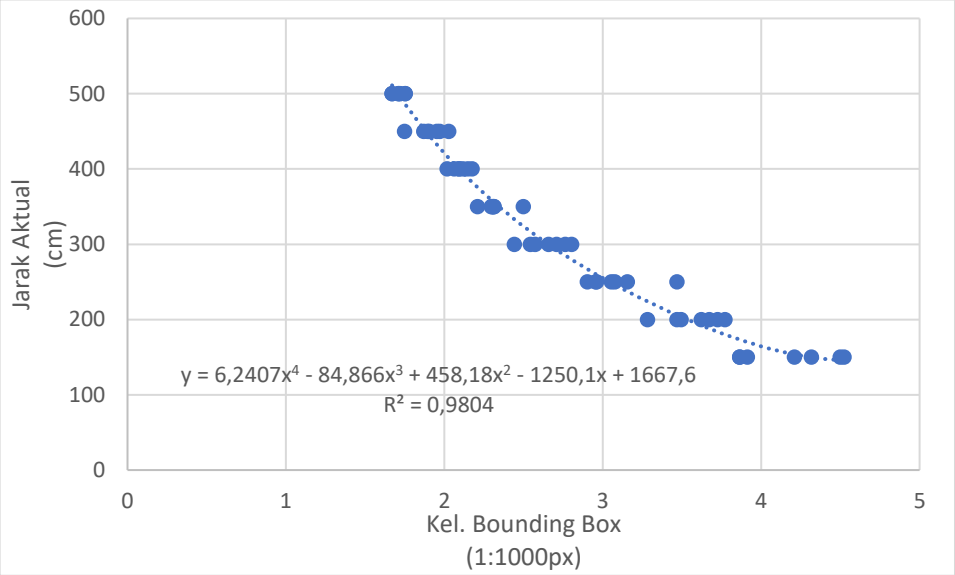
Nomor Sampel	Keliling *Bounding box* (x1000px)	Jarak Sebenarnya (cm)
1	4,318	150
2	4,21	150
3	4,524	150
4	4,5	150
5	3,914	150
6	3,864	150
7	3,864	150
8	3,772	200
9	3,496	200
10	3,622	200
11	3,468	200
12	3,674	200
13	3,726	200
14	3,284	200
15	3,054	250
16	3,468	250
17	2,904	250
18	3,156	250
19	2,962	250
20	3,076	250
21	2,956	250
22	2,764	300
23	2,804	300
24	2,574	300
25	2,544	300
26	2,442	300
27	2,71	300
28	2,658	300
29	2,3	350
30	2,21	350
31	2,316	350
32	2,31	350
33	2,5	350
34	2,296	350
35	2,312	350
36	2,176	400
37	2,018	400
38	2,092	400

39	2,15	400
40	2,092	400
41	2,062	400
42	2,118	400
43	2,028	450
44	1,894	450
45	1,87	450
46	1,954	450
47	1,904	450
48	1,75	450
49	1,972	450
50	1,722	500
51	1,67	500
52	1,71	500
53	1,754	500
54	1,67	500
55	1,71	500
56	1,754	500



Gambar 4. 7 Grafik hubungan antara keliling *bounding box* objek deteksi dengan jarak sebenarnya

Penentuan persamaan garis pada grafik dilakukan dengan menggunakan tool Trendline pada *Microsoft Excel*. Hasil proses oleh tool *Trendline* pada *Microsoft Excel* dapat dilihat pada Gambar 4.8.



Gambar 4. 8 Hasil pengolahan tool Trendline pada *Microsoft Excel* terhadap grafik hubungan antara keliling *bounding box* objek deteksi dengan jarak sebenarnya

Berdasarkan hasil pada gambar 4 didapati model matematis yang mendekati nilai aktualnya sesuai pada persamaan (4.2).

$$y = 6{,}2407x^4 - 84{,}866x^3 + 458{,}18x^2 - 1250{,}1x + 1667{,}6 \quad (4.2)$$

Dimana:

y = Jarak kendaraan

x = Keliling *Bounding box* skala 1:1000px

Dengan persamaan tersebut didapati nilai $R^2 = 0{,}9804$

Menggunakan model matematis pada persamaan (4.2) didapati kalkulasi jarak kendaraan pada sistem *speed limiter* dapat dilakukan menggunakan sintaks berikut:

```
//Keliling bounding box
kel = (2*h+2*w)/1000
//Kalkulasi Jarak
dis = 6.2407*(kel*kel*kel*kel) - 84.866*(kel*kel*kel) +
      458.18 * (kel*kel)-1250.1*kel + 1667.6
dis = round(dis/10)
dis = dis/10 //Menghasilkan nilai jarak dengan satuan Meter
```

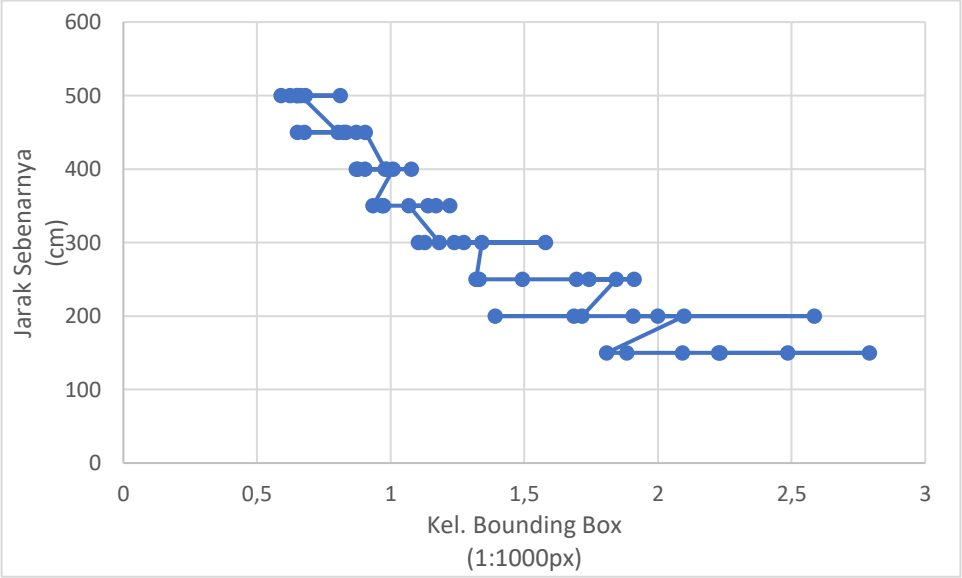
3. Kalibrasi Jarak pada Objek Motor

Kalibrasi jarak pada objek motor menggunakan 56 sampel foto dengan masing-masing foto terdapat satu objek motor. Kalibrasi jarak yang dilakukan hanya menggunakan satu objek motor sebagai dataset kalibrasi. Hubungan antara *bounding box* pada masing-masing dataset terhadap jarak sebenarnya dapat dilihat pada Tabel 9.

Tabel 9. Hubungan antara keliling *bounding box* objek deteksi dengan jarak sebenarnya

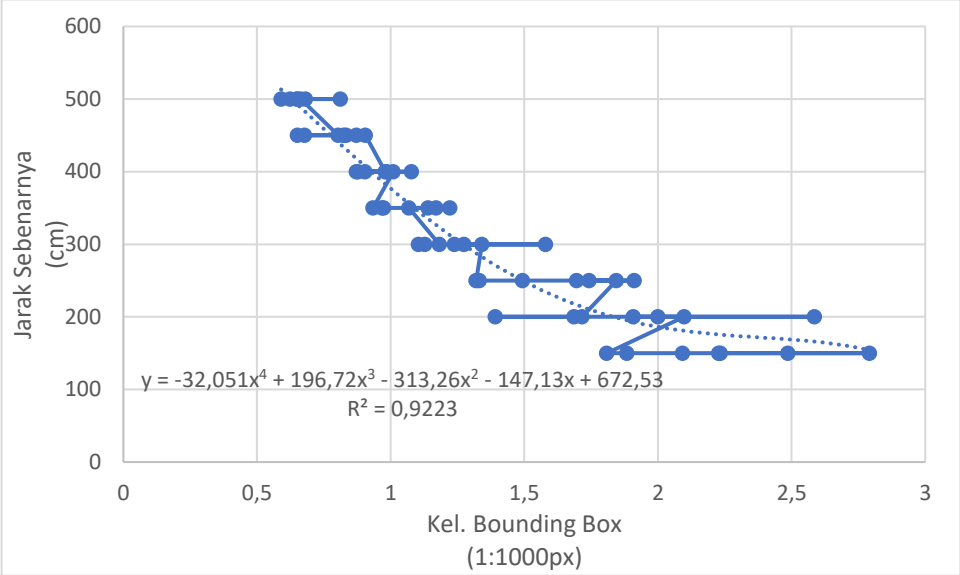
Nomor Sampel	Keliling *Bounding box* (x1000px)	Jarak Sebenarnya (cm)
1	2,228	150
2	2,234	150
3	2,486	150
4	2,792	150
5	2,092	150
6	1,884	150
7	1,808	150
8	2,098	200
9	1,908	200
10	2	200
11	2,586	200
12	1,686	200
13	1,392	200
14	1,716	200
15	1,844	250
16	1,696	250
17	1,742	250
18	1,912	250
19	1,494	250
20	1,332	250
21	1,32	250
22	1,342	300
23	1,274	300
24	1,238	300
25	1,58	300
26	1,128	300
27	1,104	300
28	1,182	300
29	1,068	350
30	1,14	350
31	1,17	350
32	1,222	350
33	0,974	350
34	0,968	350
35	0,934	350
36	1,01	400
37	0,904	400
38	0,986	400
39	1,078	400

40	0,872	400
41	0,878	400
42	0,98	400
43	0,906	450
44	0,834	450
45	0,872	450
46	0,824	450
47	0,678	450
48	0,652	450
49	0,804	450
50	0,664	500
51	0,682	500
52	0,65	500
53	0,812	500
54	0,59	500
55	0,65	500
56	0,624	500



Gambar 4 .9 Grafik hubungan antara keliling *bounding box* objek deteksi dengan jarak sebenarnya

Penentuan persamaan garis pada grafik dilakukan dengan menggunakan tool Trendline pada *Microsoft Excel*. Hasil proses oleh tool *Trendline* pada *Microsoft Excel* dapat dilihat pada gambar 4.10.



Gambar 4 .10 Hasil pengolahan tool Trendline pada *Microsoft Excel* terhadap grafik hubungan antara keliling *bounding box* objek deteksi dengan jarak sebenarnya

Berdasarkan hasil pada Gambar 4.10 didapati model matematis yang mendekati nilai aktualnya sesuai pada persamaan (4.3).

$$y = -32{,}051x^4 + 196{,}72x^3 - 313{,}26x^2 + 147{,}13x - 672{,}53 \quad (4.3)$$

Dimana:

y = Jarak kendaraan

x = Keliling *Bounding box* skala 1:1000px

Dengan persamaan tersebut didapati nilai $R^2 = 0{,}9223$

Menggunakan model matematis pada persamaan (4.3) didapati kalkulasi jarak kendaraan pada sistem *speed limiter* dapat dilakukan menggunakan sintaks berikut:

```
//Keliling Bounding box
kel = (2*h+2*w)/1000
//Kalkulasi Jarak
dis = -32.051 * (kel*kel*kel*kel) + 196.72 * (kel*kel*kel)
      -313.26 * (kel*kel) - 147.13 * kel + 672.53
dis = round(dis/10)
dis = dis/10 //Menghasilkan nilai jarak dengan satuan Meter
```

4.2.2. Pengujian sistem *Speed limiter*

4.2.2.1 Pengujian Sistem Deteksi objek

Pengujian sistem deteksi objek bertujuan untuk mengetahui performa sistem deteksi obnjek dengan pre-trained weights yang dimiliki oleh yolov4-tiny. Pengujian dilakukan dengan menggunakan 150 citra yang didapat dari pemecahan

video yang didapat dari 3 lokasi berbeda pada jalan di Kelurahan Telukjambe Timur, Kabupaten Karawang. Pencarian nilai IoU dilakukan dengan menghitung nilai IoU pada hasil deteksi objek dengan citra yang sudah dilakukan anotasi sebelumnya.

Pengujian difokuskan pada deteksi 3 kelas yaitu kelas manusia, motor dan mobil. Hasil pengujian sistem deteksi objek dengan *pre-trained weights* yolov4-tiny dapat dilihat pada Tabel 10. Pada tabel 10 dapat diketahui bahwa sistem deteksi memiliki nilai *Average Precision (AP)* sebesar 100% untuk kelas mobil, 92,74% untuk kelas motor dan 100% untuk kelas Manusia. Hasil pengujian juga menunjukkan nilai *False Negatif* yang cukup tinggi yaitu 144 objek yang mana hal ini menandakan bahwa sistem memiliki kepekaan deteksi yang cukup sebesar 68,3%.

Tabel 10. Pengujian sistem deteksi dengan *pre-trained wighths* yolov4-tiny

Mobil	AP	100,00%
	TP	80
	FP	0
Motor	AP	92,74%
	TP	85
	FP	15
Manusia	AP	100,00%
	TP	145
	FP	0
FN		144
Waktu Proses Rata-Rata		0,2906
Recall		0,88235
Precision		0,95
F1-Score		0,914926
IoU		66,14%
mAP @0,5		97,58%

Waktu yang dibutuhkan oleh sistem untuk melakukan deteksi citra terbilang baik dengan hanya membutuhkan waktu rata-rata 0,29 detik atau 3,4 FPS. *Recall* dan *Precision* dari sistem deteksi juga terbilang baik dengan nilai *Recall* sebesar 0,88235 dan nilai *Precision* sebesar 0,95 atau 95%. Dengan nilai *Recall* dan *Precision* yang besar sistem meiliki *F1-Score* yang tinggi juga sebesar 0,914926

atau 91,49% yang menandakan sistem deteksi memiliki akurasi prediksi yang baik. Dengan nilai *Average Precision* pada masing-masing kelas didapati nilai *mean Average Precision* pada sistem sebesar 97,58%.

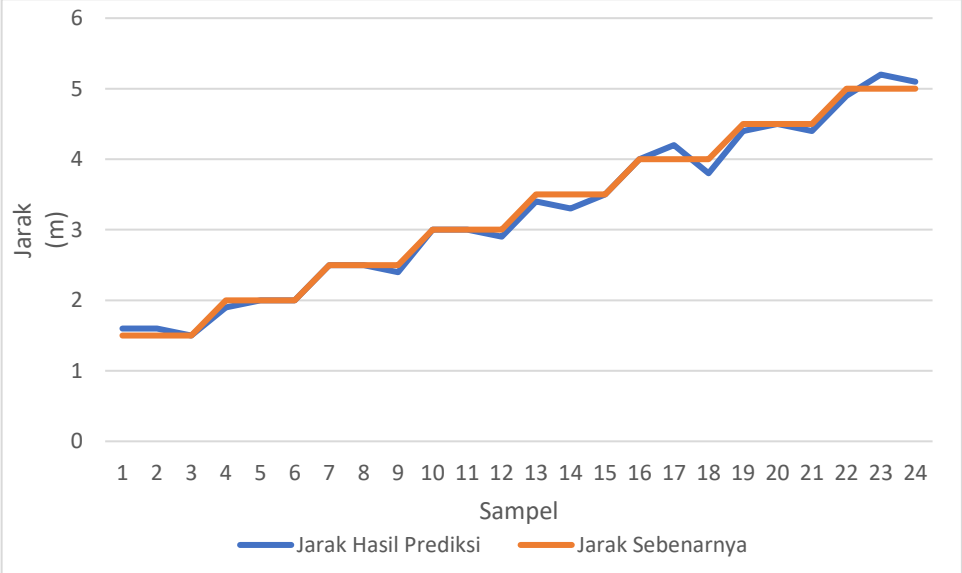
Akurasi deteksi dapat juga dilihat dari nilai *confidence level* hasil deteksi sistem. Pengujian pada tahapan ini dilakukan dengan mencari nilai rata-rata dari *confidence* hasil deteksi pada masing-masing frame video terhadap masing-masing kelas yang dideteksi. Hasil pengujian tingkat akurasi deteksi yang diberikan oleh sistem dapat dilihat pada Tabel 11.

Tabel 11. Pengujian sistem deteksi dengan *pre-trained wighths* yolov4-tiny

Lokasi Video	Mobil	Motor	Manusia	Rata-rata waktu proses tiap frame (detik)
Jalan Adiarsa	66,95%	68,41%	60,13%	0,2347
Jalan Babakan Sananga	64,05%	73,18%	65,52%	0,2535
Jalan Galuh Mas	65,3%	61%	65%	0,2433

4.2.2.2 Pengujian Sistem Prediksi Jarak

Pengujian prediksi jarak dilakukan pada masing-masing kelas objek. Hasil pengujian disajikan dalam bentuk tabel yang dapat dilihaty pada Lampiran B serta grafik yang akan dibahas. Grafik hasil pengujian prediksi jarak pada kelas manusia dapat dilihat pada Gambar 4.11.



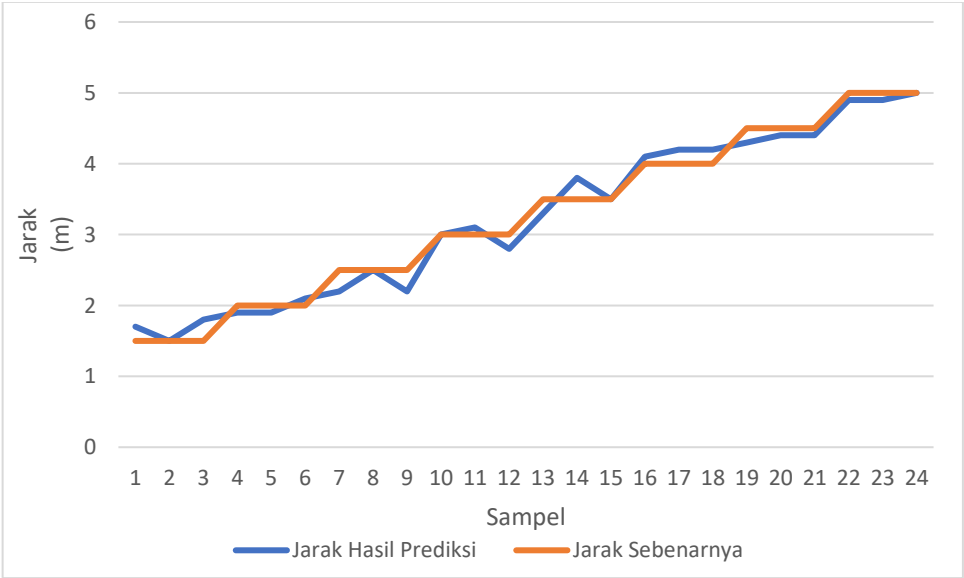
Gambar 4 11 Grafik hasil pengujian prediksi jarak kelas objek kelas manusia

Pengujian prediksi jarak pada kelas manusia menghasilkan nilai kesalahan rata-rata sebesar 0,075 meter dengan nilai kesalahan rata-rata 2,69% atau akurasi

prediksi sebesar 97,31%. Citra hasil pengujian jarak dapat dilihat pada Gambar 4.12.

Gambar 4 12 Hasil Pengujian Prediksi Jarak pada Kelas Manusia

Grafik hasil pengujian prediksi jarak pada kelas motor dapat dilihat pada Gambar 4.13.

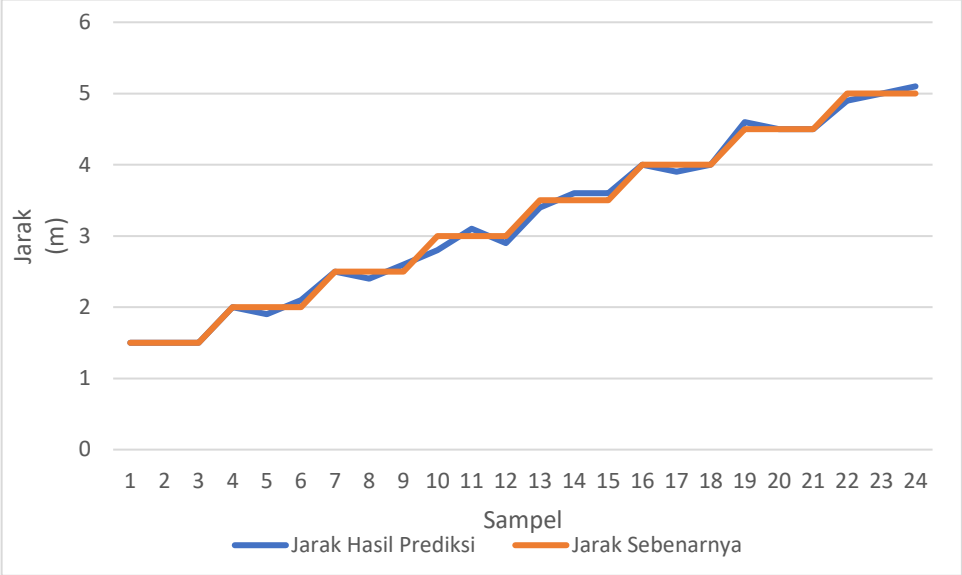


Gambar 4 13 Grafik hasil pengujian prediksi jarak kelas objek kelas motor

Pengujian prediksi jarak pada kelas motor menghasilkan nilai kesalahan rata-rata sebesar 0,1375 meter dengan nilai kesalahan rata-rata 5,08%atau akurasi prediksi sebesar 94,92%. Prediksi jarak pada kelas motor merupakan prediksi dengan nilai akurasi paling rendah dibandingkan nilai akurasi pada kelas lainnya. Contoh citra hasil pengujian prediksi jarak pada kelas motor dapat dilihat pada Gambar 4.14.

Gambar 4 14 Hasil Pengujian Prediksi Jarak pada Kelas Motor

Grafik hasil pengujian prediksi jarak pada kelas Mobil dapat dilihat pada Gambar 4.15.

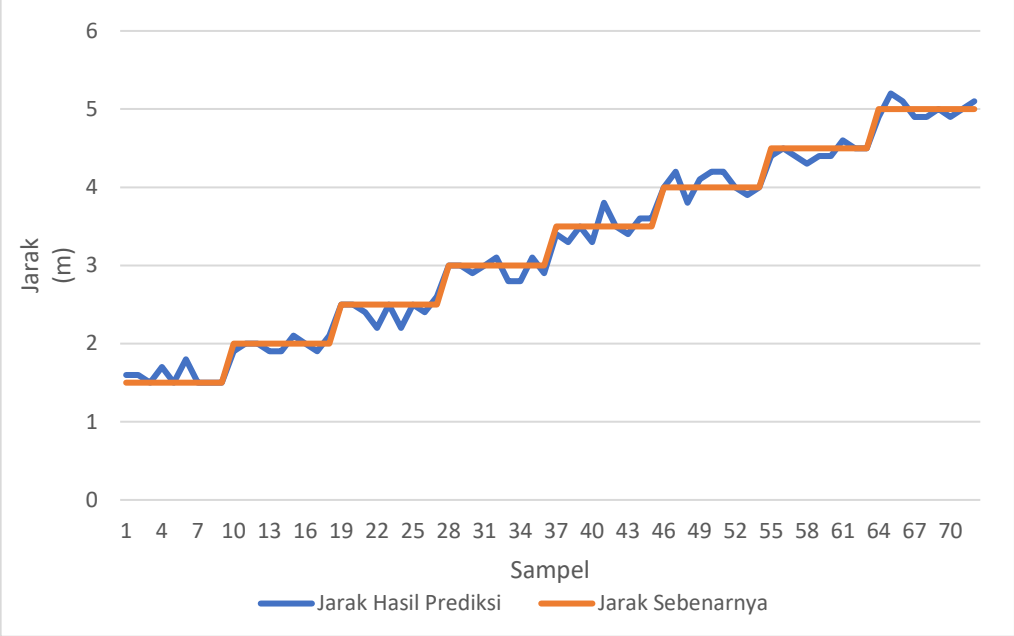


Gambar 4 15 Grafik hasil pengujian prediksi jarak kelas objek kelas mobil

Pengujian prediksi jarak pada kelas motor menghasilkan nilai kesalahan rata-rata sebesar 0,0625 meter dengan nilai kesalahan rata-rata 2,03%atau akurasi prediksi sebesar 97,97%. Prediksi jarak pada kelas mobiul merupakan kelas dengan nilai akurasi prediksi yang paling tinggi dibandingkan dengan kelas lainnya. Contoh citra hasil pengujian prediksi jarak pada kelas motor dapat dilihat pada Gambar 4.16.

Gambar 4 16 Hasil Pengujian Prediksi Jarak pada Kelas Mobil

Secara keseluruhan nilai akurasi prediksi jarak pada seluruh kelas adalah 97% dengan nilai selisih rata-rata dari prediksi jarak dengan jarak yang sebenarnya sernilai 0,092 m. Grafik hasil pengujian jarak pada keseluruhan kelas dapat dilihat pada Gambar 4.17.



Gambar 4 17 Grafik hasil pengujian prediksi jarak pada keseluruhan kelas objek

4.2.2.3 Pengujian Kontrol Relay

Pengujian kontrol relay bertujuan untuk mengetaui performa kerja relay pada sistem *speed limiter* dalam memutus jalur tegangan pada *throtle* ketika objek di depan sepeda berjarak kurang dari 3 meter. Pemilihan set poin di 3 meter ini didapat dari pertimbangan kekuatan rem sepeda yang pada kecepatan maksimalnya baru bisa berhenti ketika dilakukan rem total pada jarak 4 meter. Dengan asumsi bahwa objek di depan sepeda masih dalam kondisi bergerak maka set poin sejauh 3 meter ini dipilih. Citra yang digunakan pada pengujian ini adalah dengan mengambil citra secara acak pada citra pengujian deteksi objek dan menerapkan prediksi jarak pada citra. Hasil pengujian kerja relay pada sistem *speed limiter* dapat dilihat pada Tabel 12.

Tabel 12. Hasil Pengujian Kerja Kontrol Relay

No	Jarak Prediksi (m)	Kondisi Relay
1	2,1	LOW
2	1,2	LOW
3	2,9	LOW
4	3,4	HIGH

5	4,3	HIGH
6	2,7	LOW
7	3,3	HIGH
8	1,5	LOW
9	1,7	LOW
10	3,7	HIGH
11	2,2	LOW
12	3,6	HIGH
13	4,8	HIGH
14	2,3	LOW
15	4,4	HIGH
16	5,0	HIGH
17	3,2	HIGH
18	2,2	LOW
19	1,6	LOW
20	3,6	HIGH

Berdasarkan hasil yang didapat pada Tabel 12 dapat disimpulkan bahwa kinerja kontrol relay pada sistem berjalan dengan sangat baik dan menghasilkan 100% kondisi True Positive. Hal ini mengartikan bahwa sistem tidak memiliki kesalahan logika kerja dan sesuai dengan hasil yang diharapkan.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil pengujian yang dilakukan pada sisitem *speed limiter* menggunakan algoritma YOLO pada versi YOLOV4 dapat disimpulkan sebagai poin-poin berikut:

1. Sistem speed limiter melakukan deteksi objek menggunakan algoritma YOLO sebagai network *CNN*-nya. Prediksi jarak pada sistem *speed limiter* dilakukan dengan menghitung keliling *bounding box* pada objek deteksi. Untuk memperbaiki performanya maka dilakukan Kalibrasi jarak yang menghasilkan model matematis melalui tool *Trendline* pada *Microsoft Excel*. Kalibrasi jarak menghasilkan model persamaan matematis sistem yang baik dengan nilai R^2 pada masing-masing kelas selalu melebihi 0,9.
2. Antarmuka *hardware* sistem dilakukan dengan menambahkan relay pada jalur *throtle* menuju *motor driver* untuk menghubungkan dan memutus jalur tegangannya. Sedangkan pada antarmuka *software* sistem dapat dengan mudah diakses pada aplikasi *STS Dashboard*.
3. Sistem dapat mendeteksi objek dengan kelas manusia, motor, dan mobil dengan baik dengan tingkat F1-Score sebesar 0,914926, IoU sebesar 66,14%, dan nilai mAP pada treshold 0,5 sebesar 97,58%. Hasil model prediksi jarak pada sistem *speed limiter* sangatlah baik dengan nilai akurasi 97,31% pada kelas Manusia, 94,92% pada kelas Motor, serta 97,97% pada kelas Mobil dengan akurasi keseluruhan kelas adalah sebesar 97%.
4. Hasil performa kerja sistem kontrol relay bekerja dengan baik dan menghasilkan kesalahan logika sebesar 0% yang mengartikan bahsa sistem sepenuhnya menjalankan hasil sesuai dengan yang diharapkan.

5.2 Saran

Saran penulis pada penelitian selanjutnya dalam upaya pengembangan produk *speed limiter* pada sepeda listrik adalah sebagai berikut:

1. Membuat sistem pengereman otomatis dari data prediksi jarak agar sistem keselamatan berkendara semakin baik dan mencegah terjadinya kecelakaan akibat kelalaian pengemudi meskipun sistem pembatas kecepatan telah aktif.
2. Menerapkan lebih banyak kelas objek agar sistem pembatas kecepatan dapat berlaku pada kelas lainnya.
3. Menggunakan resolusi kamera yang lebih baik agar kualitas deteksi objek dan prediksi jarak lebih optimal.
4. Menggunakan spesifikasi *hardware* yang lebih baik dalam implementasinya agar waktu proses deteksi objek dapat lebih singkat dan lebih *real time*.

DAFTAR PUSTAKA

[1] Undang-Undang No.22 Tahun 2009, Tentang Lalu Lintas dan Angkutan Jalan, Jakarta, 2009.

[2] M. L. Andika, "detikOto," detik, 16 November 2022. [Online]. Available: https://oto.detik.com/berita/d-5256457/kerugian-akibat-kecelakaan-lalu-lintas-januari-oktober-2020-rp-163-m-lebih. [Accessed 4 April 2022].

[3] Undang-Undang Nomor 22 Tahun 2009 tentang Lalu Lintas dan Angkutan Jalan, Jakarta, 2009.

[4] "Penjualan E-bike Meningkat di Masa Pandemi," Melotrinic, 30 Maret 2021. [Online]. Available: https://melotronic.com/penjualan-e-bike-meningkat-di-masa-pandemi/. [Accessed 20 April 2021].

[5] F. Amalia, PERANCANGAN SISTEM BERBASIS INTERNET OF THINGS (IOT) UNTUK EFISIENSI BIAYA PEMAKAIAN ENERGI LISTRIK PADA GEDUNG KULIAH JURUSAN TEKNIK ELEKTRO, Palembang: Politeknik Negeri Sriwijaya, 2020.

[6] V. I. Jalled F, "Object Detection using Image Processing," 2016. [Online]. Available: Retrieved from http://arxiv.org/abs/1611.07791.

[7] "About OpenCV," OpenCV, [Online]. Available: https://opencv.org/about/. [Accessed 20 April 2022].

[8] D. Thuan, Evolution of YOLO Algorithm and YOLOv5: The State-of-the-art Object Detection, Oulu: Oulu University of Applied Sciences, 2021.

[9] D. S. G. R. ,. F. A. Redmon. J, "You only look once: Unified, real-time object detection.," in *IEEE conference on computer vision and pattern recognition*, Las Vegas, 2016.

[10] "*Microsoft Excel*," Wikipedia, 5 Maret 2022. [Online]. Available: https://id.wikipedia.org/wiki/Microsoft_Excel. [Accessed 16 Juni 2022].

[11] H. M. S. N. S. R. R. F. Umar Y, "Deteksi Penggunaan Helm Pada Pengendara".

[12] A. Amwin, DETEKSI DAN KLASIFIKASI KENDARAAN BERBASIS ALGORITMA YOU ONLY LOOK ONCE (YOLO), Yogyakarta: Fakultas Teknbologi Industri Universitas Islam Indonesia, 2021.

[13] E. W. Wayan Gede Suka Parwita, "Hybrid Recommendation System Memanfaatkan Penggalian Frequent Itemset dan Perbandingan Keyword," *IJCCS,* vol. 9, no. 2, pp. 167-176, 2015.

LAMPIRAN A

Klasifikasi objek *pre-trained model* YOLOV4

Nama File	Klasifikasi Objek		
coco.names	person	frisbee	toilet
	bicycle	skis	tvmonitor
	car	snowboard	laptop
	motorbike	sports ball	mouse
	aeroplane	kite	remote
	bus	baseball bat	keyboard
	train	baseball	cell phone
	truck	glove	microwave
	boat	skateboard	oven
	traffic light	surfboard	toaster
	fire hydrant	tennis racket	sink
	stop sign	bottle	refrigerator
	parking	wine glass	book
	meter	cup	clock
	bench	fork	vase
	bird	knife	scissors
	cat	spoon	teddy bear
	dog	bowl	hair drier
	horse	banana	toothbrush
	sheep	apple	sofa
	cow	sandwich	pottedplant
	elephant	orange	bed
	bear	broccoli	diningtable
	zebra	carrot	handbag
	giraffe	hot dog	tie
	backpack	pizza	suitcase
	umbrella	donut	chair
			cake

LAMPIRAN B

Tabel Hasil Pengujian Prediksi Jarak

Nomor	Kelas	Jarak Pembacaan (m)	Jarak Sebenarnya (m)	Selisih (m)	Galat
1	Manusia	1,6	1,5	0,1	6,67%
2	Manusia	1,6	1,5	0,1	6,67%
3	Manusia	1,5	1,5	0	0,00%
4	Manusia	1,9	2	0,1	5,00%
5	Manusia	2	2	0	0,00%
6	Manusia	2	2	0	0,00%
7	Manusia	2,5	2,5	0	0,00%
8	Manusia	2,5	2,5	0	0,00%
9	Manusia	2,4	2,5	0,1	4,00%
10	Manusia	3	3	0	5,00%
11	Manusia	3	3	0	0,00%
12	Manusia	2,9	3	0,1	3,33%
13	Manusia	3,4	3,5	0,1	2,86%
14	Manusia	3,3	3,5	0,2	5,71%
15	Manusia	3,5	3,5	0	0,00%
16	Manusia	4	4	0	0,00%
17	Manusia	4,2	4	0,2	5,00%
18	Manusia	3,8	4	0,2	5,00%
19	Manusia	4,4	4,5	0,1	5,00%
20	Manusia	4,5	4,5	0	0,00%
21	Manusia	4,4	4,5	0,1	2,22%
22	Manusia	4,9	5	0,1	2,00%
23	Manusia	5,2	5	0,2	4,00%
24	Manusia	5,1	5	0,1	2,00%
25	Motor	1,7	1,5	0,2	13,33%
26	Motor	1,5	1,5	0	0,00%
27	Motor	1,8	1,5	0,3	20,00%
28	Motor	1,9	2	0,1	5,00%
29	Motor	1,9	2	0,1	5,00%
30	Motor	2,1	2	0,1	5,00%
31	Motor	2,2	2,5	0,3	12,00%
32	Motor	2,5	2,5	0	0,00%
33	Motor	2,2	2,5	0,3	12,00%
34	Motor	3	3	0	0,00%
35	Motor	3,1	3	0,1	3,33%
36	Motor	2,8	3	0,2	6,67%

37	Motor	3,3	3,5	0,2	5,71%
38	Motor	3,8	3,5	0,3	8,57%
39	Motor	3,5	3,5	0	0,00%
40	Motor	4,1	4	0,1	2,50%
41	Motor	4,2	4	0,2	5,00%
42	Motor	4,2	4	0,2	5,00%
43	Motor	4,3	4,5	0,2	4,44%
44	Motor	4,4	4,5	0,1	2,22%
45	Motor	4,4	4,5	0,1	2,22%
46	Motor	4,9	5	0,1	2,00%
47	Motor	4,9	5	0,1	2,00%
48	Motor	5	5	0	0,00%
49	Mobil	1,5	1,5	0	0,00%
50	Mobil	1,5	1,5	0	0,00%
51	Mobil	1,5	1,5	0	0,00%
52	Mobil	2	2	0	0,00%
53	Mobil	1,9	2	0,1	5,00%
54	Mobil	2,1	2	0,1	5,00%
55	Mobil	2,5	2,5	0	0,00%
56	Mobil	2,4	2,5	0,1	4,00%
57	Mobil	2,6	2,5	0,1	4,00%
58	Mobil	2,8	3	0,2	6,67%
59	Mobil	3,1	3	0,1	3,33%
60	Mobil	2,9	3	0,1	3,33%
61	Mobil	3,4	3,5	0,1	2,86%
62	Mobil	3,6	3,5	0,1	2,86%
63	Mobil	3,6	3,5	0,1	2,86%
64	Mobil	4	4	0	0,00%
65	Mobil	3,9	4	0,1	2,50%
66	Mobil	4	4	0	0,00%
67	Mobil	4,6	4,5	0,1	2,22%
68	Mobil	4,5	4,5	0	0,00%
69	Mobil	4,5	4,5	0	0,00%
70	Mobil	4,9	5	0,1	2,00%
71	Mobil	5	5	0	0,00%
72	Mobil	5,1	5	0,1	2,00%
Jumlah				**6,6**	**235%**
Rata-rata				**0,091667**	**3%**