Pendeteksian Sepeda Motor di Jalur Khusus Sepeda Menggunakan Algoritma Pendeteksi Objek YOLO

Ignasius Widira Kristianto [1] **Yulia Ery Kurniawati** [2]

Informatika, Fakultas Industri Kreatif Institut Teknologi dan Bisnis Kalbis
Jalan Pulomas Selatan Kav. 22, Jakarta 13210
[1]*Email: ignasiuskristianto@gmail.com*
[2] *Email) yulia.kurniawati@kalbis.ac.id*

Abstract: *Bicycles as transportation by the public is increasing. The government supports this by providing bike lanes. However, limited supervision by security forces poses a safety threat for cyclists against irresponsible persons. This research proposes a method of detecting vehicles in bike lanes from CCTV video using YOLOv3 algorithm and creating a detection area on the bike lane. The YOLO algorithm was chosen because it has a high FPS value. Vehicles that can be identified are motorcycles and bicycles. The bike lanes detection area is determined manually following traffic markings, while the vehicle object detection used YOLO v3 Tiny. YOLO v3 Tiny obtained an mAP accuracy value of 72.73% with CPU processing performance reaching three frames per second. Based on application testing, the application can distinguish bicycles and motorcycles in the bike lanes. However, YOLOv3 Tiny's low mAP accuracy can cause misdetection in applications.*
Keywords: *bike lane, closed circuit television (cctv), object detection, yolo*

Abstrak: *Penggunaan sepeda sebagai alat transportasi oleh masyarakat semakin meningkat. Hal tersebut didukung oleh pemerintah dengan menyediakan jalur khusus sepeda. Namun keterbatasan pengawasan oleh pihak keamanan menjadi ancaman keselamatan bagi pengendara sepeda terhadap oknum tidak bertanggung jawab. Penelitian ini mengusulkan metode pendeteksian kendaraan di jalur sepeda dari video CCTV menggunakan Algoritma YOLOv3 dan pembuatan area deteksi di jalur sepeda. Algoritma YOLO dipilih karena memiliki nilai FPS yang tinggi. Kendaraan yang dapat terdeteksi adalah sepeda motor dan sepeda. Area deteksi jalur sepeda ditemtukan secara manual mengikuti marka lalu lintas, sedangkan pendeteksian objek kendaraan menggunakan YOLO v3 Tiny. YOLO v3 Tiny memiliki nilai akurasi mAP yaitu 72,73% dengan performa pemrosesan CPU mencapai tiga frame per detik. Berdasarkan pengujian terhadap aplikasi, aplikasi dapat membedakan sepeda dan sepeda motor yang ada di jalur sepeda. Namun akurasi mAP YOLOv3 Tiny yang rendah menyebabkan kesalahan pendeteksian pada aplikasi.*
Kata kunci: *closed circuit television (cctv), deteksi objek, jalur sepeda, yolo*

I. PENDAHULUAN

Kesadaran masyarakat untuk hidup sehat dan mendukung kelestarian alam, mengubah perilaku hidup khususnya penggunaan transportasi. Sepeda menjadi pilihan transportasi yang murah dan mudah. Pemerintah telah menyediakan sarana, prasarana serta regulasi untuk mendukung hal tersebut. Namun, ada saja oknum yang tetap melanggar hak para pesepeda dan membahayakan keselamatan berlalu lintas.

Keterbatasan pihak kepolisian untuk mengawasi secara langsung di lapangan secara terus-menerus, mendorong penggunaan teknologi informasi untuk mengatasi keterbatasan tersebut. Diperlukan sistem teknologi informasi untuk mendeteksi secara cepat dan akurat terhadap pelanggaran tersebut.

Ada beberapa algoritma yang dapat digunakan untuk mendeteksi objek antara lain Faster R-CNN, R-FCN, YOLO, dan SSD. Perbandingan akurasi dan *frame per second* (FPS) yang dilakukan oleh Jonathan Hui [1] menunjukkan bahwa nilai akurasi dari algoritma pendeteksi objek tersebut tidak terpaut jauh perbedaannya. Namun ketika membandingkan kecepatan proses berdasarkan *frame per second*

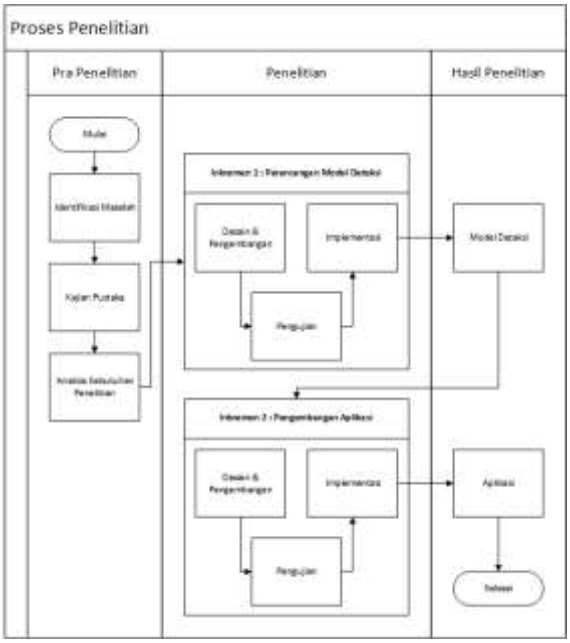
(FPS) terlihat YOLO dapatkan nilai lebih tinggi dibandingkan algoritma lainnya.

Pada penelitian yang dilakukan oleh S. Lu, dkk pada tahun 2019 berjudul "*A Real-time Object Detection Algorithm for Video*" [2] dan penelitian oleh M. Harahap, dkk pada tahun 2019 berjudul "Sistem Cerdas Pemantauan Arus Lalu Lintas dengan YOLO (*You Only Look Once v3*)" [3] mengusulkan algoritma YOLO untuk mendeteksi objek dari video secara *real-time*. Algoritma YOLO yang digunakan pada penelitian [3] mampu mengklasifikasikan kendaraan dengan mAP (*mean Average Precision*) pada CCTV *Fix* yang paling tinggi yaitu 97% sedangkan pada CCTV PTZ adalah 99%. Penelitian [2] mengembangkan YOLO dengan mengurangi lapisan konvolusi sehingga menghasilkan pendeteksian yang lebih cepat secara signifikan.

Oleh karena itu penelitian ini mengusulkan aplikasi untuk mendeteksi sepeda motor yang memasuki jalur khusus sepeda menggunakan algoritma pendeteksian objek YOLO. Penelitian ini akan menggunakan data berupa video *Closed Circuit Television* (CCTV) lalu lintas dari web *Jakarta Smart City*. Video CCTV lalu lintas tersebut akan dibuat area pengawasan virtual pada jalur khusus sepeda. Setelah itu, aplikasi akan mendeteksi sepeda motor yang masuk ke area pengawasan tersebut.

II. METODE PENELITIAN

Proses penelitian dimulai dengan mengamati fenomena yaitu adanya pelanggaran di jalur sepeda. Selanjutnya dilakukan kajian pustaka yang relevan dengan masalah tersebut. Dari referensi yang ada, maka dianalisa kebutuhan penelitian yang akan dilakukan. Hasil analisa mendapatkan algoritma pendeteksi objek YOLO sebagai algoritma yang akan digunakan dalam penelitian ini. Penelitian ini terbagi menjadi dua tahapan yaitu inkremen 1 untuk pembuatan model pendeteksi sepeda motor di jalur sepeda (pelatihan model dan pengujian model) dan inkremen 2 untuk proses pembuatan aplikasi. Proses penelitian dapat digambarkan pada Gambar 1.



Gambar 1 Proses penelitian

YOLO adalah salah satu algoritma pendeteksi objek. YOLO menganggap pendeteksian objek sebagai sebuah masalah regresi tunggal, langsung dari piksel citra ke kotak pembatas dan probabilitas kelas. Sebuah jaringan saraf secara serentak memprediksi banyak kotak pembatas dan probabilitas kelas untuk kotak itu. YOLO menggunakan pendekatan jaringan saraf untuk mendeteksi citra menjadi bagian-bagian / wilayah dan memprediksi setiap kotak pembatas dan probabilitasnya [4].

YOLO versi tiga (YOLOv3) merupakan peningkatan YOLO versi sebelumnya dengan peningkatan pada fitur deteksi multiskala, jaringan pengekstraksi yang lebih kuat, dan perubahan dalam fungsi *loss*. Sehingga YOLOv3 ini dapat mendeteksi lebih banyak objek dari besar sampai kecil [5], [6].

Pendeteksian sepeda motor menggunakan model bawaan YOLO (*pretrain*) menggunakan *dataset* MS COCO dapat mengenali 80 kelas, sehingga jika diimplementasikan pada penelitian ini menyebabkan kesalahan deteksi seperti pada Gambar 2.

Gambar 2 Hasil pendeteksian menggunakan model bawaan YOLO

Oleh karena itu, pada penelitian ini diusulkan YOLOv3 dengan kustomisasi pada data untuk membedakan kendaraan yang diperbolehkan dan dilarang di jalur khusus sepeda, sehingga menghasilkan pendeteksian yang spesifik hanya mendeteksi sepeda dan sepeda motor.

A. Pelatihan Model

Video CCTV didapat dengan mengkonversi video CCTV Bali Tower *Jakarta Smart City* [7] menggunakan aplikasi VLC. Waktu pengambilan video lalu lintas yaitu pukul 08.00 sampai 10.00 pada hari Senin sampai Minggu. Lokasi pengambilan video lalu lintas di Jalan Pemuda Rawamangun Jakarta Timur. CCTV Bali Tower di Jalan Pemuda Rawamangun dipilih karena letak pengambilannya ada di pinggir jalan sehingga jalur khusus sepeda dapat terlihat dengan baik. Durasi pengambilan video berkisar antara tiga sampai tujuh menit. Video CCTV yang dihasilkan berjumlah 101 video dengan masing-masing berkas video terdiri dari 22 FPS. Video CCTV yang digunakan sebagai *dataset* pelatihan berjumlah 84 video. *Dataset* yang digunakan didapatkan dengan membuat berkas citra dari *frame* video CCTV yang telah dikumpulkan. *Frame* video yang diambil merupakan kelipatan sebelas atau setiap 0,5 detik dari video, sehingga menghasilkan 2.029 berkas citra sebagai *dataset*.


Gambar 3 Dataset citra CCTV lalu lintas

Proses pelabelan dilakukan menggunakan aplikasi LabelImg dengan memberi kotak pada objek (sepeda atau sepeda motor) yang ada disetiap citra. Semua label kelas pada masing-masing gambar disimpan pada berkas teks (*.txt) yang berisi indeks kelas, nilai x tengah, nilai y tengah, lebar dan tinggi objek.


Gambar 4 Proses pelabelan objek pada citra

Sebelum melakukan proses pelatihan, ada beberapa berkas yang perlu disiapkan antara lain:

- *Dataset* gambar beserta label kelas dan koordinat objek
- Berkas daftar gambar untuk pelatihan (80% dari *dataset* ditentukan secara acak)
- Berkas daftar gambar untuk validasi (20% dari *dataset* ditentukan secara acak)
- Berkas konfigurasi *layer* YOLO
- Berkas bobot model bawaan *darknet*
- Berkas daftar kelas objek
- Berkas informasi objek yang berisi jumlah kelas, lokasi berkas daftar gambar pelatihan, lokasi berkas daftar gambar validasi, lokasi berkas daftar kelas objek dan lokasi menyimpan model hasil pelatihan.

Proses pelatihan YOLO menggunakan *framework Darknet* yang dapat diunduh atau salin dari repositori GitHub milik AlexeyAB. Setelah diunduh *Darknet* perlu dilakukan konfigurasi dan *build* sebelum digunakan.

Proses pelatihan menggunakan Google Colab dengan akselerasi *hardware* GPU untuk mempercepat proses pelatihan. Karena Google Colab mendukung komputasi menggunakan GPU maka parameter OpenCV dan CUDA pada *darknet* diubah menjadi aktif.

Setiap versi YOLO memiliki berkas konfigurasi dan bobot model yang berbeda. YOLOv3 menggunakan *darknet53.conv.73*, sedangkan YOLOv3 *Tiny* menggunakan *yolov3-tiny.conv.15*.

Pada berkas konfigurasi *layer* YOLO ada beberapa parameter yang perlu diubah yaitu ukuran *batch* menjadi 64, *subdivisions* menjadi 16, *width* menjadi 416, *height* menjadi 416, *max_batches* menjadi 4000, *steps* menjadi 3200 dan 3600, *filters* terakhir sebelum lapisan YOLO menjadi 21, dan jumlah kelas pada lapisan YOLO menjadi 2. Nilai *max_batches*, *steps*, dan *filters* menggunakan ketentuan sebagai berikut:

$$max\ batches\ = kelas\ x\ 2000 \quad (1)$$

Dimana *max batches* merupakan banyaknya iterasi dalam proses pelatihan didapat dari jumlah kelas dikali 2000.

$$steps = 80\%\ x\ max\_batches,\ 90\%\ x\ max\_batches \quad (2)$$

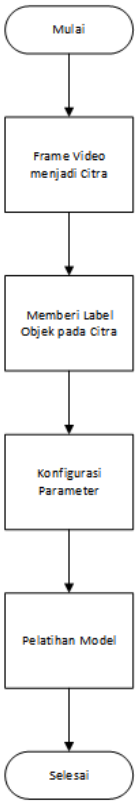
Dimana nilai *steps* didapat dari 80% dan 90% dari *max batch*.

$$filters = (kelas + 5)\ x\ 3 \quad (3)$$

Dimana *filters* merupakan jumlah *filter* yang ada pada lapisan konvolusi terakhir. Nilai *filters* didapat dengan menjumlahkan konstanta lima dengan jumlah kelas lalu dikali konstanta tiga.

Dataset dan berkas pendukungnya diunggah ke Google Drive untuk dilakukan pelatihan model menggunakan Google Colab. Google Colab mendukung perintah-perintah pada Linux sehingga memudahkan untuk melakukan proses *build* dan menjalankan perintah pada *framework darknet*.

Alur pelatihan model dapat digambarkan dengan diagaram alir pada Gambar 5:



Gambar 5 Diagram alir pelatihan model

B. Pengujian Model

Pada tahap ini model YOLO terbaik akan diuji dan digabungkan dengan area deteksi di jalur sepeda. Pembuatan area deteksi di jalur sepeda berpatokan pada marka lalu lintas yang ada pada video CCTV. Area jalur sepeda tersebut terdiri dari empat koordinat yang ditentukan secara manual menggunakan persentase dari lebar dan tinggi dari *frame* video.

Pengujian menggunakan masukan berupa video CCTV lalu lintas. Pada proses ini empat koordinat dari area jalur sepeda akan direpresentasikan menjadi sebuah poligon dan koordinat tengah objek hasil pendeteksian kendaraan dari YOLO akan di representasikan menjadi titik. Objek terdeteksi akan ditandai ketika titik berada di dalam atau menyentuh poligon tersebut. Sedangkan klasifikasi objek berdasarkan pada nilai keyakinan kelas (*confidence*) terbesar dari hasil pendeteksian kendaraan YOLO.

C. Pengembangan Aplikasi

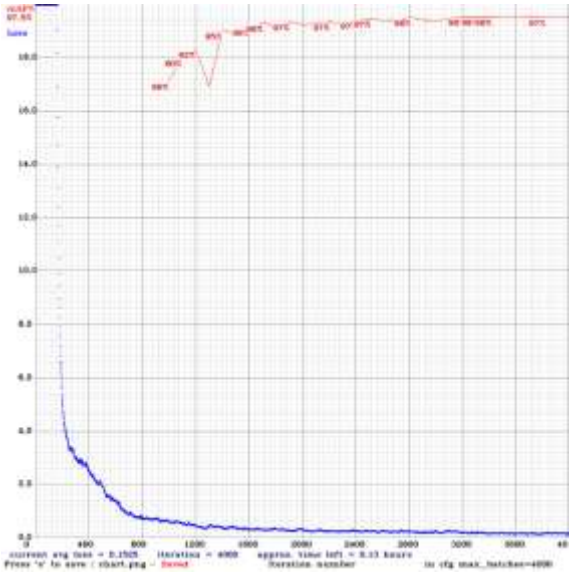
Tahap ini merupakan proses pembuatan antarmuka pengguna untuk memudahkan dalam penggunaan model pendeteksian. Tampilan dari aplikasi dibuat dengan aplikasi Qt Designer, lalu dikonversi menjadi berkas *.py menggunakan modul PyQt5 bernama pyuic5 untuk selanjutnya dimasukkan kode program model yang telah dibuat.

Berdasarkan rancangan yang telah dibuat, aplikasi ini akan mengolah berkas video dan menampilkannya secara *real-time* pada bingkai yang ada di aplikasi. Aplikasi ini dapat membaca dua ekstensi video, yaitu *.mp4 (untuk berkas luring) dan *.m3u8 (untuk berkas daring). Aplikasi akan memvalidasi berkas video CCTV dapat terbaca atau tidak.

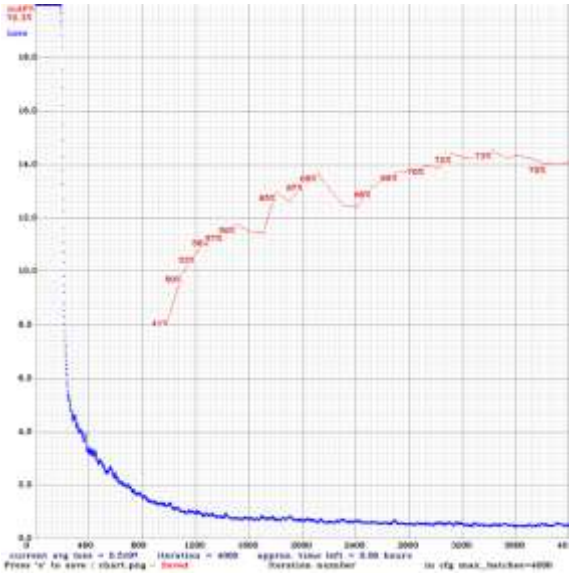
Jika dapat terbaca, selanjutnya model yang telah dibuat akan melakukan deteksi dan ditampilkan pada bingkai yang telah disediakan ketika tombol proses diklik. Menampilkan *frame* hasil deteksi tidak seperti menampilkan citra pada bingkai seperti biasa, melainkan dengan cara melalui *thread*. *Thread* ini akan menangani proses pembaruan citra pada bingkai di aplikasi ketika model mengirimkan sinyal bahwa sebuah *frame* telah selesai proses pendeteksian. Ukuran bingkai pada aplikasi yaitu 720 x 480 sehingga ukuran gambar hasil deteksi yang berukuran 1920 x 1080 perlu dilakukan penyesuaian.

III. HASIL DAN PEMBAHASAN

YOLO versi tiga memiliki dua jenis yaitu YOLOv3 dan YOLOv3 *Tiny*. Hasil pelatihan model YOLOv3 mendapatkan akurasi yang tinggi antara 85% sampai 98%, sedangkan YOLOv3 *Tiny* mendapatkan akurasi antara 41% sampai 73%.



Gambar 6 Grafik Rata-rata Loss dan Akurasi mAP YOLOv3



Gambar 7 Grafik Rata-rata Loss dan Akurasi mAP YOLOv3 Tiny

Tabel 1 merupakan perbandingan mAP dari model yang dicadangkan selama pelatihan. Pencadangan model dilakukan setiap iterasi kelipatan seribu (1000, 2000, 3000, dan 4000), setiap iterasi kelipatan 100 yang terakhir (*last*), setiap didapatkan nilai mAP terbaik (*best*), dan pada iterasi terakhir (*final*).

Tabel 1 mAP dari model yang dicadangkan

	YOLO v3 Tiny	YOLO v3
1000	41,43%	85,63%
2000	63,21%	95,74%
3000	69,03%	96,92%
4000	70,34%	97,46%
Best	72,73%	97,65%
Last	70,34%	97,46%
Final	70,34%	97,46%

Dalam kasus pendeteksian objek pelanggaran lalu lintas, diperlukan kecepatan dan ketepatan deteksi. Dalam pengujian menggunakan Google Colab dengan akselerasi GPU, didapatkan FPS dari YOLOv3 sebesar 13 FPS sedangkan YOLOv3 Tiny sebesar 25 FPS. Untuk mengimplementasikan akurasi terbaik dari model, maka dipilihlah model "*best*" dari kedua jenis YOLOv3 untuk diuji. Pengujian dilakukan secara lokal dengan komputasi CPU karena model tersebut akan diimplementasikan menjadi aplikasi.

Hasil pendeteksian dari model YOLO yang menggunakan data kustomisasi menunjukkan bahwa objek yang terdeteksi hanya ada dua yaitu sepeda dan sepeda motor, sedangkan orang dan mobil tidak terdeteksi.

Gambar 8 Hasil deteksi dengan model kustomisasi YOLO

Pengujian dilakukan dengan menggabungkan area deteksi di jalur sepeda dengan model YOLO sebagai pendeteksian kendaraan. Objek terdeteksi akan ditandai ketika titik berada di dalam atau menyentuh poligon tersebut. Sedangkan klasifikasi objek berdasarkan pada nilai keyakinan kelas (*confidence*) terbesar dari hasil pendeteksian kendaraan YOLO.

Gambar 9 Hasil pembuatan area deteksi di jalur sepeda

Berdasarkan pengujian yang dilakukan terhadap berkas video CCTV dengan menggunakan komputasi CPU didapatkan hasil lama pemrosesan satu *frame* dengan menggunakan model YOLO v3 *Tiny* yaitu berkisar antara 0,25 sampai 0,35 detik, sedangkan dengan menggunakan model YOLO v3 memerlukan waktu berkisar antara 2,0 sampai 3,5 detik. Dengan demikian, model yang akan digunakan yaitu YOLO v3 *Tiny Best* dengan akurasi 72,73% dengan kecepatan pengolahan lebih kurang 3 *frame* per detik.


Gambar 10 Hasil pendeteksian di jalur sepeda

Tahap selanjutnya yaitu mengimplementasikan model YOLO dan area deteksi di jalur sepeda kedalam aplikasi yang telah dibuat menggunakan Qt Designer. Ketika aplikasi mulai dijalankan, model pendeteksian akan dimuat. Direktori atau tautan video yang dimasukkan akan divalidasi untuk memastikan berkas video dapat dibaca. Berdasarkan rancangan yang telah dibuat, aplikasi ini akan mengolah berkas video dan menampilkannya

secara *real-time* pada bingkai yang ada di aplikasi.

Gambar 11 Tampilan aplikasi saat memproses pendeteksian di jalur sepeda

Pengujian aplikasi dilakukan untuk memastikan fungsi utama dalam aplikasi dapat berjalan sesuai harapan. Metode pengujian yang digunakan pada tahap ini adalah metode *black box*. Dari hasil pengujian didapatkan bahwa aplikasi dapat menjalankan dengan baik fungsi-fungsi yang ada.

IV. SIMPULAN

Dari pembahasan tersebut dapat disimpulkan:

1. Penerapan algoritma YOLO dengan kustomisasi data dapat mendeteksi kendaraan secara spesifik untuk sepeda dan sepeda motor saat berada didalam atau menyentuh area deteksi di jalur sepeda.
2. Model YOLOv3 mendapatkan nilai akurasi 85% sampai 98%, sedangkan YOLOv3 Tiny mendapatkan akurasi 41% sampai 73%. Kecepatan proses menggunakan GPU YOLOv3 sebesar 13 FPS dan YOLOv3 Tiny sebesar 25 FPS, sedangkan kecepatan proses menggunakan CPU YOLOv3 sebesar 0,3 FPS dan YOLOv3 Tiny 3 FPS.
3. Model YOLOv3 *Tiny* yang diterapkan pada aplikasi memiliki nilai keakuratan mAP yaitu 72.73% dengan kecepatan lebih kurang tiga frame per detik. YOLOv3 *Tiny* dipilih karena memiliki kecepatan proses yang tinggi namun memiliki akurasi mAP yang rendah.

Dalam pengembangan aplikasi pendeteksian memiliki keterbatasan pada penentuan area jalur sepeda yang masing dilakukan secara manual serta penerapan model YOLO v3 *Tiny* yang memiliki akurasi mAP yang rendah. Oleh karena itu, pengembangan ke depan diharapkan:

1. Menerapkan pendeteksian jalur sepeda yang otomatis serta algoritma pendeteksi objek yang lebih cepat dan akurat.
2. Menambahkan jenis kendaraan yang dapat terdeteksi, seperti otopet, skuter, hoverboard, dan unicycle sebagai kendaraan yang diperbolehkan, serta bajaj dan kendaraan bermotor lainnya sebagai kendaraan yang dilarang.
3. Menggunakan video CCTV lalu lintas dari lokasi dan penyedia CCTV lalu lintas lainnya.
4. Menandai sepeda yang keluar dari jalur sepeda.

DAFTAR RUJUKAN

[1] J. Hui, "Object detection: speed and accuracy comparison (Faster R-CNN, R-FCN, SSD, FPN, RetinaNet and YOLOv3)," *Medium*, 2018. [Daring]. Tersedia pada: https://medium.com/@jonathan_hui/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359. [Diakses: 09-Agu-2020].

[2] S. Lu, B. Wang, H. Wang, L. Chen, M. Linjian, dan X. Zhang, "A Real-time Object Detection Algorithm for Video," *Comput. Electr. Eng.*, vol. 77, hal. 398–408, 2019.

[3] M. Harahap, J. Elfrida, P. Agusman, M. Rafael, R. Abram, dan K. Andrianto, "Sistem Cerdas Pemantauan Arus Lalu Lintas dengan YOLO (You Only Look Once v3)," hal. 367–376, 2019.

[4] J. Redmon, S. Divvala, R. Girshick, dan A. Farhadi, "You Only Look Once: Unified, Real-time Object Detection," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, hal. 779–788, 2016.

[5] J. Redmon dan A. Farhadi, "YOLOv3: An Incremental Improvement," 2018.

[6] E. Y. Li, "Dive Really Deep into YOLO v3: A Beginner's Guide," *Towards Data Science*, 2019. [Daring]. Tersedia pada: https://towardsdatascience.com/dive-really-deep-into-yolo-v3-a-beginners-guide-9e3d2666280e. [Diakses: 06-Jun-2020].

[7] Bali Tower, "CCTV Bali Tower Pemuda Rawamangun kamera 1," *Jakarta Smart City*. [Daring]. Tersedia pada: http://cctv.balitower.co.id/Rawamangun-004-704029_1/tracks-v1/mono.m3u8.