SISTEM DETEKSI *QR CODE* PADA MOBIL BERGERAK DENGAN METODE *FASTER* R-CNN

QR CODE DETECTION SYSTEM ON MOVING CAR USING FASTER R-CNN METHOD

Ruben Haswinsa[1], Iwan Iwut Tritoasmoro[2], Nur Ibrahim[3]
[1,2,3] Prodi S1 Teknik Telekomunikasi, Fakultas Teknik Elektro, Universitas Telkom
[1]rubenhaswinsa@student.telkomuniversity.ac.id, [2]iwaniwut@telkomuniversity.ac.id,
[3]nuribrahim@telkomuniversity.ac.id

Abstrak

Pada tugas akhir ini dirancang sebuah sistem yang dapat membaca *QR Code* pada mobil yang bergerak. Metode yang digunakan pada penelitian tugas akhir ini yaitu menggunakan metode *Faster* R-CNN dan *pre-trained model* ResNet50 sebagai model *object detection* yaitu *QR Code*. Penelitian ini menggunakan 400 data latih berupa citra *QR Code* dan 15 data uji berupa video dengan *frame rate* sebesar 60 fps. Analisis performa sistem dilakukan dengan dua buah parameter pengujian sistem yaitu *loss training* dan akurasi sistem. Pada penelitian tugas akhir ini dapat diketahui bahwa konfigurasi model terbaik terdapat pada model dengan jumlah *step training* 20K dan *batch size* 1. Variasi kecepatan terbaik untuk membaca *QR Code* yaitu pada kecepatan 20 km/jam dan 40 km/jam dengan akurasi sebesar 80%. Sistem ini mendapatkan *frame rate* sebesar 4,9-5,3 fps.

Kata Kunci: *QR Code, Object Detection, Faster R-CNN*

Abstract

In this final project, a system that can read *QR Code* on a moving car is designed. The method used in this research is using the Faster R-CNN method and the pre-trained model ResNet50 as the object detection model, namely *QR Code*. This research use 400 training data in the form of *QR Code* images and 15 testing data in the form of video with a frame rate of 60 fps. System performance analysis is carried out by using two system test parameters, namely loss training and system accuracy. In this final project research, it can be seen that the best model configuration is in a model with a number of training steps 20K and a batch size 1. The best speed variation for reading *QR Code* is at a speed of 20 km/hour and 40 km/hour with an accuracy of 80%. This system gets a frame rate of 4.9-5.3 fps.

Keyword: *QR Code, Object Detection, Faster R-CNN*

1. Pendahuluan

Berkembangnya teknologi informasi membuat segala sesuatu yang biasa dikerjakan manusia dapat ditingkatkan lebih baik lagi dengan memanfaatkan kemajuan teknologi. Saat ini banyak sekali digunakan kamera untuk mengamati kondisi lalu lintas di jalan raya. Banyak hal yang bisa diperoleh dengan penggunaan kamera ini seperti contohnya kondisi lalu lintas di jalan raya. Namun ada suatu masalah yang timbul yaitu untuk mengetahui suatu identitas kendaraan diperlukan seorang pengamat yang akan mengamati setiap plat kendaraan kemudian mencari identitas kendaraan tersebut pada *database* kepolisian. Hal tersebut tentu sangat melelahkan dan tidak efisien. Oleh karena itu, diperlukan sebuah sistem yang dapat mengenali identitas pada kendaraan dengan tujuan untuk mengurangi kerja manusia dan meningkatkan efisiensi dalam proses identifikasi pelanggaran lalu lintas.

Pada penelitian sebelumnya, pengenalan identitas kendaraan dilakukan dengan sistem pengenalan plat nomor otomatis menggunakan *QR Code* oleh [1]. Kelemahan [1] yaitu sistem tidak dapat membaca *QR Code* pada kendaraan yang bergerak sehingga sistem tidak dapat mengenali identitas kendaraan pada kendaraan yang bergerak. Kemudian dalam [2], dibuat sistem pengenalan identitas kendaraan menggunakan *QR Code* pada kendaraan bergerak dengan metode *Contrast Limited Adaptive Histogram Equalization*. Kelemahan [2] yaitu sistem memiliki akurasi yang kecil dalam membaca *QR Code* pada kecepatan diatas 30 km/jam.

Pada tugas akhir ini akan dibuat sebuah sistem pengenalan identitas kendaraan yang dapat membaca *QR Code* pada kendaraan yang bergerak. Diperlukan kamera untuk melakukan proses pengambilan gambar berupa

video pada kendaraan berjenis mobil dan *QR Code* yang terpasang pada mobil. Selanjutnya, video akan diproses menggunakan metode *Faster* R-CNN untuk mendeteksi posisi *QR Code* kemudian *embedding data* dalam *QR Code* dibaca menggunakan aplikasi yang dibuat oleh penulis. Data dalam *QR Code* berisi informasi tentang identitas sebuah mobil seperti nama pemilik, plat nomor kendaraan, dan pajak kendaraan.

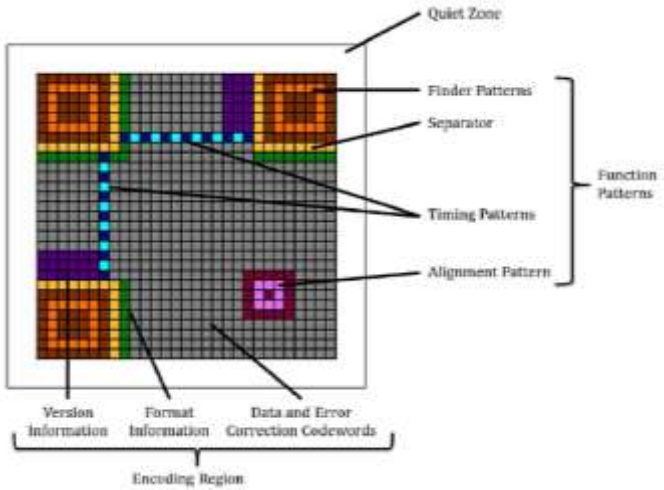
2. Dasar Teori dan Metodologi

2.1 *Object Detection*

Object Detection adalah proses mencari dan mengklasifikasikan objek dalam suatu gambar. *Object detection* berkaitan dengan identifikasi benda dalam dunia nyata seperti orang, binatang, dan benda diam maupun benda bergerak. Algoritma *object detection* menggunakan berbagai aplikasi pengolah gambar untuk mengekstraksi bagian objek yang diinginkan. Hal itu biasa digunakan dalam aplikasi seperti pengambilan gambar, keamanan, medis, dan pertahanan [3].

2.2 *Quick Response Code* (*QR Code*)

QR Code adalah jenis *barcode* matriks atau kode dua dimensi yang dapat menyimpan informasi data dan dikembangkan oleh Denso Wave Corporation di Jepang. QR adalah singkatan dari "*Quick Response*" yang menunjukkan bahwa isi kode harus diterjemahkan dengan sangat cepat dan dengan kecepatan tinggi. *QR Code* terdiri dari modul hitam yang disusun dalam pola persegi pada latar belakang putih [2].



Gambar 1. Struktur *QR Code* [4]

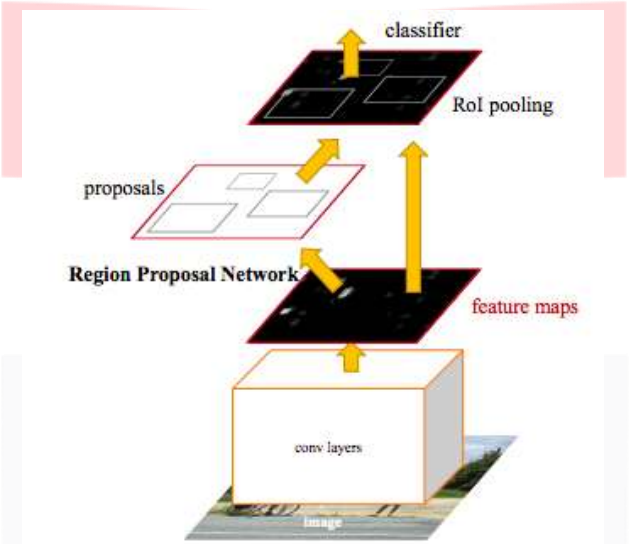
Berikut merupakan penjelasan dari istilah-istilah yang berkenaan dengan gambar 1 yaitu sebagai berikut.

a. *Finder Patterns*: deteksi posisi pola khusus yang terletak di tiga sudut (kiri atas, kanan atas, dan kiri bawah) dari setiap simbol.
b. *Separators*: area lebar satu modul spasi putih antara masing-masing *Finder Patterns* dan *Encoding Region*.
c. *Timing Patterns*: Ada *Timming Patterns*, yaitu *Horizontal Timming Patterns* dan *Vertical Timming Patterns*. Pola ini terdiri dari modul gelap dan terang secara bergantian. *Horizontal Timming Patterns* ditempatkan di baris ke-6 dari *QR Code* di antara *Separators*. *Vertical Timming Patterns* terletak di kolom ke-6 dari *QR Code* di antara *Separators*. Pola-pola ini membantu dalam menentukan kepadatan simbol, modul koordinat dan *version information area*.
d. *Alignment Patterns*: terdiri dari 5 × 5 modul gelap, 3 × 3 modul terang dan satu modul gelap di tengah. *QR Code* versi 2 dan yang lebih besar harus memiliki *Alignment Patterns* dan nomor *Alignment Patterns* tergantung pada versi simbol.
e. *Encoding Region*: berisi *format information*, *version information*, *data and error correction codes*. Untuk *format information*, baris satu modul harus dicadangkan di dekat *finder patterns* dan *version information* bagian kiri-atas, kanan-atas, kiri-bawah, area blok 6×3 di atas *finder patterns* bagian kiri-bawah dan 3×6 blok di sebelah kiri *finder patterns* bagian kanan-atas dicadangkan.
f. *Quiet Zone*: area lebar 4-modul yang tidak berisi data, dan digunakan untuk memastikan bahwa teks atau tanda di sekitarnya tidak boleh menyesatkan data *QR Code*.
g. *Format Information*: informasi tentang *error correction level*.

h. *Version Information*: versi dari sebuah *QR Code*, versi terkecil adalah versi 1 (21 x 21) modul dan versi terbesar adalah versi 40 (177 x 177) modul.

2.3 *Faster* R-CNN

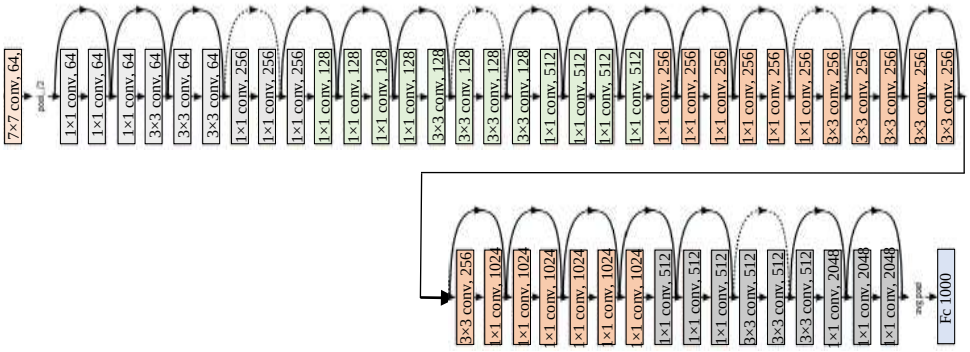
Faster R-CNN terdiri dari dua modul. Modul pertama adalah *Regional Proposal Network*, dan modul kedua adalah detektor Fast R-CNN [5]. Perbedaan utama *Faster* R-CNN dengan Fast R-CNN adalah Fast R-CNN menggunakan *selective search* untuk menghasilkan *region proposal*, sedangkan *Faster* R-CNN menggunakan *Regional Proposal Network*. Biaya dan waktu untuk menghasilkan *region proposal* jauh lebih kecil pada *Regional Proposal Network* dibandingkan dengan *selective search*. Secara singkat, *Regional Proposal Network* mengambil *image feature map* sebagai *input* dan menghasilkan satu set *region proposal* objek kemudian memberi peringkat pada *region box* (disebut *anchor*) dan mengusulkan wilayah yang paling mungkin berisi objek. Lalu, nilai dari *region proposal* masuk ke dalam proses RoI *pooling* untuk menyamakan dimensi citra antar *input* dan *output*. Setelah itu, citra yang telah diproses pada RoI *pooling* menjadi *input* pada *Fast* R-CNN *detector* atau *classifier*.



Gambar 2. Arsitektur *Faster* R-CNN [5]

2.4 *Pre-trained* Model ResNet

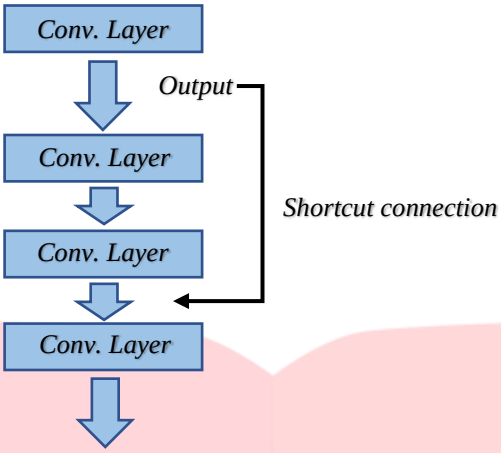
Residual Neural Network atau yang lebih populer disebut sebagai ResNet adalah salah satu model dalam *Deep Neural Network* yang dibuat oleh Kaiming He et al [6]. ResNet memiliki beberapa macam arsitektur, yaitu 18, 34, 50, 101, dan 152 *layer*. Arsitektur ResNet terdiri dari 5 variasi *convolutional layer* dan memiliki 3 variasi ukuran filter yaitu 1×1, 3×3, dan 7×7 serta memiliki kedalaman yang bervariasi pada setiap *convolutional layer*.



Gambar 3. Arsitektur ResNet50 [7]

ResNet memperkenalkan suatu metode baru yang disebut *shortcut connection*. Metode ini digunakan untuk mengatasi *vanishing gradient problem*, yaitu semakin kecilnya nilai *gradient* yang disebabkan oleh semakin

banyak *layer* yang digunakan sehingga akurasi menjadi menurun. Dengan menggunakan *short connection*, maka *output layer* yang baru terhubung dengan *output layer* sebelumnya sehingga meningkatkan akurasi dibandingkan dengan yang tidak menggunakan ResNet.



Gambar 4. Konsep *Shortcut Connection* [7]

2.5 *Dataset*

Dataset yang digunakan pada tugas akhir ini yaitu citra dengan objek *QR Code*. *Dataset* dibagi menjadi tiga, yaitu data latih, data validasi, dan data uji. Rincian *dataset* dapat dilihat pada Tabel 1.

Tabel 1. Rancangan *Dataset*

Kelas	Data Latih	Data Validasi	Data Uji	Kecepatan Mobil
QR Code	400	10	5 video	20 km/jam
			5 video	40 km/jam
			5 video	60 km/jam

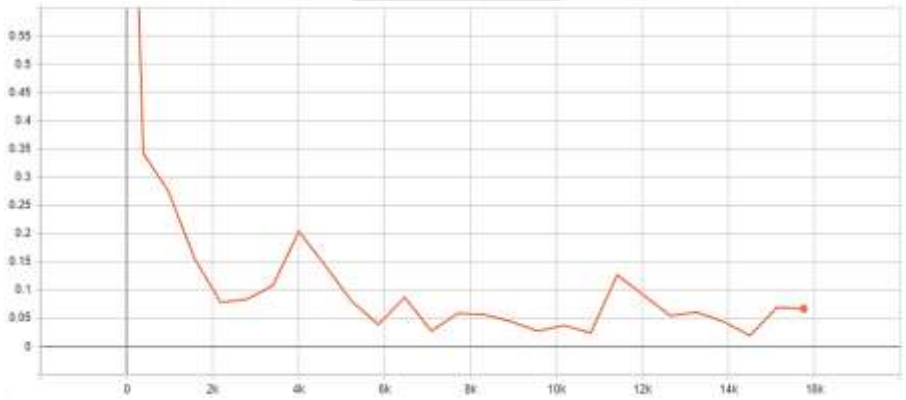
3. Pembahasan

3.1 Pengujian *Loss* Model *Object Detection*

Pada sub-bab ini, penulis akan menyajikan grafik *Loss* dari model *object detection* yang sudah dilatih menggunakan *pre-trained model* ResNet50 pada TensorFlow. Penulis menggunakan sebuah *library* pada TensorFlow untuk menampilkan grafik *Loss* yaitu TensorBoard.

3.1.1 Model 16K

Pada sub-bab ini, penulis akan menyajikan grafik *Loss* dari model *object detection* dengan *step training* 16K dan ukuran *batch size* 1. Grafik *Loss* dapat dilihat pada gambar 5 berikut.

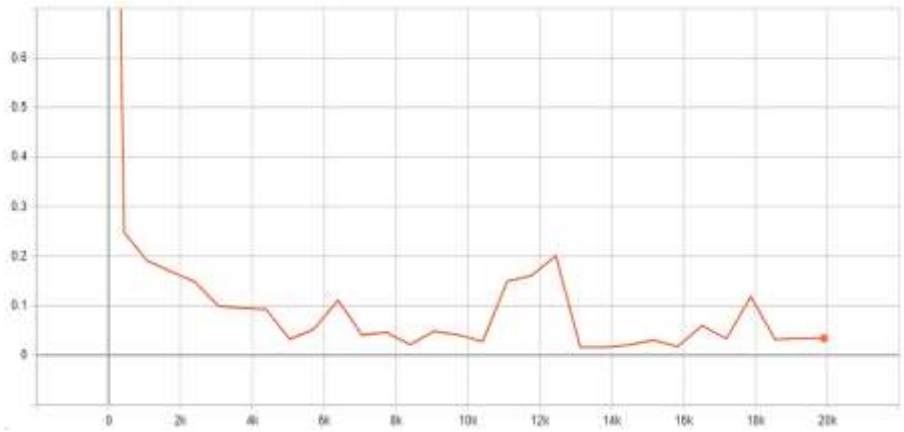


Gambar 5. Grafik *Loss* Model 16K

Pada Gambar 5 dapat dilihat bahwa bentuk grafik dari *Loss* model ini menurun seiring berjalannya waktu mengikuti jumlah *step training* dengan *Loss* yang lebih kecil dari model sebelumnya yaitu sebesar 0,06680 atau 6,680%. Hal tersebut menunjukkan bahwa model yang telah dilatih ini semakin bagus dalam memahami karakteristik setiap citra dalam *dataset* dan memiliki akurasi yang tinggi yaitu sebesar 93,32%.

3.1.2 Model 20K

Pada sub-bab ini, penulis akan menyajikan grafik *Loss* dari model *object detection* dengan *step training* 20K dan ukuran *batch size* 1. Grafik *Loss* dapat dilihat pada gambar 6 berikut.

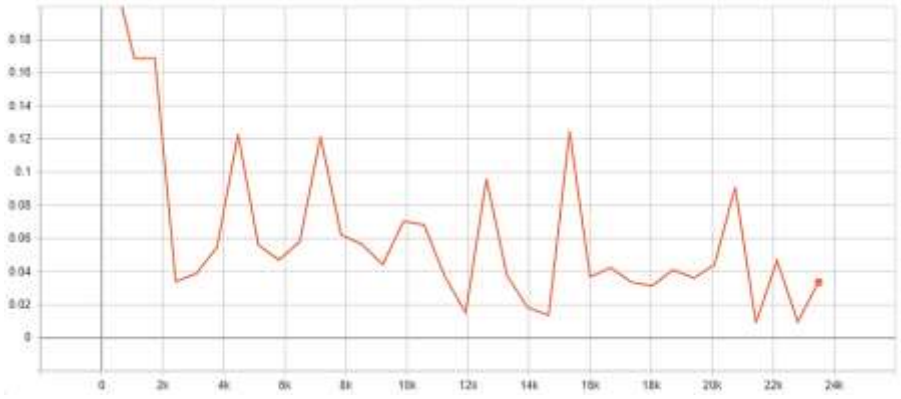


Gambar 6. Grafik *Loss* Model 20K

Pada Gambar 6 dapat dilihat bahwa bentuk grafik dari *Loss* model ini menurun seiring berjalannya waktu mengikuti jumlah *step training* dengan *Loss* sebesar 0,03425 atau 3,425%. Hal tersebut menunjukkan bahwa model yang telah dilatih ini semakin bagus dalam memahami karakteristik setiap citra dalam *dataset* dan memiliki akurasi yang tinggi yaitu sebesar 96,575%.

3.1.3 Model 24K

Pada sub-bab ini, penulis akan menyajikan grafik *Loss* dari model *object detection* dengan *step training* 24K dan ukuran *batch size* 1. Grafik *Loss* dapat dilihat pada gambar 7 berikut.



Gambar 7. Grafik *Loss* Model 24K

Pada Gambar 7 dapat dilihat bahwa bentuk grafik dari *Loss* model ini menurun seiring berjalannya waktu mengikuti jumlah *step training* dengan *Loss* sebesar 0,03364 atau 3,364%. Hal tersebut menunjukkan bahwa model yang telah dilatih ini semakin bagus dalam memahami karakteristik setiap citra dalam *dataset* dan memiliki akurasi yang tinggi yaitu sebesar 96,636%.

3.2 Pengujian Model *Object Detection* Pada Kendaraan

Pada sub-bab ini, penulis akan menyajikan hasil pengujian model *object detection* pada kendaraan bergerak yaitu mobil. Pengujian ini dilakukan dengan tujuan untuk mengetahui akurasi model dan *frame rate* yang dihasilkan. Pengambilan data uji berupa video dengan durasi 3 detik dan memiliki *frame rate* sebesar 60 fps.

3.2.1 Pengujian Model 16K

Pada sub-bab ini, penulis akan menyajikan hasil dari pengujian dari model *object detection* dengan jumlah *step training* yaitu 16K dan *batch size* 1. Hasil pengujian model ini dapat dilihat pada gambar berikut.

Gambar 8. Hasil Pengujian Model 16K

Pada Gambar 8 dapat dilihat bahwa model sistem sukses dijalankan pada program yang telah dibuat oleh penulis. Hasil yang didapat yaitu *QR Code* dapat terdeteksi dengan dengan akurasi sebesar 100% dan menghasilkan *frame rate* sebesar 5,3 fps dari *frame rate* video yang sebenarnya yaitu 60 fps.

3.2.2 Pengujian Model 20K

Pada sub-bab ini, penulis akan menyajikan hasil dari pengujian dari model *object detection* dengan jumlah *step training* yaitu 20K dan *batch size* 1. Hasil pengujian model ini dapat dilihat pada gambar berikut.

Gambar 9. Hasil Pengujian Model 20K

Pada Gambar 9 dapat dilihat bahwa model sistem sukses dijalankan pada program yang telah dibuat oleh penulis. Hasil yang didapat yaitu *QR Code* dapat terdeteksi dengan dengan akurasi sebesar 100% dan menghasilkan *frame rate* sebesar 5,2 fps dari *frame rate* video yang sebenarnya yaitu 60 fps. Model ini

merupakan model terbaik yang penulis dapatkan karena dapat mendeteksi *QR Code* dan model ini termasuk dalam *good fitting*. Model ini juga akan digunakan untuk analisis pengujian sistem pada sub-bab 3.3.

3.2.3 Pengujian Model 24K

Pada sub-bab ini, penulis akan menyajikan hasil dari pengujian dari model sistem dengan jumlah *step training* yaitu 24K dan *batch size* 1. Hasil pengujian model ini dapat dilihat pada gambar berikut.

Gambar 10. Hasil Pengujian Model 24K

Pada Gambar 10 dapat dilihat bahwa model sistem sukses dijalankan pada program yang telah dibuat oleh penulis. Hasil yang didapat yaitu *QR Code* dapat terdeteksi dengan dengan akurasi sebesar 100% dan menghasilkan *frame rate* sebesar 4,9 fps dari *frame rate* video yang sebenarnya yaitu 60 fps. Model ini merupakan model terburuk yang penulis dapatkan karena meskipun dapat mendeteksi QR dengan baik namun model ini juga dapat mendeteksi objek lain selain *QR Code* yaitu pohon sehingga model ini mengalami *overfitting*.

3.3 Pengujian Sistem Berdasarkan Kecepatan Mobil

Pada sub-bab ini, penulis akan menyajikan hasil pengujian sistem berdasarkan kecepatan mobil yang dikelompokkan menjadi 3 variasi kecepatan mobil yaitu 20 km/jam, 40 km/jam, dan 60 km/jam. Hasil pengujian sistem pada sub-bab ini dapat dilihat pada beberapa tabel berikut.

Tabel 2. Pengujian Sistem dengan Kecepatan 20 km/jam

Video ke-	Jarak Objek	Jumlah *QR Code*	Hasil	Akurasi
1	1 meter	1	1	100 %
2	1 meter	1	1	100 %
3	1 meter	1	1	100 %
4	1 meter	1	1	100 %
5	1 meter	1	0	0%

Berdasarkan pada tabel 2 di atas, dapat dilihat bahwa dari 5 video yang diuji, penulis mendapatkan 4 video yang dapat membaca *QR Code*. Oleh karena itu penulis mendapatkan akurasi rata-rata untuk terbacanya *QR Code* sebesar 80%. Pada tabel di bawah ini merupakan tabel hasil pengujian sistem dengan kecepatan 40 km/jam.

Tabel 3. Pengujian Sistem dengan Kecepatan 40 km/jam

Video ke-	Jarak Objek	Jumlah *QR Code*	Hasil	Akurasi
1	1 meter	1	0	0 %
2	1 meter	1	1	100 %
3	1 meter	1	1	100 %
4	1 meter	1	1	100 %
5	1 meter	1	1	100%

Berdasarkan pada tabel 3 di atas, dapat dilihat bahwa dari 5 video yang diuji , penulis 4 video yang dapat membaca *QR Code*. Oleh karena itu penulis mendapatkan akurasi rata-rata untuk terbacanya *QR Code* sebesar 80%. Pada di bawah ini merupakan tabel hasil pengujian sistem dengan kecepatan 60 km/jam.

Tabel 4. Pengujian Sistem dengan Kecepatan 60 km/jam

Video ke-	Jarak Objek	Jumlah *QR Code*	Hasil	Akurasi
1	1 meter	1	1	100 %
2	1 meter	1	0	0 %
3	1 meter	1	0	0 %
4	1 meter	1	1	100 %
5	1 meter	1	1	100%

Berdasarkan pada tabel 4 di atas, dapat dilihat bahwa dari 5 video yang diuji, penulis mendapatkan hanya 3 video yang dapat membaca *QR Code*. Oleh karena itu penulis mendapatkan akurasi rata-rata untuk terbacanya *QR Code* sebesar 60%.

4. Kesimpulan dan Saran

4.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan oleh penulis, yaitu *training* model *object detection*, pengujian model *object detection*, dan analisis pengujian sistem, maka dapat disimpulkan sebagai berikut:

1. Konfigurasi *hyperparameter* pada *step training* yaitu 16K, 20K, dan 24K serta *batch size* bernilai 1 dapat melatih data latih dengan model dari *pre-trained model* Resnet50 sehingga menghasilkan nilai *loss training* masing-masing sebesar 6,68%, 3,425%, dan 3,364% serta menghasilkan *frame rate* sebesar 4,9 - 5,3 fps pada data uji berupa video.
2. Konfigurasi model terbaik yang penulis dapatkan yaitu model 20K dengan *batch size* 1 serta model terburuk yang penulis dapatkan yaitu model 24K dengan *batch size* 1 karena model ini mengalami overfitting.
3. Variasi kecepatan terbaik untuk membaca *QR Code* yang penulis dapatkan yaitu pada kecepatan 20 km/jam & 40 km/jam dengan akurasi sebesar 80%.

4.2 Saran

Berdasarkan penelitian dan kesimpulan di atas, ada beberapa saran yang ingin penulis sampaikan untuk pengembangan penelitian berikutnya, yaitu:

1. Menggunakan metode *deep learning* yang lebih ringan seperti *Single Shot Detector* (SSD) dan *You Only Look Once* (YOLO) dengan tujuan untuk menghasilkan *frame rate* yang lebih besar.
2. Menggunakan perangkat keras dengan spesifikasi yang lebih tinggi dengan tujuan agar proses *training data* dan proses pengujian data uji dapat dilakukan pada perangkat yang sama serta dapat meningkatkan performa sistem.
3. Diperlukan konfigurasi *hyperparameter* pada *step training* dan *batch size* yang lebih bervariasi lagi untuk menghasilkan model yang optimal dan akurasi yang tinggi.

Reference:

[1] P. T. Bhupendra Moharil, Vijayendra Ghadge, Chaitanya Gokhale, “An Efficient Approach for Automatic Number Plate Recognition System Using Quick Response Codes,” vol. 3, no. 5, pp. 5108–5115, 2012.

[2] W. Hogpracha and S. Vongpradhip, “Recognition System for QR Code on Moving Car,” in *10th International Conference on Computer Science and Education, ICCSE 2015*, 2015, pp. 14–15.

[3] A. Raghunandan, Mohana, P. Raghav, and H. V. R. Aradhya, “Object Detection Algorithms for Video Surveillance Applications,” in *Proceedings of the 2018 IEEE International Conference on Communication and Signal Processing, ICCSP 2018*, 2018, p. 563.

[4] S. Tiwari, “An Introduction to QR Code Technology,” in *Proceedings - 2016 15th International Conference on Information Technology, ICIT 2016*, 2017, pp. 40–41.

[5] R. Gavrilescu, C. Zet, C. Fosalau, M. Skoczylas, and D. Cotovanu, “Faster R-CNN:an Approach to Real-Time Object Detection,” in *EPE 2018 - Proceedings of the 2018 10th International Conference and Expositions on Electrical And Power Engineering*, 2018, pp. 166–167.

[6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, pp. 773–774.

[7] J. B. Tege, S. A. Wibowo, and Y. Purwanto, “Desain dan Implementasi Estimasi Jarak Kendaraan Bermotor Berbasis Vision Menggunakan Raspberry Pi Dengan Metode Faster R-CNN,” Telkom University, 2020.