Pengenalan Karakter pada Plat Nomor Indonesia dengan Tilt Correction dan Metode Faster R-CNN

Kevin Nyoto Susanto, Kartika Gunadi, Endang Setyati
Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
E-Mail: kevinnyoto20@gmail.com, kgunadi@petra.ac.id, endang@stts.edu

ABSTRAK

Pertumbuhan jumlah kendaraan di Negara Indonesia sangat pesat dan hal tersebut menyebabkan timbulnya banyak masalah, diantaranya adalah masalah sistem keamanan parkir dan kontrol akses kendaraan bermotor. Dengan berkembangnya teknologi, plat nomor yang merupakan identitas dari suatu kendaraan dapat dideteksi secara otomatis oleh sistem dengan bantuan *Digital Image Processing* dan *Artificial Neural Network*.

Penelitian ini menggunakan metode *Canny Edge Detection* untuk mendeteksi objek plat nomor dalam gambar. Sebelum melakukan klasifikasi karakter pada plat nomor, distorsi perspektif yang ada pada gambar dapat dihilangkan menggunakan metode *Planar Homography*. *Faster R-CNN* digunakan untuk mendeteksi posisi mobil pada gambar dan mengklasifikasi karakter yang ada pada plat nomor.

Program hasil dari penelitian ini akan mendeteksi karakter plat nomor dan warna mobil pada gambar. Hasil pengujian dari dataset peneliti, akurasi deteksi karakter pada plat nomor mencapai 82,14% dan akurasi deteksi warna mobil mencapai 78,54%.

Kata Kunci: *Optical Character Recognition, Tilt Correction, Canny Edge Detection, Faster R-CNN*

ABSTRACT

The growth of the number of vehicles in Indonesia is very rapid and has caused many problems, such as the problem of parking security systems and access control of vehicles. With the development of technology, vehicles' license plates can be detected automatically by a system with the help of Digital Image Processing and Artificial Neural Network.

This study uses the Canny Edge Detection method to detect license plate objects in the image. Before classifying characters on license plates, the perspective distortion in the image can be removed using the Planar Homography method. Faster R-CNN is used to detect the position of the car in the image and classify the characters on the license plate

The results of this research program will detect the character of license plates and the color of car in the image. From the test results using the researcher's dataset, the accuracy of detection of the characters in the license plate reached 82.14% and the accuracy of detection of the cars' colors reached 78.54%.

Keywords: *Optical Character Recognition, Tilt Correction, Canny Edge Detection, Faster R-CNN*

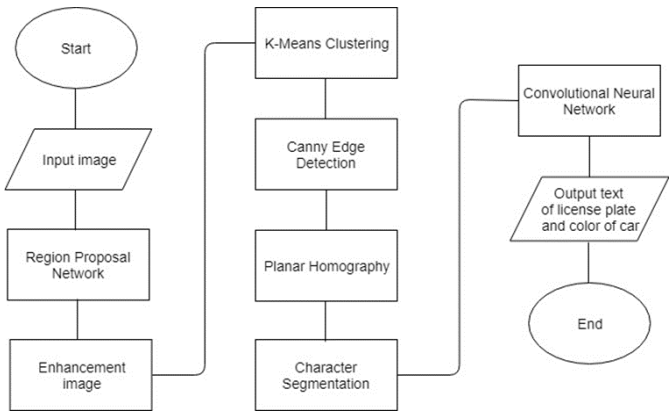
1. PENDAHULUAN

Jumlah kendaraan bermotor di negara Indonesia mengalami pertumbuhan tiap tahunnya. Berdasarkan data dari Badan Pusat Statistik, pada tahun 2016 jumlah kendaraan bermotor mencapai 129 juta dengan dominasi jumlah sepeda motor mencapai 81%. [1] Oleh karena itu mulai muncul banyak masalah, diantaranya adalah sistem keamanan kendaraan bermotor.

Plat nomor kendaraan bermotor merupakan hal yang sangat penting dalam sistem keamanan karena plat nomor merupakan identitas dari suatu kendaraan. Dengan berkembangnya teknologi saat ini, beberapa alat seperti kamera digunakan untuk melakukan deteksi plat nomor secara otomatis. Deteksi lokasi plat nomor tersebut dapat menggunakan *digital image processing* (DIP) dan pengenalan karakter pada plat nomor dapat menggunakan *artificial neural network* (ANN) sehingga dapat dilakukan secara otomatis

Penelitian mengenai pengenalan karakter pada plat nomor kendaraan terus mengalami perkembangan. Dimulai dengan penelitian untuk deteksi lokasi plat nomor sampai dengan pengenalan karakter pada plat nomor. Dalam pengenalan karakter pada citra (*Optical Character Recognition*) masalah yang umum terjadi adalah adanya citra karakter yang mengalami distorsi perspektif sehingga dapat mengurangi akurasi pengenalan karakter. Untuk itu diperlukan perbaikan posisi citra yang mengalami kemiringan atau distorsi perspektif agar akurasi pengenalan karakter semakin meningkat. [9]

Sistem pengenalan plat nomor ini menerima input berupa gambar mobil dengan plat nomor Indonesia dan dilakukan pencarian lokasi plat nomor pada gambar. Setelah itu dilakukan *tilt correction* pada gambar dan segmentasi tiap karakter. Tiap karakter yang telah disegmentasi ini akan dikenali dan pada akhirnya akan menghasilkan teks plat nomor dan warna mobil pada gambar tersebut. Gambar 1 merupakan flowchart arsitektur sistem dari program.



Gambar 1. Arsitektur Sistem

2. DASAR TEORI

2.1 Canny Edge Detector

Terdapat beberapa metode untuk mendeteksi tepian pada gambar, antara lain *Prewitt*, *Robert Cross Operator*, *Sobel Operator*, dan *Canny Edge Detector*. Dari beberapa metode tersebut, *Canny Edge Detection* dapat mendeteksi tepian pada gambar lebih baik daripada metode lainnya dan dapat menghilangkan *noise* pada gambar [3].

Untuk itu, dilakukan empat langkah untuk mengimplementasikan *Canny Edge Detector* ini, yaitu:

a. Menggunakan *Gaussian smoothing (Gaussian Averanging Operator)*

b. Menggunakan *Sobel operator*

c. Menggunakan *non-maximal suppression*

d. Lakukan *threshold* dengan menggunakan *Hysteresis Thresholding* untuk menghubungkan titik-titik tepian. [6]

2.2 Planar Homography

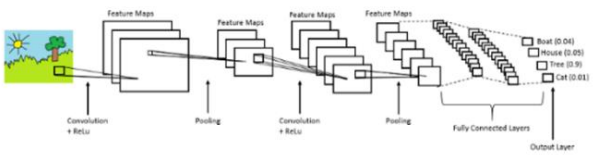
Pada dasarnya, *planar homography* menggunakan penyederhanaan matriks dari matriks kamera. Posisi titik koordinat dari bidang dapat ditentukan oleh (X, Y, 0, 1) karena tidak ada nilai Z (sama dengan nol). *Homography matrix* digunakan untuk mengubah perspektif kamera terhadap gambar secara virtual. Dengan matriks ini, gambar dapat diambil dari perspektif lain sesuai dengan *homography matrix* [10]. Persamaan 1 merupakan persamaan *homography*

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \rightarrow \begin{cases} x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1} \\ y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1} \end{cases} \quad (1)$$

2.3 Convolutional Neural Network

Salah satu metode yang penting dalam *deep learning* adalah *Convolutional Neural Network* (CNN). CNN dirancang khusus untuk pengenalan dan klasifikasi gambar. CNN memiliki beberapa lapisan (layer) yang mengekstrak informasi dari gambar dan menentukan klasifikasi dari gambar tersebut [2].

Terdapat tiga layer yang membentuk CNN yaitu *Convolutional Layer*, *Pooling Layer* dan *Fully Connected Layer*. Secara umum, *layer* tersebut diatur dengan cara menumpuk beberapa *convolutional layer* diikuti dengan *pooling layer*. Pola ini diulangi hingga input gambar semakin mengecil. *Layer* terakhir adalah *fully connected layer* yang memberikan hasil output akhir, berupa skor klasifikasi [5]. Gambar 2 merupakan visualisasi dari proses CNN.



Gambar 2. Proses *Convolutional Neural Network* [7]

2.4 Faster R-CNN

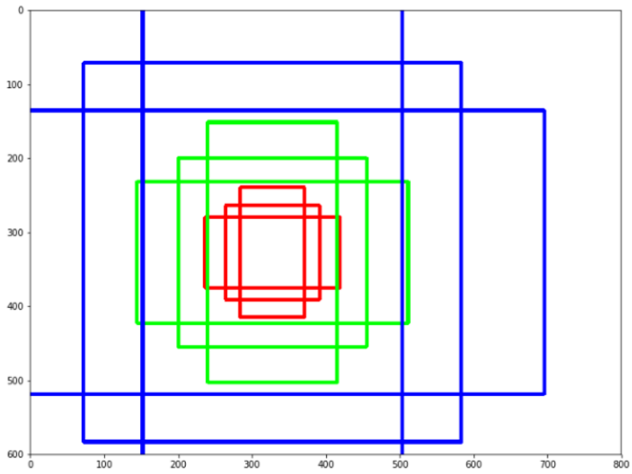
Metode CNN mengalami perkembangan secara terus-menerus dan akhirnya muncul metode yang merupakan improvisasi dari CNN seperti *Region-based* CNN (R-CNN), *Fast* R-CNN dan *Faster* R-CNN. Diantara perkembangan metode tersebut, *Faster* R-CNN memiliki tingkat akurasi dan kecepatan yang lebih tinggi dari perkembangan metode CNN lainnya.

Kontribusi *Faster* R-CNN pada metode sebelumnya adalah mengganti *selective search* dengan *region proposal network* (RPN) yang meningkatkan kecepatan secara signifikan untuk pencarian *region* yang mengandung objek.

2.5 Region Proposal Network

Region Proposal Network (RPN) merupakan salah satu metode pengganti *selective search* dalam menemukan objek pada gambar. RPN menyusun *region boxes* yang disebut *anchors* untuk mencari *region* yang diperkirakan mengandung objek [5].

Anchor memiliki peran penting dalam *Faster* R-CNN. Dalam konfigurasi *default Faster* R-CNN, terdapat 9 *anchors* di satu posisi pada gambar. Sebagai contoh, Gambar 3 menunjukkan visualisasi 9 *anchors* pada posisi koordinat (320, 320) dengan ukuran gambar sebesar (600,800) [4].

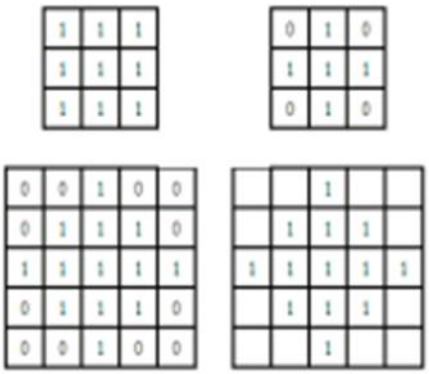


Gambar 3. Visualisasi 9 anchors pada posisi (320,320) di gambar (600, 800) [4]

Output dari RPN adalah sekelompok kotak yang akan diklasifikasi oleh *regressor* untuk dideteksi apakah ada objek didalamnya. RPN memprediksi kemungkinan *anchor* dapat menjadi *background* atau *foreground* [4].

2.6 Morphological Function

Tujuan dari *morphological function* ini adalah untuk menghilangkan *noise* dari gambar setelah dilakukan operasi seperti *thresholding*. *Morphological function* bisa sangat berguna dalam segmentasi gambar karena berhubungan langsung dengan ekstraksi bentuk dari sebuah gambar. Untuk melakukan *morphological function* ini, dibutuhkan *structuring element*. *Structuring element* adalah matriks biner (berisi 1 atau 0) dengan ukuran yang sangat kecil. Biasanya, *structuring elemet* ini direpresentasikan sebagai matriks persegi dengan dimensi ganjil. Disini, *structuring element* berperan sebagai filter untuk konvolusi terhadap gambar. Gambar 4 merupakan visualisasi dari *structuring element*. *Morphological Function* terbagi menjadi beberapa jenis, diantaranya adalah *dilation, erosion, opening, closing*.



Gambar 4. *Structuring Element* [8]

2.7 K-Means Clustering

K-Means Clustering merupakan salah satu metode pengelompokan yang sudah populer. Dalam metode ini, jumlah *cluster* (k) yang juga disebut *centoid* telah ditentukan sebelum analisis pada data dilakukan. *Centeroid* ini berperan sebagai titik tengah dari *cluster*. Posisi awal dari *centeroid* ini adalah *random*, lalu dilakukan iterasi posisi *centeroid* akan berubah sesuai dengan rata-rata jarak dari data yang ada pada *cluster*. Iterasi dilakukan sampai posisi dari *centeroid* tidak berubah.

2.8 Levenshtein Distance

Levenshtein Distance adalah jumlah minimal dari perubahan karakter (penyisipan, penghapusan, penggantian) antara 2 kalimat. Sebagai contoh, untuk mengganti kata "kata" dan "kota" hanya perlu mengganti "a" dengan "o", dengan demikian maka hanya perlu 1 penggantian. Maka *Levenshtein Distance* dari "kata" dan "kota" adalah 1. Karena itu, *Levenshtein Distance* dapat digunakan juga sebagai metode untuk mengukur perbedaan dari 2 kalimat. *Levenshtein Distance* juga dikenal sebagai *Edit Distance*.

2.9 Delta-E

Digunakan untuk mengukur tingkat perbedaan antara 2 warna, berdasarkan koordinat L*a*b*. Jika nilai *Delta-E* semakin besar, maka perbedaan antara 2 warna semakin besar, dan sebaliknya jika nilai *Delta-E* semakin kecil maka perbedaan antara 2 warna semakin kecil.

3. DESAIN SISTEM

3.1 Region Proposal Network

Region Proposal Network digunakan untuk mendeteksi objek pada gambar. Pada proses ini, objek yang akan dideteksi adalah mobil yang terdapat pada gambar. Untuk dapat mendeteksi mobil pada gambar, perlu dilakukan *training* pada dataset terlebih dahulu. *Training* untuk RPN ini membutuhkan data gambar beserta *groundtruth* yang berisikan koordinat dari mobil pada gambar. Hasil dari *training* tersebut adalah *model* yang akan digunakan untuk mendeteksi mobil yang ada pada gambar saat *testing*.

3.2 Enhancement Image

Enhancement Image perlu dilakukan pada input gambar sebelum dilakukan proses selanjutnya. Proses ini melingkupi konversi gambar ke *grayscale* dan pengaplikasian *morphological function* pada gambar.

3.3 K-Means Clustering

K-Means Clustering ini diaplikasikan pada hasil gambar dari proses *Region Proposal Network* (RPN) yang merupakan gambar mobil pada gambar. Nilai *k* yang berbeda-beda telah dicoba dan hasil yang optimal adalah ketika nilai $k = 3$, sehingga digunakan nilai tersebut untuk mengaplikasikan *k-means clustering*. Setelah dilakukan *k-means clustering*, maka hasil tersebut diplotkan dalam bentuk histogram dan warna yang diambil adalah warna dengan frekuensi tertinggi yang merupakan warna dominan dari mobil.

3.4 Canny Edge Detection

Sebelum dilakukan *Canny Edge Detection*, diaplikasikan *Gaussian Blur* agar *noise* pada gambar berkurang. Untuk mengaplikasikan *Canny Edge Detection*, dibutuhkan 2 nilai *threshold* (*upper threshold* dan *lower threshold*), dan pada penelitian ini digunakan nilai *upper threshold* berdasarkan nilai *Otsu Thresholding* dari gambar tersebut × 2 dan *lower threshold* berdasarkan nilai *Otsu Thresholding* dari gambar tersebut.

3.5 Planar Homography

Dari hasil *Canny Edge Detection*, dilakukan pencarian kontur yang ada pada gambar. Tentu saja, sangat banyak kontur yang terdeteksi, sehingga harus dilakukan filter pada kontur yang ditemukan. Kontur adalah garis tepi dari objek pada gambar.

Seleksi kontur yang dilakukan akan menghasilkan koordinat dari plat nomor pada input gambar. Setelah mengetahui koordinat dari plat nomor pada gambar, dilakukan perhitungan untuk mencari *Homography Matrix* supaya kemiringan pada gambar dapat dikoreksi

Untuk mendapatkan *Homography Matrix*, dibutuhkan 4 titik koordinat asal dan 4 titik koordinat destinasi perubahan *pixel*. 4 titik koordinat destinasi ini merupakan titik koordinat plat nomor pada gambar nantinya akan digunakan untuk memotong gambar sehingga menghasilkan gambar plat nomor saja. Setelah mendapatkan *Homography Matrix*, maka matriks tersebut akan dikonvolusi pada gambar.

3.6 Character Segmentation

Gambar plat nomor yang telah didapat harus disegmentasi (dipotong) per karakter supaya dapat diklasifikasi satu per satu. Segmentasi dilakukan dengan mencari kontur dari karakter pada plat nomor. Setelah didapat kontur karakter, maka karakter-karakter yang ada akan disegmentasi, dikonversi ke *binary image* dan di *resize* dengan ukuran 28 × 28 *pixel* sebelum dilakukan *Convolutional Neural Network*.

3.7 Convolutional Neural Network

Sebelum dapat mengklasifikasi gambar karakter, perlu dilakukan training pada model CNN (*Convolutional Neural Network*). *Training* dilakukan dengan jumlah *dataset* sebanyak 2000 karakter dan juga dilakukan validasi hasil training dengan jumlah *dataset* sebanyak 400 karakter. Seluruh *dataset* berukuran 28 × 28 *pixel* dan merupakan *binary image*.

Karena CNN merupakan *supervised training*, *dataset training* dan validasi ini tidak hanya terdiri dari gambar karakter saja, selain itu setiap gambar karakter tersebut diberikan label yang menjadi penanda dari gambar karakter tersebut. Label ini merupakan angka dari 0 sampai 35 sesuai dengan jumlah *class* yang akan diklasifikasi. *Class* yang akan diklasifikasi adalah huruf kapital (A – Z) dan angka (0 – 9).

4. HASIL EKSPERIMEN

Implementasi sistem dilakukan pada computer dengan spesifikasi RAM 4GB, DDR3, HDD 500 GB, CPU Core i5-2450M, GPU NVIDIA GeForce GT 630M, dan OS Microsoft Windows 10 Professional.

Implementasi pengkodean sistem, menggunakan Bahasa pemrograman Python dengan versi 3.5.0. *Framework* yang digunakan untuk sistem ini adalah Flask *Micro-Framework.* Adapun beberapa *library* yang mendukung sistem ini adalah OpenCV, Matplotlib, Scikit-learn, Numpy dan SQLite3.

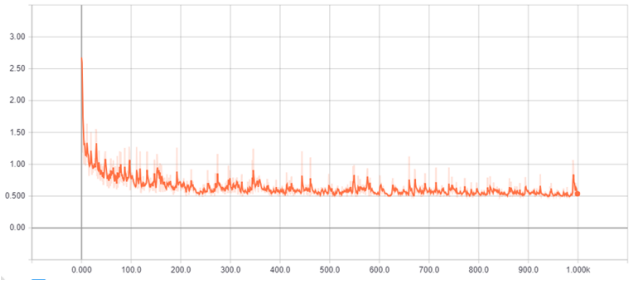
4.1 Pengujian terhadap hasil training

4.1.1 Training Region Proposal Network

Model pertama pada *training* RPN dilakukan dengan jumlah *step* sebesar 1000, jumlah data adalah 535 gambar mobil, jumlah data per *batch* adalah 256 gambar dan *learning rate* sebesar 0.001. Tabel 1 dan Gambar 5 merupakan visualisasi perbandingan *loss* tiap *step* pada training RPN *model* pertama.

Tabel 1. Perbandingan *loss* tiap *step* pada *training* RPN

Step	*Loss*
200	0.78
400	0.62
600	0.58
800	0.56
1000	0.49



Gambar 5. Perbandingan *loss* tiap *step* pada *training* RPN *model* pertama

Model kedua *training* RPN dilakukan dengan jumlah *step* yang berbeda, yaitu sebesar 100 dengan *learning rate* sebesar 0,001. Grafik perbandingan *step* dan *loss* untuk model kedua dapat dilihat pada Gambar 6.

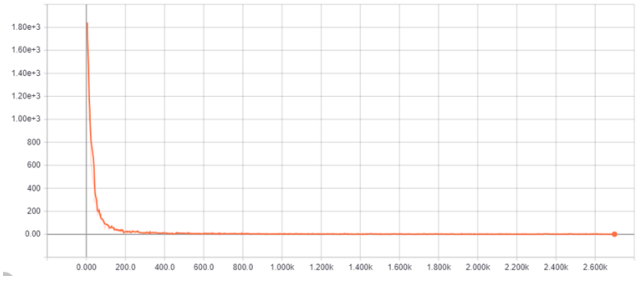
Gambar 6. Perbandingan *loss* tiap *step* pada *training* RPN *model* kedua

4.1.2 Training Convolutional Neural Network

Model pertama pada *training* CNN dilakukan dengan jumlah *step* sebesar 2700, jumlah data adalah 3364 gambar karakter, jumlah data per *batch* adalah 100 dan *learning rate* sebesar 0.0001. Sama halnya seperti *training* RPN, nilai *loss* pada *training* CNN semakin kecil dengan bertambahnya *step* yang dilakukan. Tabel 2 dan Gambar 7 merupakan visualisasi perbandingan *loss* tiap *step* pada *training* CNN *model* pertama.

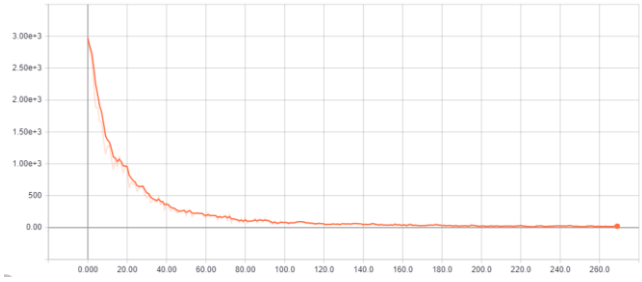
Tabel 2. Perbandingan *loss* tiap *step* pada *training* CNN

Step	*Loss*
300	15.96
600	10.52
900	2.91
1200	3.24
1500	4.81
1800	3.76
2100	1.04
2400	1.61
2700	0.58



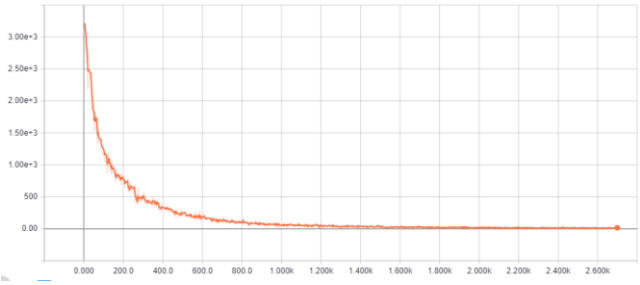
Gambar 7. Perbandingan *loss* tiap *step* pada *training* CNN *model* pertama

Model kedua *training* CNN dilakukan dengan *step* sebesar 270 dan *learning rate* sebesar 0,0001. Grafik perbandingan *step* dan *loss* untuk *model* kedua dapat dilihat pada Gambar 8.



Gambar 8. Perbandingan *loss* tiap *step* pada *training* CNN *model* kedua

Model ketiga *training* CNN dilakukan dengan *step* sebesar 2700 dan *learning rate* sebesar 0,00001. Grafik perbandingan *step* dan *loss* untuk *model* ketiga dapat dilihat pada Gambar 9.



Gambar 9. Perbandingan *loss* tiap *step* pada *training* CNN *model* ketiga

4.2 Pengujian terhadap rangkaian sistem program

Pengujian terhadap rangkaian sistem program dilakukan pada Gambar 10.

Gambar 10. Input gambar

4.2.1 Region Proposal Network

Gambar 11 merupakan hasil dari proses Region Proposal Network pada Gambar 10. Dapat dilihat dari Gambar 11, telah didapatkan posisi mobil pada gambar. Ketika posisi mobil sudah didapatkan maka akan mempermudah proses deteksi plat nomor.

Gambar 11. Hasil dari proses *Region Proposal Network*

4.2.2 Konversi ke Grayscale Image

Gambar 12 merupakan hasil konversi ke grayscale dari Gambar 11.

Gambar 12. Hasil dari konversi ke Grayscale

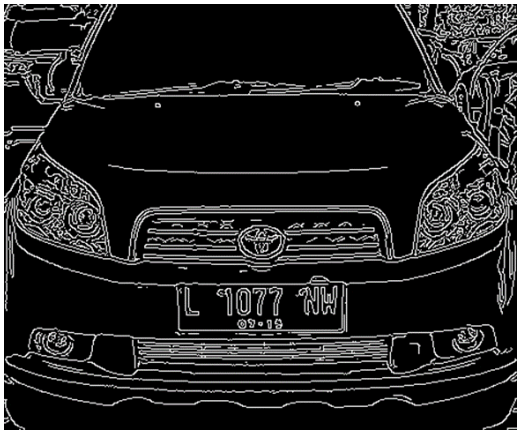
4.2.3 Morphological Function

Gambar 13 merupakan hasil proses *Morphological Function* dari Gambar 12.

Gambar 13. Hasil dari proses Morphological Function

4.2.4 Canny Edge Detection

Gambar 14 merupakan hasil proses *Canny Edge Detection* dari Gambar 13.



Gambar 14. Hasil dari proses Canny Edge Detection

4.2.5 Seleksi Kontur

Gambar 15. Hasil dari proses Seleksi Kontur

Gambar 15 merupakan hasil proses seleksi kontur dari Gambar 14. Tujuan dari seleksi kontur tersebut adalah untuk mendapatkan posisi plat nomor pada gambar.

4.2.6 Planar Homography

Gambar 16 merupakan hasil proses tilt correction dengan menggunakan metode *Planar Homography* dari Gambar 15.

Gambar 16. Hasil dari proses *Planar Homography*

4.2.7 Convolutional Neural Network

Dari proses Planar Homography, posisi plat nomor telah diketahui dan dapat dilakukan segmentasi karakter. Masing-masing karakter akan diproses pada Convolutional Neural Network untuk diklasifikasi. Gambar 17 merupakan hasil segmentasi dan klasifikasi dari Gambar 16.

Gambar 17. Hasil dari segmentasi karakter dan klasifikasi CNN

4.2.8 K-Means Clustering

Untuk mendapatkan warna dominan pada gambar mobil, maka Gambar 11 harus dipotong dengan posisi tertentu dibagian atas sehingga hanya didapatkan *body* mobil saja. Gambar 18 merupakan hasil pemotongan dari Gambar 11 dan Gambar 19 merupakan hasil perhitungan warna dominan dari Gambar 18.

Gambar 18. Hasil pemotongan dari gambar mobil

Gambar 19. Hasil deteksi warna dominan dari gambar mobil

4.2.9 Perhitungan akurasi

Perhitungan akurasi dibagi menjadi 2 yaitu perhitungan akurasi untuk deteksi plat nomor dan warna mobil. Pengujian pertama telah dilakukan pada seluruh data *testing* dengan jumlah data sebanyak 535 gambar, dan untuk pengenalan karakter pada plat nomor rata-rata akurasinya mencapai 82,14% sedangkan untuk deteksi warna mobil rata-rata akurasinya mencapai 78,54%. Tabel 3 merupakan sample dari hasil perhitungan akurasi pengenalan karakter pada plat nomor, dan Tabel 4 merupakan sample dari hasil perhitungan akurasi deteksi warna mobil.

Tabel 3. Sample dari hasil pengenalan karakter pada plat nomor beserta akurasi

Plat nomor	Hasil klasifikasi	Akurasi
L1077NW	L1077NW	100 %
L1239EX	1239G	57.14 %
N1792BT	N17979T	71.43 %
W1290XS	W1290XS	100 %
W1784BZ	D1784BZ	85.71 %

Tabel 4. Sample dari hasil deteksi warna mobil beserta akurasi

Warna mobil	Hasil klasifikasi	Akurasi
(174, 179, 187)	(174, 179, 187)	100 %
(21, 22, 22)	(21, 22, 22)	100 %
(70, 40, 40)	(18, 16, 17)	83.75 %
(160, 160, 160)	(23, 23, 24)	41.98 %
(95, 95, 95)	(38, 37, 36)	74.28 %

Pengujian kedua dilakukan *testing* pada *dataset* menggunakan *model* CNN dengan jumlah *step* sebesar 270 dan *learning rate* sebesar 0,0001 serta model RPN dengan jumlah *step* sebesar 1000 dan *learning rate* sebesar 0,001 menghasilkan akurasi untuk pengenalan karakter pada plat nomor sebesar 71,75% sedangkan akurasi untuk deteksi warna mobil pada gambar mencapai 78,01%.

Pengujian ketiga dilakukan *testing* pada *dataset* menggunakan *model* CNN dengan jumlah *step* sebesar 2700 dan *learning rate* sebesar 0,0001 serta model RPN dengan jumlah *step* sebesar 100 dan *learning rate* sebesar 0,001 menghasilkan akurasi untuk pengenalan karakter pada plat nomor sebesar 81,57% sedangkan akurasi untuk deteksi warna mobil pada gambar mencapai 74,6%.

Pengujian keempat dilakukan *testing* pada *dataset* menggunakan *model* CNN dengan jumlah *step* sebesar 2700 dan *learning rate* sebesar 0,00001 serta model RPN dengan jumlah *step* sebesar 1000 dan *learning rate* sebesar 0,001 menghasilkan akurasi untuk pengenalan karakter pada plat nomor sebesar 72,3% sedangkan akurasi untuk deteksi warna mobil pada gambar mencapai 78,55%.

5. KESIMPULAN

Setelah dilakukan perancangan sistem, pengimplementasian, dan pengujian terhadap sistem yang telah dibuat, dapat ditarik kesimpulan sebagai berikut:

- Mendapatkan posisi mobil dan klasifikasi tiap karakter plat nomor pada gambar dapat dilakukan menggunakan metode *Faster R-CNN* .
- *Canny Edge Detection* dapat digunakan untuk mendeteksi posisi plat nomor dan segmentasi karakter pada plat nomor.
- *Planar Homography* dapat digunakan sebagai metode untuk *tilt correction* pada gambar.
- Akurasi untuk pengenalan karakter pada plat nomor mencapai 82,14% dan untuk deteksi warna mobil mencapai 78,54% dengan menggunakan *dataset*.

6. DAFTAR PUSTAKA

[1] Badan Pusat Statistik. 2018, May 30. Perkembangan Jumlah Kendaraan Bermotor Menurut Jenis, 1949-2016. Retrieved from Badan Pusat Statistik: https://www.bps.go.id/linkTableDinamis/view/id/1133

[2] Ciaburro, G., & Venkateswaran, B. 2017. *Neural Networks with R.* Birmingham: Packt Publishing Ltd.

[3] Das, S. 2016. Comparison of Various Edge Detection Technique. *International Journal of Signal Processing, Image Processing and Pattern Recognition Vol.9. No.2*, 143-158.

[4] Gao, H. 2017, September 27. *Faster R-CNN Explained*. Retrieved from A Medium Corporation Web Site: https://medium.com/@smallfishbigsea/faster-r-cnn-explained-864d4fb7e3f8

[5] Jørgensen, H. 2017. Automatic License Plate Recognition using Deep Learning Techniques.

[6] Nixon, M., & Aguado, A. 2002. *Feature Extraction and Image Processing.* Oxford: Newnes.

[7] Prabhu. 2018, March 4. *Understanding of Convolutional Neural Network (CNN)—Deep Learning*. Retrieved from Medium: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

[8] Srisha, R., & Khan, A. 2013. Morphological Operations for Image Processing : Understanding and its Applications. *NCVSComs-13* (pp. 17-19). India: Vignan's University.

[9] Tedjojuwono, S. M. 2015. Fast Performance Indonesian Automated License Plate Recognition Algorithm Using Interconnected Image Segmentation. ICSIIT 2015, 11-12.

[10] Zhang, W., Li, X., & Ma, X. 2008. Perspective Correction Method for Chinese Document Images. *International Symposium on Intelligent Information Technology Application Workshops*, 467-470.