Jurnal Elektro
p-ISSN: 1979-9780
e-ISSN: 2746-4288

Jurnal Elektro Vol.16 No.2 Oktober 2023
journal homepage: https://ejournal.atmajaya.ac.id/index.php/JTE

Implementasi Convolutional Neural Network Pada Robot Tikus Micromouse Untuk Navigasi Labirin Dengan Pendeteksi Garis Menggunakan Raspberry Pi

Devin Pangestu[1], Ferry Rippun Gideon Manalu[2*]

[1,2]Program Studi Teknik Elektro, Fakultas Teknik
Universitas Katolik Atma Jaya Indonesia, Jakarta 12930, Indonesia

Article Info

Article history:

Received
02 12 2023

Accepted
14 12 2023

Keywords:
Convolutional Neural Network, Line Detector, Micromouse, Raspberry Pi

Abstract

Micromouse is a science and technology project competition that began in the late 1970s. Simply put, the micromouse competition is an event where robotic mice can navigate a 16x16 maze without human assistance. Convolutional Neural Network, also known as CNN, is an extension of the Artificial Neural Network (ANN) designed to process two-dimensional data and is often used for classification, recognition, and prediction tasks. In the conducted research, a robotic mouse was designed using Raspberry Pi to control servo motor movements and utilize a camera to detect maze obstacles, such as intersections, through image scanning. The robotic mouse was tested using a CNN architecture model on TensorFlow, assisted by a line detection algorithm. The testing results indicated that the CNN model architecture showed signs of overfitting (the model learned features too well from the training data) with an accuracy of 96.53%, and the model accuracy evaluation result was 97.97% on the designed dataset. As part of the testing, a maze size of 5x5 was used. The testing of the Convolutional Neural Network model and line detection algorithm, when applied to the robotic mouse, demonstrated that the mouse could navigate according to the path. However, the recorded data was influenced by external light reflecting off the surface of the path, causing the line detection algorithm to perceive it as an obstacle, leading the robot to turn back..

Info Artikel

Histori Artikel:

Diterima:
02 12 2023

Disetujui:
14 12 2023

Kata Kunci:
Convolutional Neural Network, Micromouse, Pendeteksi Garis, Raspberry Pi

Abstrak

Micromouse adalah kompetisi proyek sains dan teknologi yang dimulai pada akhir 1970-an. Secara sederhana, kompetisi micromouse adalah acara di mana robot tikus bisa memecahkan labirin 16x16 tanpa bantuan manusia. Convolutional Neural Network yang juga dikenal dengan CNN adalah pengembangan dari Artificial Neural Network (ANN) untuk mengolah data dua dimensi dan sering digunakan untuk permasalahan klasifikasi, pengenalan, dan prediksi. Dalam penelitian yang dilakukan, robot tikus dirancang menggunakan Raspberry Pi untuk mengontrol pergerakan motor servo beserta penggunaan kamera untuk mendeteksi rintangan labirin berupa persimpangan melalui pemindaian gambar. Robot tikus diuji menggunakan perancangan arsitektur model CNN pada robot tikus menggunakan Tensorflow dan dengan bantuan algoritme pendeteksi garis. Hasil pengujian diperoleh arsitektur model CNN menunjukkan indikasi overfitting (model terlalu mempelajari fitur pada data pelatihan) dengan hasil akurasi sebesar 96.53% dan hasil evaluasi akurasi model sebesar 97.97% terhadap dataset yang dirancang. Sebagai pengujian, ukuran labirin yang digunakan hanya berukuran 5x5. Hasil uji model Convolutional Neural Network dan algoritme pendeteksi garis ketika diaplikasikan pada robot tikus dihasilkan robot tikus mampu berjalan sesuai dengan jalur, tetapi perekaman yang dihasilkan karena terpengaruh oleh cahaya eksternal yang memantul melalui permukaan jalur, algoritme pendeteksi garis menganggapnya sebagai halangan dalam jalur yang mengharuskan robot untuk berputar balik.

*Corresponding author. Ferry Rippun Gideon Manalu
Email address: ferry.rippun@atmajaya.ac.id

1. LATAR BELAKANG

Micromouse adalah kompetisi proyek sains dan teknologi yang dimulai pada akhir 1970-an. "*Principles and Practice of Micromouse*" adalah sebuah kursus praktik inovasi yang dirancang berdasarkan kompetisi *micromouse* bagi mahasiswa tingkat akhir. Berpartisipasi dalam proses pengembangan *micromouse* dapat melatih kemampuan praktik inovatif mahasiswa. Pengetahuan teoritis mengenai *micromouse* diterapkan dalam praktik perancangan robot tikus. Selain itu, perancangan *micromouse* mengandung berbagai disiplin ilmu. Secara sederhana, kompetisi *micromouse* adalah acara di mana robot tikus bisa memecahkan labirin 16x16 tanpa bantuan manusia. Robot tikus perlu menyelesaikan beberapa tugas seperti pencarian labirin, menghafal jalan dan menyelesaikan labirin [1].

Pada penelitian yang berjudul "*Spatial Based Deep Learning Autonomous Wheel Robot Using CNN"* [2], metode *CNN* menggunakan arsitektur *Resnet-34* dan perancangan robot menggunakan *NVIDIA Jetson Nano* sebagai otak utama dalam pemrosesan gambar *CNN* untuk mendeteksi rintangan berupa jalur yang harus dilalui robot. Penelitian ini berfokus pada pengembangan arsitektur dan prosedur pelatihan yang tepat menggunakan *Convolutional Neural Network (CNN)* untuk mengkategorikan bentuk simpang dan dengan bantuan algoritme pendeteksi garis yang membuat robot tikus dapat menelusuri labirin yang dihadapi.

2. DASAR TEORI

2.1 Kompetisi Micromouse

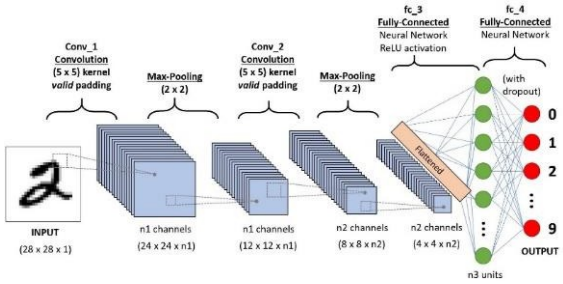
Micromouse adalah kompetisi proyek sains dan teknologi yang dimulai pada akhir 1970-an. "*Principles and Practice of Micromouse*" adalah sebuah kursus praktik inovasi yang dirancang berdasarkan kompetisi *micromouse* bagi mahasiswa tingkat akhir. Berpartisipasi dalam proses pengembangan robot tikus dapat melatih kemampuan praktik inovatif mahasiswa. Selain itu, perancangan robot tikus mengandung berbagai disiplin ilmu baik dari bidang robotika, sensor maupun elektronika.

Gambar 1. Labirin 16x16

Secara sederhana, kompetisi *micromouse* adalah acara di mana robot tikus bisa memecahkan labirin 16x16 sesuai dengan Gambar 1. Robot tikus perlu menyelesaikan beberapa tugas seperti pencarian labirin, menghafal jalan dan menyelesaikan labirin. Robot tikus yang dirancang harus memiliki bentuk dimensi tidak lebih dari 25 cm x 25 cm dan tinggi tidak dibatasi. Robot tikus juga tidak dibatasi untuk menggunakan metode pemecahan jalur apapun selama tidak mengikuti metode dan rancangan dari pemenang kompetisi sebelumnya [3].

2.2 *Convolutional Neural Network* (CNN)

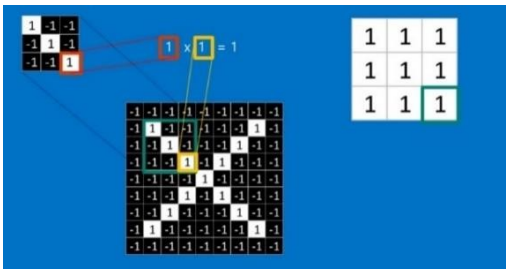
Convolutional Neural Network yang juga dikenal dengan CNN adalah pengembangan dari *Artificial Neural Network* (ANN) untuk mengolah data dua dimensi dan sering digunakan untuk permasalahan klasifikasi, pengenalan, dan prediksi pada objek gambar.



Gambar 2. Layer pada Convolutional Neural Network

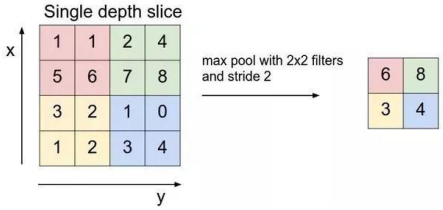
Pada CNN, data yang dikirimkan ke dalam jaringan adalah data dua dimensi, sehingga bobot (weight) pada hubungan antar neuron akan berbeda. Operasi komputasi linear yang dilakukan pada CNN yaitu akan digunakan operasi konvolusi sesuai dengan arsitektur model pelatihan yang digunakan pada setiap lapisan. Tidak seperti ANN pada umumnya, neuron pada lapisan CNN diatur dalam 3 dimensi yaitu: lebar, tinggi, dan kedalaman seperti yang dijelaskan pada Gambar 2.

Layer setelah *layer input* dibagi menjadi beberapa bagian yaitu *convolution layer*, *pooling layer*, dan *fully connected layer*. Pada *convolution layer*, dilakukan operasi perkalian *filter* konvolusi yang diinisialisasi secara acak. Konvolusi adalah istilah matematis yang berarti mengaplikasikan sebuah fungsi pada *output* fungsi lain secara berulang. Gambar 3 menunjukkan cara operasi *filter* dalam proses konvolusi.



Gambar 3. Contoh operasi konvolusi pada convolution layer

Setelah melewati proses pengolahan konvolusi, output dari proses konvolusi akan menjadi input bagi lapisan pooling yang bertujuan melakukan proses reduksi ukuran data citra. *Pooling* dilakukan untuk memperkecil ukuran fitur dan tetap menyimpan informasi-informasi pentingnya.



Gambar 4. Operasi max pooling untuk mereduksi gambar

Salah satu metode reduksi yang banyak digunakan untuk CNN adalah *max pooling*, dimana *max pooling* membagi matriks gambar ke dalam beberapa bagian kecil dan memilih nilai paling besar di dalamnya sebagai representasi matriks gambar baru yang bentuk dimensinya sudah direduksi. Proses akhir dari pelatihan data adalah *fully connected layer*. *Output* dari *layer* ini adalah *array* dengan jumlah kelas *array* yang telah melalui proses konvolusi dan *pooling layer* menggunakan model yang dipilih [2]. Terdapat fitur-fitur penting dalam perhitungan nilai bobot pada arsitektur *Convolutional Neural Network* diantaranya fungsi aktivasi, fungsi kehilangan dan optimisasi.

2.3 Pendeteksi Garis Jalur

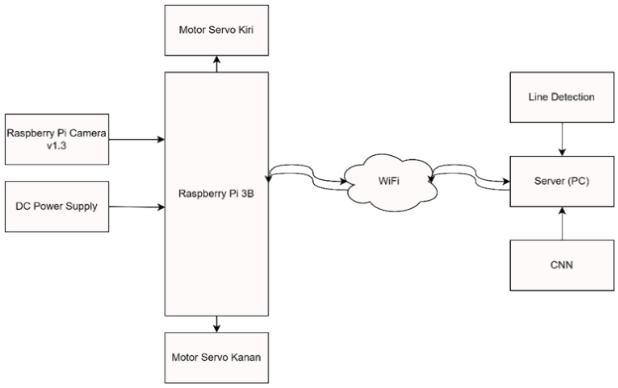
Deteksi jalur merupakan salah satu fitur yang dimiliki oleh pustaka *OpenCV*. Fitur yang dikembangkan ini lebih popular daripada pengembangan deteksi jalur berbasis jaringan saraf tiruan dikarenakan memiliki kekurangan dalam hal proses kecepatan, memori yang digunakan serta menggunakan gambar dengan dimensi besar [3].Tidak seperti lingkungan lalu lintas yang nyata, jalur

pada labirin memiliki posisi yang tetap dengan tipe yang sama yaitu hitam dan putih sehingga informasi berupa garis yang diproses juga lebih sedikit. Oleh karena itu, pendeteksian jalur menggunakan garis yang dikembangkan oleh pustaka *OpenCV* menjadi pilihan dalam penelitian ini karena dapat dideteksi dengan cepat dan akurat tanpa mempengaruhi kinerja *Convolutional Neural Network.*

3. PERANCANGAN SISTEM

Bab ini berisi penjelasan tentang perancangan sistem untuk dapat menjalankan program penelusuran labirin. Dimulai dari metodologi penelitian, cara kerja sistem keseluruhan, perancangan perangkat keras robot tikus, perancangan program model sistem robot tikus (*Raspberry Pi*, *Raspberry Pi Camera Module*), dan perancangan perangkat lunak pengumpul data, dan konfigurasi komunikasi antara komputer eksternal dan *Raspberry Pi*.

3.1 Diagram Blok Sistem



Gambar 5. Blok Diagram Perancangan Robot Tikus

Berdasarkan gambar 5, Sistem dirancang dengan *Raspberry Pi* sebagai media pengambilan gambar yang kemudian diproses menjadi sebuah gambar masukan. Masukan yang berupa gambar akan dialihkan menuju komputer melalui jaringan lokal, kemudian komputer akan melakukan proses prediksi bentuk persimpangan oleh sebuah model CNN yang telah dilatih dan bantuan algoritme pendeteksi garis. Keluaran dari hasil prediksi akan memberikan perintah kepada *Raspberry Pi* untuk menggerakkan motor servo dalam menjelajahi sebuah labirin.

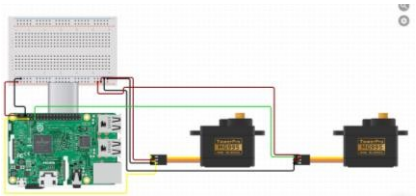
3.2 Persiapan Modul Komputer dan *Raspberry Pi*

Perancangan perangkat lunak robot penjelajah labirin terdiri dari beberapa proses konfigurasi dan pembuatan program pada terminal. Konfigurasi yang ada yaitu: *Raspberry Pi OS*, tampilan *Desktop Raspberry Pi*, *remote access Raspberry Pi*, instalasi *OpenCV*, dan kebutuhan modul lainnya. Instalasi *Raspberry Pi OS* dapat mengikuti petunjuk yang tertera pada https://www.raspberrypi.com/software/ dan dibutuhkan sebuah *SD card* untuk menyimpan data sistem operasi. Kebutuhan modul untuk seluruh program python yaitu:

1. pandas==1.3.5 (Komputer)
2. pandas-profiling==3.1.0 (Komputer)
3. tensorflow==2.4.0 (Komputer)
4. tensorflow-gpu==1.15.0 (Komputer)
5. Keras==2.2.4 (Komputer)
6. Keras-Applications==1.0.8 (Komputer)
7. Keras-Preprocessing==1.1.2 (Komputer)
8. matplotlib==3.5.1 (Komputer)
9. matplotlib-inline==0.1.3 (Komputer)
10. numpy==1.19.5 (*Raspberry Pi* & Komputer)
11. pygame==2.1.2 (*Raspberry Pi*)

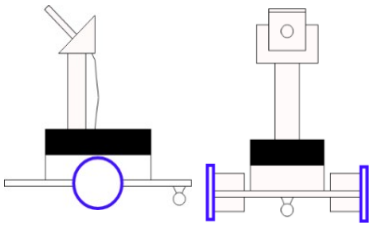
12. pyserial==3.5 (*Raspberry Pi*)
13. scikit-learn==0.23.2 (Komputer)
14. scikit-plot==0.3.7 (Komputer)
15. opencv-contrib-python==4.2.0 (*Raspberry Pi* & Komputer)

3.3 *Perancangan Perangkat Keras*

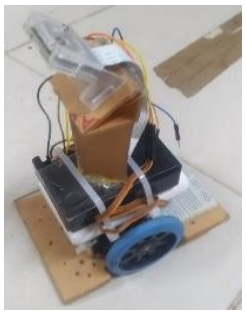


Gambar 6. Rangkaian skematik robot tikus

Berdasarkan Gambar 6, Perangkat keras yang digunakan yaitu *Raspberry Pi* yang bertindak sebagai *client* dan sebuah komputer eksternal yang bertindak sebagai server. *Raspberry Pi* dirancang sebagai perekam dan akan dikirimkan menuju komputer eksternal melalui *WiFi* dan hasil rekaman akan diproses pada komputer eksternal, kemudian hasil dari proses akan dikirim kembali menuju *Raspberry Pi* melalui *WiFi* untuk menggerakkan motor servo. Pada *Raspberry Pi* akan digunakan pin GPIO (*General Purpose Input Output*) yang akan bertindak sebagai pin output untuk mengirimkan sinyal hasil prediksi untuk menghidupkan dan menggerakkan motor servo. Pin yang digunakan antara lain pin 5V, GND, 17 dan 18. Sketsa awal serta wujud dari robot tikus yang dirancang diperlihatkan pada Gambar 7 dan Gambar 8.



Gambar 7. Sketsa robot tikus yang dirancang

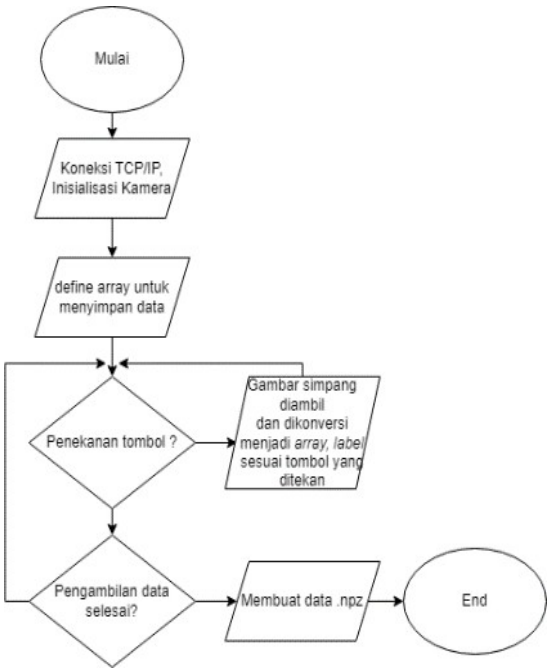


Gambar 8. Wujud robot tikus yang dirancang

3.4 Perancangan Perangkat Lunak

Perancangan perangkat lunak robot penjelajah labirin terdiri dari pembuatan program pengumpul data, program pelatihan data dan pengaplikasian hasil pelatihan data. Pertama-tama, dibutuhkan sebuah

program untuk mengambil data gambar jalur agar pengambilan data lebih efisien dan efektif dalam segi ukuran gambar maupun saat menjalankan program. Diagram alir program pengambilan data gambar ditunjukkan pada Gambar 9.

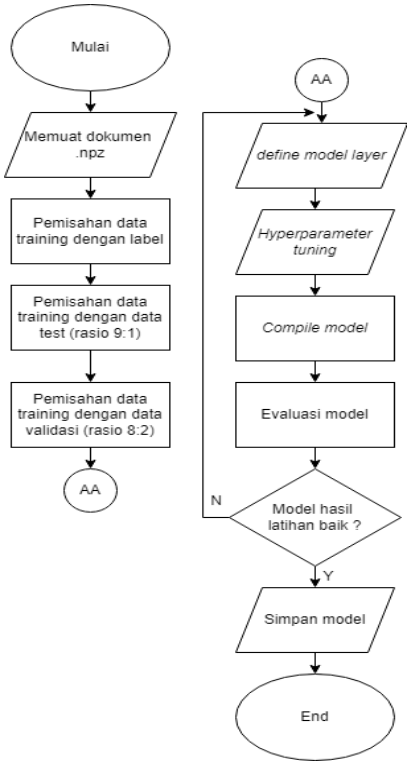


Gambar 9. Diagram Alir Program Pengumpul Data

Untuk melakukan pengambilan data, program untuk mengaktifkan kamera dijalankan terlebih dahulu, kemudian program pengambilan data gambar dijalankan. Perekaman dilakukan segera setelah program pengambilan data gambar dijalankan. Saat *Raspberry Pi* merespon perintah yang diberikan *user*, proses perekaman yang terjadi saat perintah diberi akan ditangkap menjadi gambar dan diubah menjadi wujud angka dalam bentuk *array* satu dimensi. Pembentukan *array* satu dimensi disertai perintah yang diberikan oleh *user* saat gambar ditangkap. Proses ini terus berlanjut hingga *user* menghentikan program yang dijalankan secara manual dan sekumpulan *array* satu dimensi yang telah ditangkap akan dikompilasi menjadi satu dokumen *numpy*.

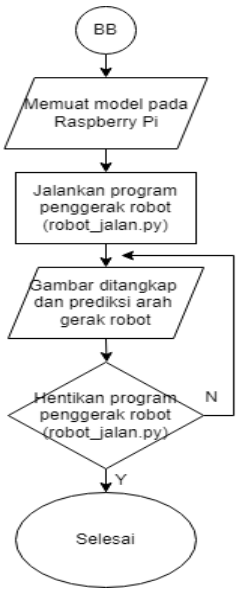
Dokumen *numpy* akan dijadikan sebagai data pelatihan untuk dijadikan sebual model yang akan diimplementasikan pada robot penjelajah labirin. Data yang telah diambil akan disimpan dalam bentuk *array* dan dikonversi menjadi dokumen *numpy* dengan tujuan meminimalisir ukuran *file size*. Setelah itu dokumen *numpy* akan menjalani pelatihan melalui program pelatihan data menggunakan metode *Convolutional Neural Network*. Diagram alir dari proses pelatihan ditunjukkan pada Gambar 10.

Proses pelatihan akan dijalankan hingga dibentuk menjadi sebuah model yang digunakan untuk mengklasifikasikan cara berjalan robot melalui sebuah gambar yang ditangkap oleh kamera *Raspberry Pi*. Model yang disimpan berupa dokumen dengan format ".h5" yang berisi algoritme hasil pelatihan data yang berisi jumlah neuron (*node*) sesuai dengan rancangan arsitektur model, bobot (*weight*) beserta bias. Model yang dirancang menjadi dokumen dan telah dievaluasi akan digunakan sebagai media untuk memprediksi gambar jalur yang ditangkap oleh kamera *Raspberry Pi* dan memerintahkan robot tikus untuk bergerak sesuai dengan prediksi gambar jalur. Diagram alir dari program penggerak robot tikus ditunjukkan pada Gambar 11.



Gambar 10. Diagram Alir Program Pelatihan Data

kkkk



Gambar 11. Diagram Alir Program Eksekusi Robot

Diagram alir dimulai dengan melakukan konfigurasi model yang telah dirancang dengan cara mendefinisikan sebuah model menjadi sebuah proses yang harus dilewati dan menghasilkan *output* berupa hasil prediksi. Setelah konfigurasi model telah dilakukan, dilanjutkan dengan membuat koneksi secara TCP/IP menggunakan IP Address antara komputer eksternal dengan *Raspberry Pi* melalui *WiFi* agar komputer eksternal dapat menerima gambar yang ditangkap oleh kamera *Raspberry Pi* dan memberi

perintah berupa hasil prediksi yang akan dikirimkan kembali ke *Raspberry Pi*. Setelah itu kamera akan mengirim data gambar secara terus menerus untuk diprediksi oleh model serta algoritme pendeteksi garis dan menggerakkan motor servo pada robot.

4. PENGUJIAN SISTEM

Bab ini berisikan hasil pengujian dan realisasi perancangan robot penjelajah labirin untuk bergerak menelusuri labirin, menganalisa hasil pelatihan, dan menganalisa kestabilan robot. Pengujian yang dilakukan yaitu pengujian kamera, pengukuran kendali robot dan motor, hasil evaluasi akurasi dan performa robot penjelajah labirin, dan pengujian robot secara keseluruhan.

4.1 Pengujian Kamera

Pengujian ini bertujuan untuk memastikan kamera dapat menangkap dan merekam gambar dengan hasil yang baik. Pengujian dilakukan dengan menghubungkan kamera dengan Raspberry Pi, kemudian menjalankan perintah di terminal *Raspberry Pi* (stream_server_test dan stream_client). Setelah perintah dijalankan maka gambar yang ditangkap dari kamera akan ditampilkan pada jendela *image* dan kamera berhasil merekam gambar dengan menghasilkan gambar per detik (*frame per second*) sebesar 30 yang merupakan batas maksimal gambar per detik yang dihasilkan oleh kamera *Raspberry Pi* v1.3. Perhitungan nilai frame per second dihasilkan melalui persamaan (8).

$$fps = \frac{1}{t_1 - t_0} \tag{8}$$

t_1=waktu program berakhir
t_0=waktu program mulai berjalan

Hasil pengujian kamera dengan menjalankan model *Convolutional Neural Network* dan algoritme pendeteksi garis diperlihatkan pada Gambar 12.

Gambar 12. Pengujian frame per second kamera
dengan menjalankan model Convolutional Neural Network

Pengujian kamera menunjukkan bahwa pengiriman gambar setelah model *Convolutional Neural Network* dan algoritme pendeteksi garis diaplikasikan, kamera dapat merekam hingga 10 gambar per detik.

4.2 Pengujian Kendali Motor

Pengujian ini bertujuan untuk mengukur jarak yang ditempuh robot saat diberikan waktu yang ditentukan. Pengujian ini dilakukan untuk mengetahui seberapa cepat robot dapat melintasi satu segmen labirin. Segmen labirin yang diujikan memiliki ukuran 18cm x 18cm untuk satu blok segmen. Pengujian dilakukan dengan menghubungkan setiap motor dengan *Raspberry Pi* sesuai dengan pin GPIO motor yang diinisialisasi pada *Raspberry Pi*. Setelah itu menjalankan perintah di laptop (rc_control_test.py) kemudian tunggu hingga jendela *pygame* terhubung dan tekan tombol panah pada *keyboard*, setelah menjalankan semua perintah maka motor akan bergerak sesuai perintah yang diberikan pengguna melalui

keyboard. Perhitungan jarak yang ditempuh robot dihitung dari lokasi perpindahan roda. Hasil pengujian kendali robot melalui komputer eksternal diperlihatkan pada Tabel 1.

Tabel 1. Tabel pengukuran jarak kendali motor servo setiap detik

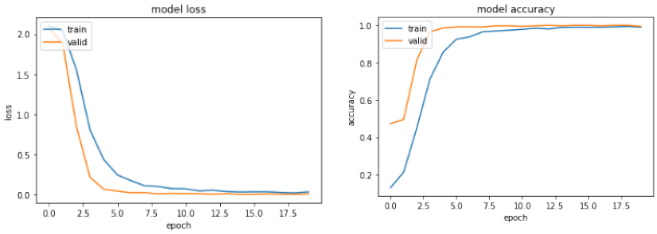
Waktu (detik)	Jarak (cm)
1	10
2	20
3	30
4	40
5	50

4.3 Pengujian Hasil Pelatihan

Pengujian ini bertujuan untuk mengevaluasi model yang akan digunakan pada robot tikus. Pengujian dilakukan dengan menggunakan model hasil pelatihan untuk memprediksi data *test* yang telah dipisahkan dari data *training* dan data *validasi* sebelum melakukan *training*. Parameter yang digunakan pada metode *Convolutional Neural Network* adalah sebagai berikut:

1. Jumlah *convolution layer*: 2 (32 filter dan 16 filter)
2. Fungsi Aktivasi pada *convolution layer*: *relu*
3. Jumlah *hidden layer*: 2 (128 dan 64 node)
4. Optimisasi: *Adaptive Moment (Adam)*
5. Fungsi Aktivasi pada *hidden layer*: *relu*
6. Fungsi Aktivasi pada *output layer*: *softmax*
7. Fungsi Kehilangan: *categorical_crossentropy*

Hasil akhir dari proses pelatihan untuk model *Convolutional Neural Network* dengan nilai akurasi yaitu 96,53% ditunjukkan pada Gambar 14 dalam bentuk grafik:



Gambar 13. Grafik akurasi dan loss data pelatihan terhadap data validasi pada arsitektur Convolutional Neural Network

Dari pengujian yang dilakukan, dihasilkan bahwa robot tikus mampu berjalan sesuai dengan jalur, tetapi karena terkendala oleh tinggi kamera yang rendah, area perekaman yang kurang luas, dan cahaya yang mempengaruhi algoritme pendeteksi garis, robot tikus mendeteksi adanya garis yang merupakan pantulan cahaya dari jalur yang mengharuskan robot untuk berputar balik. Algoritme penjelajahan robot tikus masih belum optimal dalam pengambilan keputusan terhadap jalur simpang tiga (simpang tiga bentuk T, simpang tiga arah kiri dan simpang tiga arah kanan). Penjelajahan yang dilakukan robot tikus tidak keluar dari arena trek.

5. KESIMPULAN

Berdasarkan perancangan dan hasil dari pengujian yang dilakukan, dapat disimpulkan bahwa metode *Convolutional Neural Network* yang dijalankan dapat mendeteksi dengan baik apabila posisi *robot* sejajar dengan jalur labirin, tetapi akibat *overfitting* (tingkat kecocokan berlebihan) membuat *robot* sulit memprediksi bentuk persimpangan saat posisi *robot* dalam keadaan miring. Besar dokumen *numpy* yang menyimpan data jalur yang digunakan di dalam sistem ini adalah 4161 data gambar untuk 8 bentuk simpang (bentuk tidak ada persimpangan, belok kiri, belok kanan, satu arah, simpang tiga T, simpang

tiga kiri, simpang tiga kanan, dan jalur buntu) dan 574 data yang terbagi menjadi 229 data *test* dan 345 data evaluasi. Dengan dokumen ini, didapatkan hasil akurasi pengenalan jalur dengan metode *Convolutional Neural Network* sebesar 96.53%, akurasi terhadap data evaluasi sebesar 97.97% dan akurasi terhadap data *test* sebesar 98.82% yang menunjukkan model terlalu mempelajari dan mencocokkan fitur yang ada dalam data pelatihan.

DAFTAR PUSTAKA

[1] R. Misra and R. Adler, "Micromouse Competition Rules," *IEEE,* 2018.

[2] E. W. Prasetyo, N. Hidetaka, D. A. Prasetya, W. Dirgantara and H. F. Windi, "Spatial Based Deep Learning Autonomous Wheel Robot Using CNN," *Lontar Komputer,* vol. 11, no. 3, 2020.

[3] R. M. Wang, "Application of IEEE Standard Micromouse in the Engineering Practice Teaching," *RESEARCH AND EXPLORATION IN LABORATORY,* vol. 33, no. 9, pp. 117-121, 2014.

[4] M. A. Pangestu and H. Bunyamin, "Analisis Performa dan Pengembangan Sistem Deteksi Ras Anjing pada Gambar dengan Menggunakan Pre-Trained CNN Model," *Jurnal Teknik Informatika dan Sistem Informasi,* vol. 4, no. 2, pp. 337-344, 2018.

[5] J. Feng and S. Lu, "Performance Analysis of Various Activation Functions in Artificial Neural Networks," *Journal of Physics Conference Series,* vol. 1237, no. 2, 2019.

[6] M. G. Abdolrasol, S. M. Hussain, T. S. Ustun, M. R. Sarker, M. A. Hannan, R. Mohamed, J. A. Ali, S. Mekhilef and A. Milad, "Artificial Neural Networks Based Optimization Techniques: A Review," *Electronics,* vol. 10, no. 21, 2021.

[7] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arxiv,* vol. 9, 2014.

[8] H. G. Zhu, "An Efficient Lane Line Detection Method Based on Computer Vision," *Journal of Physics: Conference Series,* vol. 1802, 2021.