Implementasi ANN pada microcontroller pada robot line follower

Laporan Tugas Akhir

Oleh:
Alex Sandro Wijaya (4242211002)
Alex Sonang Nababan (4242211023)

Program Studi Teknik Rekayasa Elektronika
Jurusan Teknik Elektro
Politeknik Negeri Batam
2022

Lembar Pengesahan

Tugas Akhir disusun untuk memenuhi salah satu syarat memperoleh gelar Sarjana Terapan Teknik (S.Tr.T)

Disusun oleh:
Alex Sandro Wijaya (4242211002)
Alex Sonang Nababan (4242211023)

Tanggal Sidang : 14 January 2026

Disetujui oleh :

1. Nanta Fakih Prebianto,S.ST.,M.Sc
NIK: 198903022019031016

1. Sumantri Kurniawan Risandriya,ST, MT
NIK: 197604072012121005

2. Ika Karlina Laila Nur Suciningtyas, S.Si.,M.Si
NIK: 198807062019032012

KATA PENGANTAR

Puji syukur kami panjatkan kepada Tuhan Yang Maha Esa, karena atas rahmat dan karunia-Nya penulis akhirnya dapat menyelesaikan laporan Tugas Akhir dengan judul *"Implementasi Artificial Neural Network (ANN) pada Mikrokontroler Arduino Uno untuk Robot Line Follower"*.

Penyusunan laporan ini merupakan salah satu syarat dalam menyelesaikan pendidikan pada Program Studi Teknik Elektronika, Politeknik Negeri Batam. Selama proses penulisan, penulis mendapatkan banyak dukungan, bimbingan, dan bantuan dari berbagai pihak. Dengan segala kerendahan hati, penulis menyampaikan terima kasih kepada:

1. Tuhan Yang Maha Esa atas segala nikmat dan kemudahan yang diberikan.
2. Kedua orang tua tercinta yang tidak henti-hentinya memberikan doa, dukungan, dan kasih sayang.
3. Dosen pembimbing yang telah sabar memberikan arahan, masukan, serta motivasi kepada penulis dalam menyelesaikan Tugas Akhir ini.
4. Seluruh dosen dan staf Program Studi Teknik Elektronika yang telah membekali penulis dengan ilmu dan pengetahuan selama masa studi.
5. Teman-teman seperjuangan yang selalu memberi semangat dan membantu penulis dalam proses penyusunan laporan ini.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, kritik dan saran yang membangun sangat penulis harapkan demi perbaikan ke depannya. Semoga laporan ini dapat memberikan manfaat serta menambah wawasan bagi pembaca.

Implementasi ANN pada microcontroller pada robot line follower

Abstrak

Robot line follower merupakan salah satu jenis robot mobile yang dirancang untuk mengikuti jalur berupa garis dengan memanfaatkan sensor sebagai masukan utama. Pada umumnya, sistem kendali robot jenis ini masih menggunakan metode konvensional seperti logika if-else atau kontrol PID. Meskipun metode tersebut sederhana dan mudah diimplementasikan, keduanya memiliki keterbatasan dalam hal fleksibilitas, kemampuan adaptasi terhadap perubahan kondisi lintasan, serta memerlukan proses tuning parameter yang cukup kompleks. Untuk mengatasi keterbatasan tersebut, penelitian ini mengembangkan sistem kendali berbasis machine learning dengan menerapkan algoritma Artificial Neural Network (ANN). Model ANN dilatih menggunakan data pembacaan sensor garis dan data keluaran motor yang diperoleh melalui proses simulasi awal. Pelatihan dilakukan dengan pendekatan supervised learning menggunakan arsitektur jaringan feedforward dua lapis tersembunyi. Hasil model terlatih kemudian diimplementasikan pada mikrokontroler Arduino agar mampu memproses data sensor secara real-time dan menghasilkan kendali motor secara otomatis. Pengujian dilakukan pada dua kondisi pencahayaan yang berbeda, yaitu kondisi terang dan gelap total, untuk mengevaluasi kinerja sistem terhadap variasi lingkungan. Hasil pengujian menunjukkan bahwa sistem kendali berbasis ANN mampu mengendalikan robot line follower secara stabil dan responsif pada kedua kondisi tersebut. Sistem mencapai tingkat keberhasilan sebesar 90% dalam menyelesaikan lintasan uji, dengan pergerakan yang lebih halus dan konsisten dibandingkan metode konvensional. Dengan demikian, penerapan algoritma ANN pada sistem kendali robot line follower terbukti dapat meningkatkan akurasi navigasi, konsistensi respons, serta adaptabilitas terhadap perubahan kondisi lingkungan, sekaligus menunjukkan bahwa implementasi machine learning dapat dilakukan secara efektif pada perangkat mikrokontroler dengan sumber daya terbatas.

Kata kunci: *Robot line follower, Artificial Neural Network (ANN), Machine Learning, Arduino, Sistem Kendali*

Implementation of ANN on microcontroller for line follower robot

Abstract

A line follower robot is a type of mobile robot designed to follow a predefined path marked by a line, utilizing sensors as its primary input. In general, conventional control methods such as if-else logic or PID control are still widely used. Although these methods are relatively simple and easy to implement, they have limitations in terms of flexibility, adaptability to changing track conditions, and require a time-consuming parameter tuning process. To overcome these limitations, this study develops a machine learning–based control system by applying an Artificial Neural Network (ANN) algorithm. The ANN model is trained using sensor reading data and motor output data obtained from initial simulations. The training process is conducted using a supervised learning approach with a Multi hidden-layer feedforward architecture. The trained model is then implemented on an Arduino microcontroller to enable real-time sensor data processing and automatic motor control. System testing was carried out under two different lighting conditions bright and completely dark to evaluate the robot's performance under varying environmental factors. The experimental results show that the ANN-based control system can guide the line follower robot stably and responsively in both lighting conditions. The system achieved an overall success rate of 90% in completing the test track, with smoother and more consistent movement compared to conventional control methods. In conclusion, the implementation of the ANN algorithm in the line follower robot control system effectively improves navigation accuracy, response consistency, and adaptability to environmental changes, while also demonstrating that machine learning techniques can be efficiently implemented on microcontroller-based systems with limited computational resources.

***Keywords**: Line follower robot, Artificial Neural Network (ANN), Machine Learning, Arduino, Control System*

Daftar Isi

Daftar Gambar

Daftar Tabel

Bab 1. Pendahuluan

1.1 Latar Belakang

Perkembangan teknologi kecerdasan buatan (*Artificial Intelligence/AI*), khususnya dalam bidang *machine learning*, telah membawa dampak besar pada peningkatan kemampuan sistem otomasi dan robotika. Salah satu bentuk penerapan teknologi ini adalah pada *robot line follower*, yakni robot yang mampu mengikuti jalur tertentu berdasarkan data sensor yang dibaca secara *real-time*. Pada sistem konvensional, pengendalian robot semacam ini masih mengandalkan logika *if-else* atau algoritma PID (*Proportional-Integral-Derivative*). Meskipun metode tersebut sederhana dan mudah diterapkan, keterbatasannya terletak pada rendahnya fleksibilitas, lambatnya adaptasi terhadap perubahan jalur, dan kebutuhan akan penyetelan manual yang memakan waktu. Dalam menghadapi tantangan tersebut, pendekatan *machine learning*, khususnya *Artificial Neural Network (ANN)*, menjadi solusi yang menjanjikan. ANN mampu belajar dari data sensor, mengenali pola-pola kompleks, dan membuat keputusan kendali tanpa perlu aturan eksplisit. Dengan memanfaatkan arsitektur *feedforward*, ANN dapat memproses input dari sensor secara berurutan dan menghasilkan output kendali untuk aktuator (motor) secara cepat. Metode *feedforward* ANN sendiri mengandalkan jaringan saraf tiruan yang tersusun dari lapisan input, hidden layer, dan output yang saling terhubung, tanpa memiliki loop atau umpan balik, sehingga cocok diterapkan pada sistem embedded seperti mikrokontroler. Implementasi *feedforward* ANN pada mikrokontroler seperti Arduino memberikan tantangan tersendiri karena keterbatasan dalam komputasi dan memori. Namun, dengan teknik pelatihan yang dilakukan di luar perangkat (misalnya menggunakan Python), lalu hasilnya disisipkan ke dalam mikrokontroler, sistem dapat berjalan secara efisien dan *real-time*. Pendekatan ini memungkinkan robot *line follower* untuk beroperasi secara mandiri, adaptif terhadap berbagai bentuk jalur, serta lebih akurat dalam pengambilan keputusan dibandingkan pendekatan tradisional. Penelitian oleh[1] membuktikan bahwa penggunaan *Artificial Neural Network (ANN)* dalam sistem kontrol *robot line follower*, khususnya dengan pendekatan *Q-Learning* yang ditingkatkan, mampu meningkatkan akurasi navigasi dan kecepatan respons dibandingkan metode konvensional. Sistem tersebut menunjukkan kemampuan adaptasi yang lebih baik terhadap perubahan bentuk jalur dan kondisi permukaan lintasan karena sifat pembelajaran berbasis pengalaman yang ditanamkan ke dalam pengontrol. Selain itu[2] berhasil mengimplementasikan ANN yang terintegrasi dengan empat sensor inframerah dan mikrokontroler. Jaringan saraf multilayer dengan dua hidden layer tersebut dilatih menggunakan metode *supervised learning* dan menunjukkan hasil yang akurat dalam mengikuti jalur kompleks, bahkan dalam kondisi pencahayaan dan warna permukaan yang

berbeda. Ini menunjukkan potensi besar penerapan ANN dalam sistem embedded yang memiliki keterbatasan daya dan pemrosesan. Model hasil pelatihan kemudian diimplementasikan dalam mikrokontroler menggunakan struktur *feedforward* untuk memungkinkan pengambilan keputusan secara *real-time*. Hal ini didukung oleh studi[3] yang menyatakan bahwa ANN dapat tetap berjalan secara optimal pada mikrokontroler dengan konfigurasi dan pemrograman yang efisien.

Dengan dasar tersebut, penelitian ini berfokus pada perancangan dan implementasi sistem kontrol berbasis *machine learning* menggunakan algoritma *feedforward* ANN pada *robot* line follower. Diharapkan, robot yang dikembangkan mampu menunjukkan performa yang stabil, responsif, dan adaptif dalam mengikuti lintasan yang bervariasi, sekaligus menjadi alternatif solusi pengendalian yang lebih cerdas dan efisien di masa depan.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang sudah diuraikan sebelumnya, rumusan masalah yang akan dibahas yaitu :

1. Merancang dan mengimplementasikan sistem kontrol robot line follower menggunakan algoritma *feedforward Artificial Neural Network* (ANN) pada mikrokontroler Arduino, sehingga robot dapat mengikuti jalur tanpa logika kontrol konvensional.
2. Bagaimana cara melakukan pelatihan model ANN berbasis *machine learning* agar mampu mengenali pola jalur dari data sensor secara akurat dan dapat bekerja secara *real-time* saat diimplementasikan pada perangkat dengan sumber daya terbatas?
3. Bagaimana kinerja sistem *feedforward* ANN dalam mengendalikan arah pergerakan robot dibandingkan dengan metode konvensional seperti logika if-else atau PID, terutama dalam hal akurasi, adaptivitas, dan respons terhadap perubahan kondisi jalur?
4. Melatih model *machine learning* berbasis ANN dengan data sensor yang terbatas, agar mampu mengenali pola jalur secara akurat dan generalisasi keputusan tetap optimal pada kondisi jalur yang bervariasi.
5. Apa saja keunggulan dan keterbatasan penggunaan ANN dalam sistem kontrol robot line follower yang diujikan pada berbagai bentuk dan kondisi jalur?

1.3 Tujuan

Tugas Akhir ini memiliki beberapa tujuan spesifik yang disusun untuk menjawab permasalahan yang telah dirumuskan pada subbab 1.2. Adapun tujuan dari penelitian ini adalah sebagai berikut:

1. Merancang algoritma jaringan syaraf tiruan (ANN) yang mampu mengenali pola jalur pada *robot line follower* dan diimplementasikan langsung ke mikrokontroler agar dapat bekerja secara mandiri.
2. Membangun prototipe *robot line follower* berbasis ANN yang mampu mengikuti jalur dengan variasi bentuk dan kondisi secara optimal.
3. Membangun prototipe robot line follower berbasis *machine learning* (ANN) yang stabil, akurat, dan responsif dalam mengikuti jalur, serta mampu menunjukkan kinerja yang konsisten pada berbagai skenario pengujian jalur.
4. Menghasilkan laporan dan prototipe yang dapat digunakan sebagai media pembelajaran maupun referensi untuk penelitian lanjutan di bidang sistem kendali robotik.

1.4 Manfaat

Pelaksanaan Tugas Akhir ini diharapkan dapat memberikan beberapa manfaat baik dalam jangka pendek maupun jangka panjang. Adapun manfaat dari penelitian ini adalah sebagai berikut :

1. Memberikan referensi nyata bagi mahasiswa dan dosen untuk memahami implementasi jaringan syaraf tiruan pada mikrokontroler dalam sistem robot bergerak.
2. Menyediakan prototipe *robot line follower* yang dapat dijadikan media praktikum di laboratorium robotika atau sistem tertanam.
3. Menjadi dasar pengembangan sistem robot otonom yang lebih adaptif dan responsif terhadap kondisi lingkungan yang kompleks.
4. Memberikan kontribusi teknologi kepada masyarakat dan industri melalui sistem robotik yang lebih efisien untuk berbagai aplikasi seperti logistik dan otomasi rumah tangga.

1.5 Batasan Masalah

Untuk memastikan ruang lingkup penelitian tetap fokus dan sesuai dengan waktu serta sumber daya yang tersedia, berikut adalah batasan yang diterapkan dalam penelitian ini :

1. Robot yang dikembangkan adalah *robot line follower* tiga roda dengan dua motor penggerak dan satu roda pasif.
2. Pelatihan *machine learning* model ANN dilakukan di luar mikrokontroler, menggunakan bahasa pemrograman Python, kemudian bobot hasil

pelatihan dimasukkan ke dalam program Arduino untuk keperluan inferensi *feed* forward .

3. Sistem diimplementasikan menggunakan mikrokontroler Arduino Uno dengan memperhatikan keterbatasan memori dan kecepatan proses.
4. Input berasal dari sensor infrared yang mendeteksi garis jalur hitam dan putih, dan output berupa keputusan arah gerak (kiri, lurus, kanan).
5. Pengujian dilakukan pada lintasan dalam ruangan berupa garis lurus, belokan, dan percabangan sederhana.
6. Evaluasi difokuskan pada akurasi mengikuti jalur dan waktu tempuh, tanpa membahas efisiensi daya atau penghindaran rintangan.
7. Prototipe ditujukan sebagai media pembelajaran, bukan untuk aplikasi industri atau komersial.

1.6 *Work Breakdown Structure*

Tugas Akhir ini dikerjakan secara tim yang terdiri dari dua orang. Setiap anggota tim memiliki tugas dan tanggung jawab masing-masing untuk memastikan proses pengerjaan berjalan secara efektif dan efisien sesuai dengan keahlian yang dimiliki. Pembagian tugas tersebut dapat dilihat pada Tabel 1 berikut:

Tabel 1. *Work Breakdown Structure (WBS)* Tugas Akhir

No	Nama	Tugas dan Tanggung Jawab dalam Tim
1	Alex Sonang Nababan	Pengumpulan data dan implementasi algoritma ANN pada *machine learning*
2	Alex Sandro Wijaya	implementasi metode *Feedforward* pada mikrokontroler *robot line follower*

Pembagian tugas dilakukan untuk mendukung keberhasilan implementasi *Artificial Neural Network (ANN)* pada sistem *robot line follower*, dengan detail sebagai berikut:

1. Alex Sonang Nababan : Bertugas dalam pengumpulan data hasil pengujian robot line follower di lapangan, termasuk dokumentasi performansi sistem di berbagai kondisi jalur. Selain itu, juga membantu dalam proses analisis performa sistem, validasi hasil, serta pembuatan dokumentasi akhir untuk pelaporan dan evaluasi.
2. Alex Sandro Wijaya : Bertanggung jawab dalam melakukan perancangan dan implementasi algoritma *Artificial Neural Network (ANN)* jenis *Feedforward* pada mikrokontroler yang digunakan dalam sistem *robot*

line follower. Proses ini mencakup pemrograman, integrasi sensor, serta pengujian awal algoritma dalam lingkungan simulasi maupun langsung pada robot.

Pembagian ini dilakukan agar setiap anggota dapat fokus pada bagian tertentu, sehingga proses perancangan, implementasi, dan evaluasi dapat berjalan lebih optimal dan terstruktur.

Bab 2. Tinjauan Pustaka

2.1 Penelitian Terdahulu

Untuk mendukung pengembangan robot line follower yang lebih adaptif dan responsif, penulis mencari beberapa referensi dari penelitian terdahulu yang relevan sebagai acuan penting dalam penelitian ini. Hal ini bertujuan untuk memperdalam pemahaman teori dan landasan dalam mengkaji topik penelitian. Studi oleh[4] menunjukkan bagaimana PID dapat dioptimalkan pada *robot line follower* berbasis ARM (STM32F103C8), dengan penyesuaian parameter PID untuk meningkatkan stabilitas dan akurasi pada kecepatan sedang hingga tinggi. *Artificial Neural Network (ANN)* adalah salah satu metode machine learning yang terinspirasi dari cara kerja otak manusia. ANN terdiri dari lapisan input, satu atau lebih hidden layer, dan output, yang semuanya terhubung dengan bobot yang dapat dilatih. ANN dapat memetakan hubungan non-linear antara input dan output, membuatnya cocok untuk aplikasi seperti navigasi robot yang bergantung pada sensor input yang berubah-ubah. Dalam penelitian oleh[5], ANN berbasis LSTM dan CNN digunakan untuk mengontrol *robot line follower* dan menunjukkan kinerja unggul dalam mengikuti lintasan kompleks dan tikungan tajam. Sementara itu, penelitian oleh[6] menunjukkan bahwa ANN multilayer dapat menghasilkan respons sistem yang lebih cepat (~0,1 detik) dibandingkan dengan pendekatan PID, serta lebih adaptif terhadap variasi jalur.

Penelitian oleh[2] menunjukkan implementasi ANN pada *robot line follower* yang dilengkapi dengan empat sensor infrared. Mereka menggunakan arsitektur jaringan dua hidden layer (8 dan 10 neuron), dan pelatihan dilakukan menggunakan metode supervised learning. Hasilnya menunjukkan bahwa robot mampu mengikuti jalur dengan tingkat akurasi yang tinggi bahkan pada variasi warna lintasan dan kondisi pencahayaan yang berbeda.

Berikutnya, penelitian yang dilakukan oleh[3], yang merancang *robot line follower* berbasis mikrokontroler ATmega32A dengan sistem kendali konvensional. Mereka menemukan bahwa sistem hanya bekerja optimal pada kecepatan rendah (90–150 RPM), dan mulai kehilangan akurasi saat kecepatan meningkat karena keterbatasan sensitivitas sensor. Selanjutnya penelitian oleh[7] mengimplementasikan *robot line follower* berbasis Arduino Uno dan menekankan kelemahan pendekatan berbasis logika if-else yang masih memerlukan kalibrasi manual dan kurang adaptif terhadap kondisi jalur yang berubah-ubah.

Dari beberapa penelitian tersebut dapat disimpulkan bahwa pendekatan ANN memiliki potensi besar untuk mengatasi keterbatasan sistem konvensional, khususnya dalam hal fleksibilitas, akurasi, dan adaptasi lintasan. Pendekatan ini sangat mendukung implementasi sistem navigasi berbasis mikrokontroler dengan keterbatasan sumber daya.

2.2 *Machine learning*

Machine Learning merupakan cabang dari kecerdasan buatan (Artificial Intelligence) yang berfokus pada metode agar sistem dapat belajar dari data. Dalam machine learning, sebuah model dilatih menggunakan kumpulan data sehingga mampu mengenali pola dan melakukan prediksi tanpa perlu diprogram secara eksplisit. Machine learning terbagi menjadi tiga jenis utama, yaitu:

1. *Supervised Learning*, yaitu pelatihan dengan data berlabel (memiliki input dan target).
2. *Unsupervised Learning*, yaitu pelatihan tanpa target, misalnya pengelompokan (clustering).
3. *Reinforcement Learning*, yaitu pembelajaran berdasarkan reward dan punishment.

Penelitian ini menggunakan *Supervised Learning*, karena model ANN dilatih menggunakan data input sensor dan target output (aksi robot) yang sudah ditentukan.

Menurut[9], *machine learning* didefinisikan sebagai "*A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance at task T, as measured by P, improves with experience E.* ". Fungsi prediksi dalam *supervised learning* umumnya dapat dituliskan: $\hat{y}=f(x;\theta)$ Sistem dilatih menggunakan data berlabel (input-output). Contoh: klasifikasi arah belokan berdasarkan data sensor. Tujuan model adalah meminimalkan kesalahan prediksi terhadap data nyata, yang dihitung menggunakan fungsi kerugian (*loss function*):

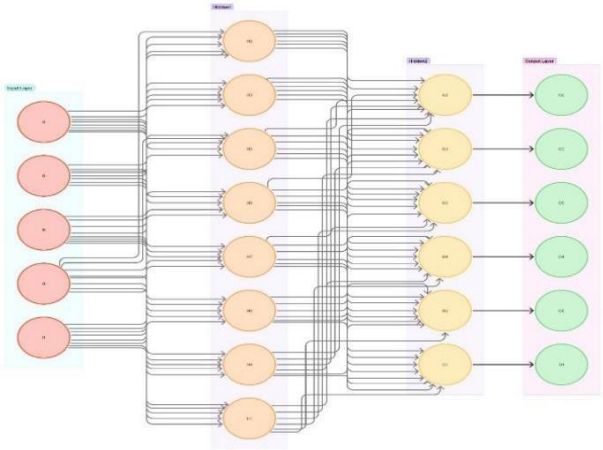
$$\text{Loss}=\frac{1}{n}\left(\sum_{i=1}^{n}(y1-\hat{y}1\right)2$$

Semakin kecil nilai Loss, semakin akurat model tersebut dalam memprediksi data. Pada sistem *Machine Learning*, keputusan diambil berdasarkan bobot yang diperoleh dari proses pelatihan, bukan dari logika pemrograman eksplisit.Dengan pendekatan ini, robot dapat melakukan navigasi jalur secara otomatis dan responsif bahkan pada kondisi jalur yang tidak ideal.

2.3 *Artificial Neural Network (ANN)*

Artificial Neural Network (ANN) atau jaringan saraf tiruan adalah model komputasi yang terinspirasi dari cara kerja otak manusia. ANN terdiri atas kumpulan neuron buatan yang terhubung satu sama lain melalui bobot *(weights)*. ANN banyak digunakan untuk klasifikasi, pengenalan pola, dan sistem pengambilan keputusan. Secara umum ANN memiliki tiga jenis lapisan:

1. *Input Layer*, Berfungsi menerima sinyal atau data dari lingkungan luar. Pada penelitian ini input berasal dari pembacaan sensor IR pada jalur robot *line follower*.
2. *Hidden Layer*, Lapisan ini memproses sinyal menggunakan bobot dan bias. *Hidden layer* berfungsi mengekstraksi pola dari data input melalui fungsi aktivasi seperti ReLU, sigmoid, atau tanh.
3. *Output Layer*, Lapisan keluaran menghasilkan keputusan akhir berdasarkan hasil perhitungan dari *hidden layer*. Pada penelitian ini, output berupa kelas aksi robot seperti belok kiri, lurus, atau belok kanan.



Gambar 1 Arsitektur Perceptron

2.4 *Multilayer Perceptron (MLP)*

Multilayer Perceptron (MLP) adalah jenis ANN yang tersusun dari satu input layer, satu atau lebih hidden layer, dan satu output layer. MLP termasuk jaringan *feedforward*, yaitu sinyal mengalir dari input menuju output tanpa umpan balik.

Setiap neuron dalam MLP menghitung : $Z_j = \int_i^j w_{ij}x_i + b_j$

Hasilnya kemudian dilewatkan ke fungsi aktivasi: $a_j = f(z_j)$

MLP dapat menangani pola yang kompleks, sehingga banyak digunakan untuk klasifikasi seperti pada robot line follower.

2.5 Fungsi Aktivasi

Fungsi aktivasi berperan untuk mengenalkan sifat non-linear pada jaringan. Tanpa fungsi aktivasi, ANN hanya menjadi model linear sederhana yang tidak mampu mempelajari pola kompleks. Beberapa fungsi aktivasi yang kami gunakan:

1. ReLU *(Rectified Linear Unit),* Umumnya digunakan pada hidden layer untuk mempercepat konvergensi.

$$f(x) = \max(0, x)$$

2. SoftMax, Digunakan untuk klasifikasi multi-kelas.

$$softmax(z_k) = \frac{e^{zk}}{\int e^{zi}}$$

Pada penelitian ini, hidden layer menggunakan ReLU dan output menggunakan softmax.

2.6 Algoritma Backpropagation

Backpropagation adalah algoritma inti yang digunakan untuk melatih MLP. Algoritma ini bekerja dalam dua fase:

1. *Forward Pass*, Menghitung output berdasarkan bobot saat ini.
2. *Backward Pass*, Menghitung error dan menyesuaikan bobot agar kesalahan menurun.

Backpropagation meminimalkan fungsi error, misalnya: $E = (\frac{1}{2} y_{target} - y)^2$.
Berikut tahapan dari Backpropagation :

1. Inisialisasi Bobot dan Parameter
   Bobot awal diinisialisasi secara acak dalam rentang kecil (misalnya −0.5 sampai 0.5). Parameter lain seperti learning rate α ditentukan.
2. Forward Pass
   - Hitung Output Hidden layer $z_j = \sum_i w_{ij} x_i + b_j$

$$y_j = f(z_j)$$

   - Hitung Output Layer $z_j = \sum_j w_{kj} y_j + b_k$

$$y_j = f(z_k)$$

3. Hitung Eror, eror ini menunjukkan seberapa jauh prediksi dari target.
   $e_k = y_{dk} - y_k$
4. Hitung Gradien Error
   - Delta pada output layer $\delta_k = e_k \, . y_k \, (1 - y_k)$
   - Delta pada hidden layer $\delta_j = y_j(1 - y_j) \sum_k w_{jk}\delta_k$
5. Update Bobot, bobot diperbaharui sesuai gradien error
   $\Delta w_{ij} = \propto . x_i \, . \delta_j$ dan $w_{ij}\,(baru) = w_{ij}(lama) + \Delta w_{ij}$
6. Hitung error total menggunakan fungsi loss seperti MSE atau entropy

2.7 Bias Pada Artificial Neural Network

Bias merupakan parameter tambahan pada setiap neuron yang berfungsi untuk menggeser nilai aktivasi neuron. Bias memungkinkan model untuk tetap menghasilkan output meskipun seluruh input bernilai nol. Dengan adanya bias, jaringan saraf tidak harus selalu melewati titik nol sehingga fleksibilitas model dalam mempelajari pola data menjadi lebih baik.

Bias ditambahkan pada hasil perkalian antara input dan bobot sebelum fungsi aktivasi diterapkan. Tanpa bias, kemampuan jaringan dalam menyesuaikan diri terhadap variasi data akan terbatas, khususnya pada kasus klasifikasi dengan pola yang tidak simetris.

2.8 Parameter pada proses Training ANN

Pada proses pelatihan Artificial Neural Network menggunakan Python, terdapat beberapa parameter utama yang sangat berpengaruh terhadap performa model, antara lain:

1. Bobot (Weight), Bobot merupakan nilai pengali antara input dan neuron pada layer berikutnya. Bobot inilah yang dipelajari selama proses training agar output jaringan mendekati target yang diinginkan.
2. Bias, berfungsi sebagai konstanta penyesuaian yang ditambahkan pada setiap neuron untuk meningkatkan kemampuan jaringan dalam merepresentasikan data.
3. Learning Rate, menentukan besar langkah perubahan bobot pada setiap iterasi training. Nilai learning rate yang terlalu besar dapat menyebabkan proses training tidak stabil, sedangkan nilai yang terlalu kecil dapat memperlambat konvergensi model.
4. Epoch, merupakan jumlah pengulangan proses training terhadap seluruh data latih. Semakin besar epoch, semakin banyak kesempatan jaringan untuk memperbaiki bobotnya hingga mencapai kondisi konvergen.

5. Loss Function, digunakan untuk mengukur selisih antara output jaringan dengan target yang diharapkan. Nilai loss yang semakin kecil menunjukkan bahwa performa model semakin baik.

2.9 Proses Training

ANN dapat melakukan pembelajaran dengan cara menyesuaikan bobot agar prediksi semakin mendekati target. Pembelajaran dilakukan dengan memberikan contoh input dan target yang sesuai (supervised learning). Proses pembelajaran melibatkan:

1. Persiapan data, data sensor garis dikumpulkan dan disusun dalam bentuk pasangan input dan target, input mempresentasikan kondisi sensor, sedangkan target mempresentasikan aksi gerak robot.
2. Inisialisasi bobot dan bias dengan nilai awal tertentu, biasanya secara acak, sebelum proses training dimulai.
3. Data input diproses melalui setiap layer jaringan menggunakan bobot dan bias untuk menghasilkan output sementara.
4. Output jaringan dibandingkan dengan target untuk menghitung nilai error menggunakan fungsi loss yang telah ditentukan.
5. Error yang diperoleh digunakan untuk memperbarui bobot dan bias melalui algoritma backpropagation agar kesalahan pada iterasi berikutnya dapat dikurangi.
6. Proses training diulang hingga mencapai jumlah epoch yang ditentukan atau hingga nilai loss mengecil.

Setelah proses training selesai, bobot dan bias hasil pelatihan disimpan dan kemudian ditransfer ke mikrokontroler untuk digunakan pada proses feedforward secara real-time.

2.10 Proses Pengumpulan dan Pelatihan Data ANN

Data yang digunakan pada proses pelatihan Artificial Neural Network (ANN) diperoleh dari hasil pembacaan sensor inframerah pada robot line follower. Robot dilengkapi dengan lima buah sensor infrared (S1–S5) yang dipasang sejajar di bagian bawah robot untuk mendeteksi keberadaan garis lintasan. Setiap sensor menghasilkan nilai biner, yaitu 1 jika sensor mendeteksi garis hitam dan 0 jika sensor berada di atas permukaan putih.

Nilai 0 dan 1 pada data seperti 0,1,0,0,0 merupakan hasil pembacaan sensor inframerah pada robot line follower. Nilai 1 menunjukkan sensor mendeteksi garis, sedangkan 0 menunjukkan sensor tidak mendeteksi garis. Nilai ini digunakan karena sensor bekerja secara digital dan lebih mudah diproses oleh mikrokontroler dan Artificial Neural Network (ANN). Kombinasi nilai tersebut

merupakan data input yang diperoleh dari sensor, bukan hasil keluaran ANN. Data input ini disimpan dalam file CSV dan digunakan sebagai data latih pada proses training ANN di Python. Setelah proses pelatihan selesai, ANN menghasilkan bobot (weight) yang kemudian dikirim ke mikrokontroler Arduino untuk digunakan pada proses feedforward saat robot berjalan secara real-time.

2.11 Pembentukan Bobot dan Bias pada ANN

Nilai weight dan bias merupakan parameter hasil pelatihan Artificial Neural Network (ANN) yang dilakukan menggunakan Keras di lingkungan Python. Angka desimal seperti 0.04533237, -1.80219328, atau 2.00442696 bukan ditentukan secara manual, melainkan dihasilkan melalui proses training menggunakan algoritma backpropagation. Pada tahap awal pelatihan, nilai weight dan bias diinisialisasi secara acak dengan nilai kecil, kemudian diperbarui secara bertahap berdasarkan error antara keluaran model dan target yang diharapkan pada data latih.

Selama proses training, setiap data input sensor diproses melalui jaringan saraf menggunakan operasi perkalian dan penjumlahan antara nilai input, weight, dan bias pada setiap neuron. Hasil perhitungan tersebut kemudian dilewatkan ke fungsi aktivasi untuk menghasilkan output sementara. Selisih antara output model dan target digunakan untuk menghitung error, yang selanjutnya dipropagasikan kembali ke setiap layer untuk menyesuaikan nilai weight dan bias. Proses ini diulang selama beberapa epoch hingga error minimum tercapai dan model memperoleh performa yang optimal.

Setelah proses training selesai, nilai weight dan bias terbaik disimpan sebagai parameter tetap dan diekstraksi dari model Keras. Parameter tersebut kemudian dikonversi ke dalam bentuk array bahasa C/C++ agar dapat digunakan pada mikrokontroler Arduino. Pada tahap implementasi, Arduino tidak lagi melakukan proses pelatihan, melainkan hanya menjalankan proses feedforward menggunakan nilai weight dan bias tersebut. Setiap pembacaan sensor real-time diproses melalui jaringan saraf untuk menghasilkan keluaran berupa keputusan arah gerak robot. Dengan demikian, weight dan bias berfungsi sebagai pengetahuan hasil pembelajaran ANN yang memungkinkan robot mengambil keputusan secara cerdas tanpa menggunakan logika konvensional berbasis aturan.

2.12 Implementasi FeedForward di Microcontroller

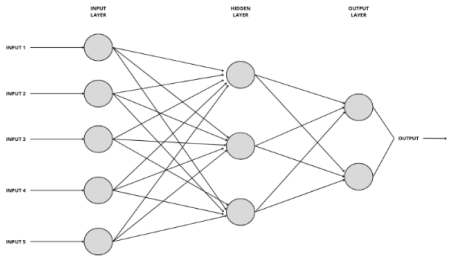
Pelatihan ANN dilakukan sepenuhnya di komputer menggunakan TensorFlow/Keras dengan algoritma backpropagation. Proses tersebut menghasilkan bobot dan bias optimal. Kemudian bobot tersebut diekspor dalam format yang sudah ditentukan dipytohn. Di microcontroller tidak ada lagi

backpropagation dan update bobot, yang hanya dilakukan adalah membaca sensor, menjalankan feedforward, dan menghasilkan keputusan gerak.

Feedforward adalah proses menghitung output jaringan berdasarkan input dan bobot yang sudah ada. Proses ini digunakan pada mikrokontroler karena ringan dan tidak memerlukan proses pembelajaran (training). *FeedForward* tidak memiliki koneksi kembali dari output ke input neuron dan oleh karena itu tidak menyimpan catatan nilai output sebelumnya. Tahapan feedforward:

1. Membaca input (sensor IR).
2. Mengalikan input dengan bobot pada hidden layer.
3. Menambahkan bias.
4. Melewatkan nilai ke fungsi aktivasi.
5. Mengalikan hasil hidden layer dengan bobot output layer.
6. Menghasilkan output berupa kelas dengan probabilitas tertinggi (softmax + argmax).

Proses *feedforward* sangat cepat sehingga cocok dijalankan di Arduino untuk mengatur pergerakan robot secara real-time.



Gambar 2 FeedForward Network[8]

FNN (Feedforward Neural Network) digunakan pada penelitian ini untuk memproses data sensor garis pada robot line follower sehingga menghasilkan keputusan gerakan secara real time di lingkungan microcontroller. Pada bagian ini dijelaskan teori FNN yang diimplementasikan di microcontroller dalam tiga bagian utama, yaitu:

2.12.1 Representasi Input dan Parameter Jaringan

Pada microcontroller, input jaringan berupa lima sinyal sensor infrared yang merepresentasikan kondisi garis:

$$X = [x_1, x_2, x_3, x_4, x_5]$$

Nilai tiap sensor berupa bilangan biner (0 = tidak mengenai garis, 1 = mengenai garis). Jaringan dilengkapi dengan parameter:

- Bobot (weight) pada tiap koneksi antar neuron

$W^{(l)}$

- Bias pada setiap neuron

$b^{(l)}$

Pada penelitian ini digunakan arsitektur: 5 input – 8 neuron – 6 neuron – 6 output. Semua bobot dan bias disimpan dalam file header C/C++ dan dimuat pada saat program berjalan di Arduino.

2.12.2 Perhitungan Hidden Layer (Weighted Sum + ReLU)

Setiap neuron pada hidden layer menghitung operasi linear :

$$h1_j = \int_{i=1}^{5}(x_i \,.W1_{ij}) + b1_j$$

Kemudian diaktifkan menggunakan fungsi ReLU *(Rectified Linear Unit)* :

$$h_j^{(1)} = \max\,(0, z_j^{(1)}$$

ReLU dipilih karena komputasinya ringan untuk microcontroller (hanya operasi perbandingan), mengurangi risiko vanishing gradient, dan cepat diproses pada integer/float 8-bit atau 32-bit.

Begitu juga dengan hidden layer 2 untuk operasi linear nya sama seperti dihidden layer 1. Hidden layer 2 memperkuat pola yang dipelajari saat pelatihan dipython, sehingga jaringan mampu mengenali pola sensor yang lebih kompleks seperti belokan tajam dan persimpangan.

2.12.3 Perhitungan Output Layer dan Softmax

Output layer tidak menggunakan fungsi ReLU, melainkan menghasilkan logit, yaitu nilai linear sebelum dikonversi ke probabilitas:

$$o_m = \int_{k=1}^{6}(h2_k \,.W3_{km}) + \, b3_m$$

Karena output ingin merepresentasikan pilihan aksi robot (klasifikasi), digunakan fungsi **Softmax**:

$$y_m = \frac{e^{om}}{\sum_{i=1}^{6} e^{oi}}$$

Softmax menghasilkan probabilitas 0-1 untuk setiap kelas aksi. Total probabilitas adalah 1 dan output terbesar adalah kelas terpilih. Dalam penelitian ini ada 6 kelas yaitu (Forward, Left-low, Left-max, Right-low, Right-max, Turn-around). Microcontroller kemudian menjalankan aksi motor berdasarkan kelas

prediksi terbesar. Keenam probabilitas tersebut mewakili enam kelas gerakan robot, yaitu : (0 = Forward, 1 = Left_Low, 2 = Left_max, 3 = Right_Low, 4 = Right_Max, 5 = Turn Around). Contoh probabilitas untuk menentukan aksi robot, ada 6 probabilitas (0.102, 0.154, 0.087, 0.215, **0.391**, 0.051) jadi nilai terbesar berada di kelas 4 yang berarti prediksi nya *RightMax* (robot akan melakukan gerakan belok kekanan max).

2.13 Hubungan Training diPyhton & FeedForward diMicrocontroller

Pada sistem ini microcontroller tidak melakukan proses training ulang. Microcontroller hanya menjalankan proses FeedForward menggunakan bobot dan bias hasil dari training. Dengan pengujian ini, sistem mampu memanfaatkan keunggulan machine learning tanpa membebani microcontroller dengan komputasi yang berat.

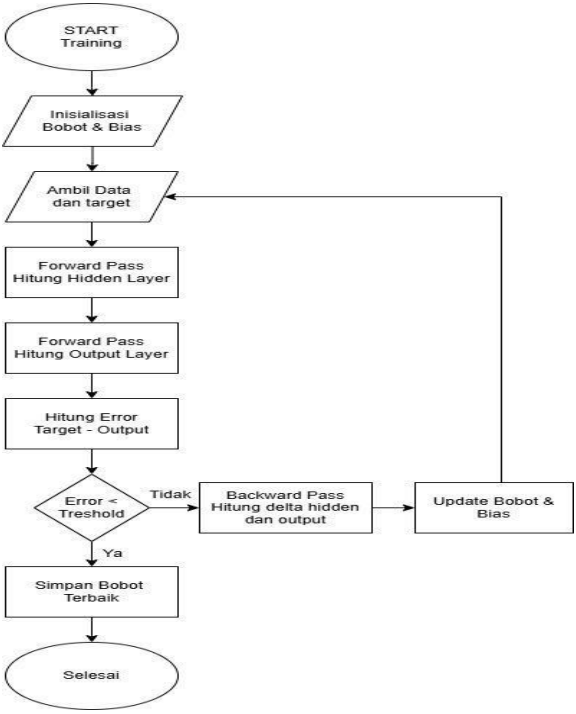
Pendekatan ini memastikan bahwa keputusan arah gerak robot merupakan hasil inferensi dari model yang telah terlatih dengan baik, sehingga respons robot terhadap kondisi sensor dapat berjalan secara cepat dan stabil.

Bab 3. Metodologi Penelitian

3.1 Perancangan

Pada Gambar 3 dijelaskan alur proses *machine learning* yang digunakan untuk membangun model *Artificial Neural Network (ANN)* sebagai sistem kendali pada *robot line follower*. Tahapan ini merupakan bagian penting sebelum implementasi ke mikrokontroler dilakukan, karena di sinilah model ANN dilatih menggunakan data sensor. Adapun tahapan proses *machine learning* tersebut adalah sebagai berikut:

1. Mulai proses training
2. Inisialisasi bobot & bias
3. Ambil data input & target
4. Forward pass – hitung hidden layer
5. Forward pass – hitung Output layer
6. Hitung error & hitung nilai loss
7. Error < Threshold?
   - Ya → Simpan Model → Selesai
   - Tidak → Backpropagation
8. Backward pass – hitung delta output
9. Backward pass – hitung delta hidden layer
10. Update bobot & bias
11. Ulangi ke loop berikutnya
12. Simpan model terbaik (model dengan error paling kecil atau akurasi terbaik)
13. Training selesai



Gambar 3 Flowchart Training

Pada gambar 4 dijelaskan terkait dengan flowchart feedforward yang dilakukan dimicrocontroller dengan beberapa tahapan. Rancangan tersebut terbagi menjadi beberapa tahapan yaitu :

1. Mulai
2. Baca sensor IR
3. Hitung Input ke Hidden Layer (Weighted Sum)
4. Hitung Aktivasi Hidden Layer
5. Hitung Output Layer
6. Hitung Softmax → Probabilitas Kelas
7. Ambil Keputusan (Argmax) (Kelas dengan probabilitas terbesar dipilih)
8. Kirim Perintah ke Motor
9. Ulangi Kembali ke Pembacaan Sensor (Loop feedforward berjalan secara terus-menerus selama robot aktif)

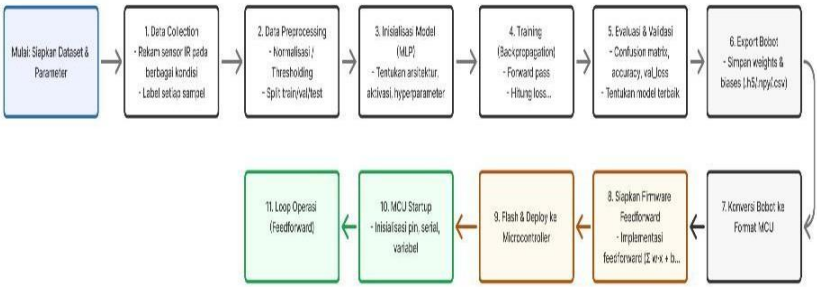
Gambar 4 Flowchart FeedForward diMicrocontroller

3.1.1 Perancangan Sistem Kerja

Sistem kerja robot line follower ini dirancang untuk dapat mengenali jalur dan melakukan koreksi arah secara otomatis melalui dua tahap yaitu, proses pembelajaran pola jalur melalui Machine Learning, dan penerapan model hasil pelatihan ke dalam microcontroller melalui metode FeedForward. Proses diawali dengan pengumpulan data sensor dan label arah gerak robot pada berbagai bentuk jalur. Data ini digunakan untuk melatih model Artificial Neural Network (ANN) menggunakan metode supervised learning. Model ANN ini dirancang dengan struktur feedforward, yang meliputi lapisan input (jumlah sensor), hidden layer, dan output layer (keputusan arah). Setelah dilakukan proses pelatihan dan evaluasi di komputer, model terbaik disimpan dalam bentuk bobot dan bias.

Bobot hasil pelatihan kemudian diterapkan ke dalam mikrokontroler Arduino, di mana sistem bekerja secara *real-time* menggunakan proses *feedforward*. Saat

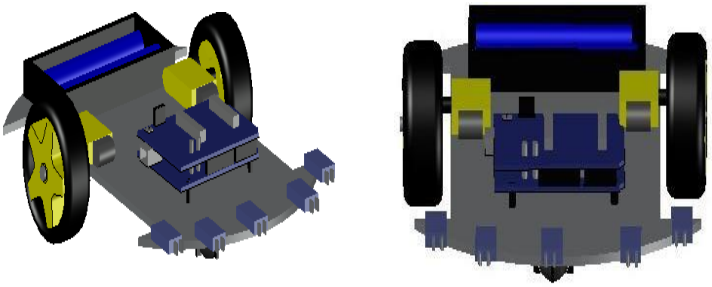
robot berjalan, sensor inframerah yang dipasang di bagian bawah robot mendeteksi warna permukaan jalur. Sensor ini membaca intensitas pantulan cahaya, warna hitam menyerap lebih banyak cahaya dibandingkan putih dan menghasilkan data digital (1/0) sebagai representasi kondisi jalur. Data dari sensor dikirim ke mikrokontroler sebagai input utama, lalu diproses oleh fungsi *feedforward ANN*. Output dari jaringan saraf digunakan untuk menentukan arah dan kecepatan motor secara langsung. Dengan demikian, robot dapat mengikuti jalur tanpa menggunakan logika *if-else* atau PID, melainkan berdasarkan pola pembelajaran yang telah dilakukan sebelumnya. Seluruh proses dari pembacaan sensor, pengolahan data melalui ANN, hingga pergerakan motor berlangsung secara terintegrasi dan *real-time*, memungkinkan sistem bekerja secara otonom dan adaptif terhadap variasi jalur.



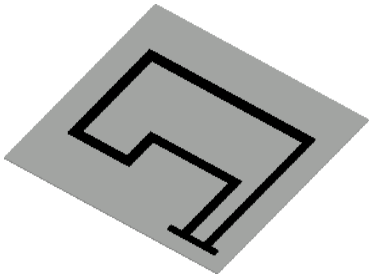
Gambar 5 Blok Diagram Sistem Kerja

3.1.2 Perancangan Mechanical

Struktur mekanik robot dibuat menggunakan rangka akrilik yang ringan dan kuat. Desain sasis menempatkan 3 roda, dua roda penggerak utama di bagian belakang dan 1 *free wheel* di depan untuk menjaga keseimbangan. Komponen seperti sensor, mikrokontroler, dan baterai dipasang secara simetris agar distribusi berat seimbang.



Gambar 6 Design Mechanical

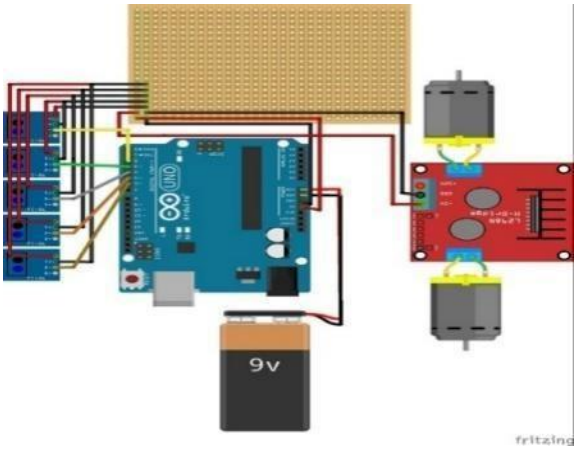


Gambar 7 Design Lintasan Uji

Sensor inframerah dipasang di bagian depan bawah sasis, berjajar sejajar jalur lintasan, untuk memastikan pembacaan jalur presisi. Desain ini memungkinkan robot berbelok secara stabil dan responsif saat mendeteksi perubahan arah jalur.

3.1.3 Perancangan Electrical

Sistem electrical dikendalikan oleh Arduino Uno sebagai pusat pemrosesan. Sensor TCRT5000 terhubung ke pin input Arduino dan memberikan sinyal.



Gambar 8 Design Electrical

berdasarkan kondisi jalur (hitam/putih). Data ini diproses oleh algoritma *Artificial Neural Network (ANN)* yang tertanam dalam program Arduino. Output dari ANN dikonversi menjadi sinyal PWM dan dikirim ke driver motor L298N, yang kemudian mengatur arah dan kecepatan dua motor DC. Sumber daya utama berasal dari baterai Li-ion 9V, yang disambungkan ke Arduino dan L298N. Semua koneksi antar komponen menggunakan jumper atau solder untuk memastikan kestabilan saat robot bergerak.

3.2 Alat dan Bahan

Sub bab ini menjelaskan mengenai alat dan bahan. Selain itu, pada bagian ini juga disertakan estimasi biaya dari setiap alat dan bahan yang digunakan dalam bentuk tabel untuk memberikan gambaran total kebutuhan biaya penelitian.

Tabel 2. Rekapitulasi Rencana Anggaran

NO	Jenis Pengeluaran	Sumber Dana	Besaran Dana (Rp)
1	Komponen Elektronik		
	-Arduino Uno	Pribadi	Rp 125,000
	- Sensor IR TCRT5000 (7 Pcs)	Pribadi	Rp 350,000
	-Driver Motor L298N	Pribadi	Rp 25,000
	- Motor DC (4 Pcs)	Pribadi	Rp 68,000
	- Baterai Li-ion (4 Pcs)	Pribadi	Rp 22,000
2	Material Rangka		
	- Akrilik	Pribadi	Rp 50,000
	- Baut, mur, kabel jumper, roda, dll	Pribadi	Rp 200,000
3	Biaya Lain-Lain	Pribadi	Rp 300,000
Jumlah			Rp 1,140,000

3.3 Pengujian

3.3.1 Pengujian 1

Pada tahap ini, dilakukan pengujian terhadap masing-masing sensor inframerah (TCRT5000) untuk memastikan bahwa setiap sensor dapat mengenali warna jalur dengan benar. Setiap sensor diuji dengan meletakkannya di atas warna hitam dan putih secara bergantian, dan dilihat output logikanya (HIGH/LOW) pada serial monitor Arduino. Tujuannya adalah memastikan bahwa pembacaan sensor akurat sebelum digunakan sebagai input pada jaringan ANN.

Tabel 3 Pengujian Sensor

Sensor	Kondisi Sensor	Output (Logika)	Keterangan
S1 (Kanan)	Sensor diarahkan ke garis	1	Sensor mendeteksi garis di sisi kanan
S1 (Kanan)	Sensor diarahkan ke latar	0	Sensor tidak mendeteksi garis
S2	Sensor diarahkan ke garis	1	Sensor mendeteksi garis
S2	Sensor diarahkan ke latar	0	Sensor tidak mendeteksi garis
S3(Tengah)	Sensor diarahkan ke garis	1	Sensor mendeteksi garis di posisi tengah
S3(Tengah)	Sensor diarahkan ke latar	0	Sensor tidak mendeteksi garis
S4	Sensor diarahkan ke garis	1	Sensor mendeteksi garis
S4	Sensor diarahkan ke latar	0	Sensor tidak mendeteksi garis
S5 (Kiri)	Sensor diarahkan ke garis	1	Sensor mendeteksi garis di sisi kiri
S5 (Kiri)	Sensor diarahkan ke latar	0	Sensor tidak mendeteksi garis

3.3.2 Pengujian 2

Sebelum model ANN diimplementasikan ke microcontroller, dilakukan tahap pengujian di python untuk memastikan bahwa model yang telah dilatih memiliki performa yang layak dan bekerja sesuai tujuan. Hasil uji ini memberikan gambaran kemampuan ANN dalam mengenali pola posisi garis dan menerjemahkannya menjadi kelas tindakan seperti *forward, left_Low, left_max, right_low, right_Max atau turn-around.*

Tabel 4 Pengujian diPyhton sebelum diimplementasikan pada robot

NO	ActionRobot	Input Sensor	Catatan
1	FORWARD	0, 0, 1, 0, 0	Robot bergerak maju
		0, 1, 1, 1, 0	
		0, 1, 1, 1, 0	
		0, 0, 1, 0, 0	
2	LEFT MAX	1, 0, 0, 0, 0	Robot bergerak kiri max
		1, 1, 0, 0, 0	
		1, 1, 0, 1, 0	
3	LEFT LOW	0, 1, 0, 0, 0	Robot bergerak kiri low
		0, 1, 1, 0, 0	
		1, 1, 1, 0, 0	
4	RIGHT MAX	0, 0, 0, 0, 1	Robot bergerak kanan max
		0, 0, 0, 1, 1	
		0, 1, 0, 1, 1	
5	RIGHT LOW	0, 0, 0, 1, 0	Robot bergerak kanan low
		0, 0, 1, 1, 0	
		0, 0, 1, 1, 1	
		1, 0, 1, 1, 1	
6	TURN AROUND	1, 1, 1, 1, 1	Robot berputar
		1, 0, 0, 0, 1	
		1, 0, 0, 1, 1	
		1, 0, 1, 0, 1	

3.3.3 Pengujian 3

Pengujian 3 dilakukan untuk mengevaluasi kinerja model Artificial Neural Network (ANN) setelah hasil proses pembelajaran (learning) dari lingkungan Python diimplementasikan ke dalam microcontroller Arduino. Pada pengujian ini, parameter hasil pembelajaran ANN berupa bobot (weight) dan bias yang diperoleh dari proses training di Python diintegrasikan ke dalam program microcontroller. Selanjutnya, microcontroller melakukan proses feedforward menggunakan nilai input sensor yang diberikan, baik secara manual (hardcoded) maupun melalui pembacaan sensor fisik. Hasil keluaran ANN berupa nilai probabilitas (softmax) dan kelas prediksi diamati melalui Serial Monitor.

Tabel 5 Pengujian Implementasi Model ANN pada Microcontroller Arduino

NO	ActionRobot	Input Sensor	Catatan
1	FORWARD	0, 0, 1, 0, 0	Robot bergerak maju
		0, 1, 1, 1, 0	
		0, 1, 1, 1, 0	
		0, 0, 1, 0, 0	
2	LEFT MAX	1, 0, 0, 0, 0	Robot bergerak kiri max
		1, 1, 0, 0, 0	
		1, 1, 0, 1, 0	
3	LEFT LOW	0, 1, 0, 0, 0	Robot bergerak kiri low
		0, 1, 1, 0, 0	
		1, 1, 1, 0, 0	
4	RIGHT MAX	0, 0, 0, 0, 1	Robot bergerak kanan max
		0, 0, 0, 1, 1	
		0, 1, 0, 1, 1	
5	RIGHT LOW	0, 0, 0, 1, 0	Robot bergerak kanan low
		0, 0, 1, 1, 0	
		0, 0, 1, 1, 1	
		1, 0, 1, 1, 1	
6	TURN AROUND	1, 1, 1, 1, 1	Robot berputar
		1, 0, 0, 0, 1	
		1, 0, 0, 1, 1	
		1, 0, 1, 0, 1	

Berdasarkan hasil pengujian yang dilakukan, dapat disimpulkan bahwa model ANN hasil pembelajaran dapat diimplementasikan dengan baik pada microcontroller Arduino dan mampu menghasilkan keluaran yang konsisten dengan hasil pengujian pada lingkungan Python.

3.3.4 Pengujian 4

Pengujian 4 dilakukan untuk menguji konsistensi hasil *feedforward ANN* pada *microcontroller* terhadap perubahan kondisi pencahayaan. Hasil pengujian menunjukkan bahwa untuk input sensor yang sama, ANN menghasilkan kelas prediksi yang sama pada kondisi terang, redup/padam, sehingga sistem bisa stabil terhadap variasi intensitas cahaya.

Tabel 6 Feedforward ANN terhadap Variasi Kondisi Pencahayaan

Kondisi Cahaya	Input Sensor (S1–S5)	Hasil Feedforward (Prediksi)
Terang	0, 1, 1, 1, 0	FORWARD
	1, 1, 0, 0, 0	LEFT_MAX
	1, 1, 1, 0, 0	LEFT_LOW
	0, 0, 0, 1, 1	RIGHT_MAX
	1, 0, 1, 1, 1	RIGHT_LOW
	1, 0, 0, 1, 1	TURN AROUND
Gelap/Redup	0, 0, 1, 0, 0	FORWARD
	1, 0, 0, 0, 0	LEFT_MAX
	0, 1, 1, 0, 0	LEFT_LOW
	0, 0, 0, 0, 1	RIGHT_MAX
	1, 0, 1, 1, 1	RIGHT_LOW
	1, 1, 1, 1, 1	TURN AROUND

3.3.5 Pengujian 5

Pengujian ini dilakukan dengan menggabungkan hasil training ANN di Python dan hasil implementasi bobot ke dalam microcontroller. Variasi epoch digunakan untuk mengamati pengaruh proses learning terhadap kestabilan hasil feedforward pada sistem tertanam. Hasil pengujian menunjukkan bahwa meskipun akurasi training meningkat seiring bertambahnya epoch, output feedforward ANN pada microcontroller tetap konsisten dan sesuai dengan hasil inferensi di Python. Hal ini membuktikan bahwa proses deploy bobot ANN ke microcontroller berhasil tanpa mengubah perilaku sistem.

Tabel 7 Pengujian Epoch terhadap Performa Training dan Hasil Feedforward di Microcontroller

Epoch	Akurasi Training	Loss Training	Input Sensor	Output ANN (Python)	Output ANN (Microcontroller)
10	1,0000	4,8280	1, 0, 0, 1, 1	TURN AROUND	TURN AROUND
			0, 0, 1, 0, 0	FORWARD	FORWARD
			1, 0, 0, 0, 0	LEFT_MAX	LEFT_MAX
			0, 1, 0, 0, 0	LEFT_LOW	LEFT_LOW
			0, 0, 0, 1, 0	RIGHT_LOW	RIGHT_LOW
			0, 0, 0, 1, 1	RIGHT_MAX	RIGHT_MAX
30	1,0000	1,9868	1, 0, 0, 1, 1	TURN AROUND	TURN AROUND
			0, 0, 1, 0, 0	FORWARD	FORWARD
			1, 0, 0, 0, 0	LEFT_MAX	LEFT_MAX
			0, 1, 0, 0, 0	LEFT_LOW	LEFT_LOW
			0, 0, 0, 1, 0	RIGHT_LOW	RIGHT_LOW
			0, 0, 0, 1, 1	RIGHT_MAX	RIGHT_MAX
50	1,0000	1,3245	1, 0, 0, 1, 1	TURN AROUND	TURN AROUND
			0, 0, 1, 0, 0	FORWARD	FORWARD
			1, 0, 0, 0, 0	LEFT_MAX	LEFT_MAX
			0, 1, 0, 0, 0	LEFT_LOW	LEFT_LOW
			0, 0, 0, 1, 0	RIGHT_LOW	RIGHT_LOW
			0, 0, 0, 1, 1	RIGHT_MAX	RIGHT_MAX
100	1,0000	0,0043	1, 0, 0, 1, 1	TURN AROUND	TURN AROUND
			0, 0, 1, 0, 0	FORWARD	FORWARD
			1, 0, 0, 0, 0	LEFT_MAX	LEFT_MAX

			0, 1, 0, 0, 0	LEFT_LOW	LEFT_LOW
			0, 0, 0, 1, 0	RIGHT_LOW	RIGHT_LOW
			0, 0, 0, 1, 1	RIGHT_MAX	RIGHT_MAX
200	1,0000	0,0011	1, 0, 0, 1, 1	TURN AROUND	TURN AROUND
			0, 0, 1, 0, 0	FORWARD	FORWARD
			1, 0, 0, 0, 0	LEFT_MAX	LEFT_MAX
			0, 1, 0, 0, 0	LEFT_LOW	LEFT_LOW
			0, 0, 0, 1, 0	RIGHT_LOW	RIGHT_LOW
			0, 0, 0, 1, 1	RIGHT_MAX	RIGHT_MAX

3.4 Jadwal Pelaksanaan

Tabel 8 Jadwal Kegiatan

Kegiatan	Juni	July	Agst	Sept	Oct	Nov	Dec
Studi Literatur							
Perancangan Sistem							
Perakitan dan Implementasi Prototipe							
Pelatihan dan Integrasi ANN							
Pengujian dan Evaluasi Robot							
Revisi Sistem							
Penyusunan Laporan dan Dokumentasi							

3.5 Evaluasi dan Analisi Akhir

Data dari pengujian dikumpulkan dan dianalisis untuk mengevaluasi performa robot. Beberapa hal yang menjadi perhatian utama dalam evaluasi:

- Apakah setiap sensor inframerah mampu mendeteksi warna jalur (hitam/putih) secara konsisten sesuai dengan logika digital yang diharapkan (LOW/HIGH). Sensor yang tidak akurat dapat memengaruhi keputusan ANN secara keseluruhan.
- Apakah robot mampu mengikuti lintasan dengan bentuk lurus, belokan, dan persimpangan secara stabil dan presisi. Hal ini dilihat dari jumlah kesalahan (keluar jalur) serta keberhasilan robot menyelesaikan lintasan.
- Apakah terdapat keterlambatan sinyal atau ketidaksesuaian gerak motor saat ANN mengambil keputusan dari input sensor. Hal ini berpengaruh pada kelancaran dan kehalusan pergerakan robot.

Jika hasil pengujian menunjukkan ketidaksesuaian, maka dilakukan pelatihan ulang terhadap bobot jaringan ANN atau arsitektur jaringannya, diikuti pengujian ulang untuk melihat perbaikannya.

Bab 4. Hasil dan Pembahasan

4.1 Gambaran Umum Pengujian

Hasil yang ditampilkan meliputi proses pelatihan model ANN pada Python, evaluasi performa, konversi bobot ke mikrokontroler, serta pengujian robot secara langsung. Pembahasan diberikan untuk menjelaskan hubungan antara pengujian, kinerja sistem, dan interpretasi dari perilaku ANN dalam mengendalikan robot. Model ANN feedforward dilatih menggunakan data sensor IR pada lingkungan Python, kemudian bobot (weights) dan bias diekspor ke mikrokontroler untuk dijalankan secara real-time. Model ANN dilatih menggunakan dataset pembacaan sensor IR (5 input) yang dipasangkan dengan label aksi robot (6 kelas).

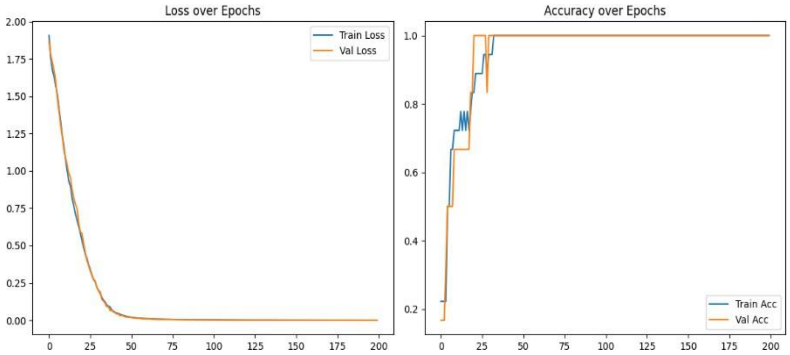
Tabel 9 Arsitektur Jaringan

Layer	Jumlah Neuron	Aktivasi
Input	5	–
Hidden Layer 1	8	ReLU
Hidden Layer 2	6	ReLU
Output Layer	6	Softmax

4.2 Hasil Pelatihan Model diPython

Pelatihan dilakukan menggunakan dataset sensor line follower melalui simulasi dan beberapa pengambilan data manual. Proses training menggunakan 80% data untuk train dan 20% data untuk validasi. Jadi, parameter pelatihan menggunakan epoch = 200, optimizer = adam, learning rate = 0,01, Loss function = Categorical Crossentropy, dan Batch size = 16.

Gambar 9 Grafik hasil train ANN dengan epoch 200



Pada grafik *Loss Over Epoch* menunjukkan bahwa model belajar secara efektif dan cepat menyesuaikan bobot untuk meminimalkan kesalahan. Sedangakan digrafik *Accuracy over epochs* menunjukkan peningkatan yang sangat cepat di sekitar epoch 50 dari nilai yang sangat rendah menjadi 1 atau bisa disebut akurasi 100%. Jadi, model pelatihan ANN belajar dengan cepat dan model ini mencapai kinerja yang hampir sempurna tidak ada tanda-tanda *overfitting*.

Gambar 10 Tabel parameter pada setiap lapisan model ANN sebagai bagian dari hasil analisa model

Layer (type)	Output Shape	Param #
dense_30 (Dense)	(None, 8)	48
dense_31 (Dense)	(None, 6)	54
dense_32 (Dense)	(None, 6)	42

Proses ini menjelaskan perhitungan parameter pada tiga lapisan Dense dalam model. Lapisan pertama memiliki delapan neuron dengan total 48 parameter, diperoleh dari perkalian jumlah fitur input dengan jumlah neuron, ditambah bias. Lapisan kedua terdiri dari enam neuron dengan 54 parameter hasil koneksi penuh dari delapan neuron sebelumnya beserta bias. Lapisan ketiga merupakan output

layer dengan enam neuron menggunakan aktivasi softmax dan memiliki 42 parameter yang berasal dari hubungan enam neuron pada lapisan sebelumnya ditambah bias. Semua parameter tersebut merupakan bobot dan bias yang disesuaikan selama pelatihan untuk menurunkan error dan meningkatkan akurasi prediksi.

4.3 Implementasi FeedForward ANN pada Microcontroller

Model yang dilatih diekspor dalam bentuk file bobot (ANN_WEIGHTS.h) kemudian ditanamkan pada microcontroller Arduino Uno. Proses FeedForward berjalan dengan struktur yang sama seperti ditraining. Berikut proses FeedForward di Microcontroller :

Hidden Layer 1 $h1_j = RELU\left(\int_{i=1}^{5} (x_i . W1_{ij}) + b1_j\right)$

Hidden Layer 2 $h2_j = RELU\left(\int_{i=1}^{8} (h1_i . W2_{ij}) + b2_j\right)$

Output Layer $o_k = \int_{j=1}^{6} (h2_j . W3_{jk}) + b3_k$

Hasil output masih berupa logit (nilai mentah ANN).

Lalu Softmax menghitung Probabilitas $P_k = \frac{e^{ok}}{\int e^{ok}}$

4.4 Pembahasan Error dan Akurasi Sistem ANN

Pengujian error pada sistem Artificial Neural Network (ANN) dilakukan untuk mengevaluasi tingkat kesesuaian antara keluaran jaringan saraf dengan target aksi yang diharapkan. Pengujian ini bertujuan untuk membuktikan bahwa model ANN hasil pelatihan di Python dapat diimplementasikan dengan baik pada mikrokontroler Arduino melalui proses feedforward. Berikut rumus menghitung nilai MAE pada sistem ini :

1. Error per data uji $Error_i = T - Y_{kelas\ target}$
   T = Nilai Target
   Y = Probabilitas kelas target
2. Hitung Nilai Mean Absolute Error (MAE) $MAE = \frac{\sum_{i=1}^{N} |T-Y|}{N}$

N = jumlah data uji

Berdasarkan hasil pengujian, sebagian besar data menghasilkan nilai error yang sangat kecil, mendekati nol. Hal ini menunjukkan bahwa ANN mampu memberikan probabilitas yang tinggi pada kelas yang benar. Beberapa kondisi tertentu menunjukkan nilai error yang relatif lebih besar, yang umumnya terjadi pada pola input sensor yang ambigu atau mendekati batas antar kelas. Namun demikian, ANN tetap mampu memilih kelas dengan probabilitas tertinggi yang sesuai dengan aksi pergerakan robot.

Tabel 10 Input Manual pada Python

No	Input Manual	Target Aksi	Output ANN	Nilai MAE
1	0, 0, 1, 0 ,0	FORWARD	0,9996	0,1999
2	0, 1, 1, 1, 0	FORWARD	0,9994	0,3998
3	1, 0, 0, 0, 0	LEFT_MAX	0,9987	0,1999
4	1, 1, 0, 1, 0	LEFT_LOW	0,9619	0,1076
5	0, 1, 0, 0, 0	LEFT_LOW	0,9972	0,000552
6	0, 1, 0, 1, 1	RIGHT_MAX	0,9983	0,200339
7	0, 0, 0, 1, 1	RIGHT_MAX	0,9976	0,1000011
8	0, 0, 1, 1, 0	RIGHT_LOW	0,9990	0,1000019
9	1, 1, 1, 1, 1	TURN_AROUND	0,9997	0,5000000
10	1, 1, 0, 0, 0	LEFT_MAX	0,9988	0,299776

Pada pengujian manual di *microcontroller* MAE tersebut dihitung dari selisih absolut antara nilai target aksi dan nilai probabilitas keluaran ANN tertinggi pada setiap data uji. Nilai MAE yang relatif kecil menunjukkan bahwa hasil prediksi feedforward ANN pada microcontroller memiliki tingkat kesesuaian yang baik terhadap target aksi yang diharapkan.

Tabel 11 Input manual pada Microcontroller

No	Input Manual	Target Aksi	Output ANN	Nilai MAE
1	0, 0, 1, 0 ,0	FORWARD	0,9998	0,0002
2	0, 1, 1, 1, 0	FORWARD	0,9999	0,0001
3	1, 0, 0, 0, 0	LEFT_MAX	0,9974	0,0026
4	1, 1, 0, 1, 0	LEFT_MAX	0,8991	0,1009
5	0, 1, 0, 0, 0	LEFT_LOW	0,9995	0,0005
6	0, 1, 0, 1, 1	RIGHT_MAX	0,9529	0,0471
7	0, 0, 0, 1, 1	RIGHT_MAX	0,9997	0,0003
8	0, 0, 1, 1, 0	RIGHT_LOW	1,0000	0
9	1, 1, 1, 1, 1	TURN_AROUND	1,0000	0
10	1, 1, 0, 0, 0	LEFT_MAX	0,9998	0,0002

Hasil Hasil tersebut membuktikan bahwa implementasi feedforward ANN pada mikrokontroler telah berjalan dengan baik dan konsisten dengan model hasil pelatihan. Dengan nilai error yang kecil dan stabil, sistem ANN dinilai layak digunakan sebagai pengendali arah gerak robot line follower secara real-time dan mampu mengambil keputusan dengan tingkat akurasi yang tinggi.

4.5 Hasil Pengujian Keseluruhan

A. Kondisi Lampu Menyala, Pada kondisi ini lintasan diuji dalam keadaan ruangan terang dengan pencahayaan lampu.

$$\%Keberhasilan = \frac{Jumlah\ Keberhasilan}{Jumlah\ Percobaan}\ x\ 100\%$$

Percobaan 1 $\%Keberhasilan = \frac{9}{10}\ x\ 100\% = 90\%$

Percobaan 2 $\%Keberhasilan = \frac{10}{10}\ x\ 100\% = 100\%$

B. Kondisi Lampu Mati Total,Pada kondisi ini lintasan diuji dalam keadaan gelap tanpa sumber cahaya tambahan. Sensor IR bekerja sepenuhnya menggunakan pancaran inframerah internal.

Percobaan 1 $\%Keberhasilan = \frac{8}{10}\ x\ 100\% = 80\%$

$$\text{Percobaan 2 } \%Keberhasilan = \frac{90}{10} x\ 100\% = 90\%$$

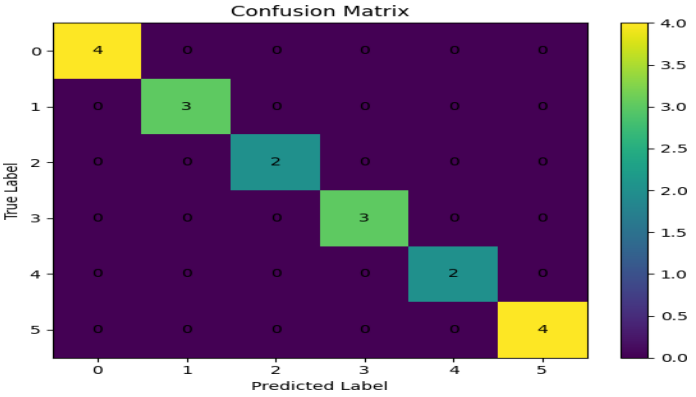
Jadi dari seluruh percobaan yang dilakukan pada kedua kondisi pencahayaan, diperoleh persentase keberhasilan sebagai berikut:

$$\%\text{Keberhasilan} = \frac{90\%+100\%+80\%+90\%}{4} \% = 90\%$$

Dengan demikian, tingkat keberhasilan keseluruhan robot line follower berbasis Artificial Neural Network (ANN) adalah 90%.

4.6 Confusion Matrix Hasil Klasifikasi

Confusion matrix pada gambar tersebut menunjukkan kinerja model Artificial Neural Network (ANN) dalam mengklasifikasikan enam kelas aksi robot. Setiap baris merepresentasikan label sebenarnya (true label), sedangkan setiap kolom menunjukkan label hasil prediksi. Nilai yang berada pada diagonal utama seluruhnya bernilai positif, sementara nilai di luar diagonal bernilai nol, yang menandakan bahwa seluruh data uji berhasil diklasifikasikan dengan benar tanpa adanya kesalahan prediksi. Hal ini menunjukkan bahwa model ANN memiliki tingkat akurasi yang sangat tinggi, di mana setiap kelas dapat dikenali secara tepat sesuai targetnya. Dengan demikian, confusion matrix ini membuktikan bahwa sistem kendali berbasis ANN mampu melakukan klasifikasi aksi robot secara konsisten dan andal.



Gambar 11 Coefision Matrix

4.7 Pembahasan

Pada tahap *machine learning*, model ANN dilatih menggunakan dataset hasil pembacaan sensor garis yang telah diberi label sesuai kelas gerakan robot (forward, left_low, left_max, right_low, right_max, dan turn_around). Proses pelatihan meliputi normalisasi data, pemilihan arsitektur jaringan (5–8–6–6), serta penggunaan fungsi aktivasi ReLU pada lapisan tersembunyi dan softmax pada lapisan output. Hasil pelatihan kemudian dievaluasi melalui akurasi, grafik loss, dan performa generalisasi sehingga bobot (*weights*) dan *bias* terbaik dapat diekstraksi untuk digunakan pada mikrokontroler.

Tahap berikutnya adalah implementasi model ke dalam mikrokontroler menggunakan proses *feedforward*. Semua bobot dan *bias* yang diperoleh dari Python disalin ke dalam berkas header agar dapat digunakan langsung oleh program Arduino. Proses ini berlangsung cepat karena hanya menggunakan operasi aritmetika dasar tanpa pelatihan ulang di sisi mikrokontroler.

Kedua tahapan ini menunjukkan bahwa pemisahan antara proses pelatihan di Python dan proses inferensi di mikrokontroler memberikan solusi yang efisien dan praktis. Mikrokontroler tidak dibebani proses pelatihan yang kompleks, namun tetap dapat menjalankan model ANN secara akurat. Secara keseluruhan, implementasi ini membuktikan bahwa model ANN berlapis banyak dapat dijalankan secara optimal pada perangkat tertanam dengan sumber daya terbatas, sekaligus mampu meningkatkan kemampuan robot dalam mengenali pola garis dan mengambil keputusan secara responsif.

Bab 5. Penutup

5.1 Simpulan

Berdasarkan keseluruhan rangkaian penelitian yang telah dilakukan, dapat disimpulkan bahwa penggunaan jaringan saraf tiruan jenis feedforward sebagai pengendali gerak pada robot line follower mampu menghasilkan respon navigasi yang lebih adaptif dibandingkan pendekatan logika berbasis kondisi (rule-based). Model ANN yang telah dilatih menggunakan data sensor pada lingkungan simulasi Python menunjukkan kemampuan mengenali pola kondisi garis secara lebih konsisten, terutama pada variasi posisi garis yang biasanya sulit ditangani oleh metode konvensional. Implementasi bobot dan bias hasil pelatihan ke dalam mikrokontroler melalui proses feedforward juga terbukti dapat dijalankan dengan baik meskipun keterbatasan kapasitas komputasi Arduino. Robot mampu mengeksekusi enam jenis aksi (maju, belok kiri rendah, belok kiri maksimum, belok kanan rendah, belok kanan maksimum, dan putar balik) berdasarkan probabilitas keluaran softmax yang dihitung secara real-time. Dengan demikian, integrasi metode pembelajaran mesin ke dalam sistem tertanam (embedded system) menunjukkan potensi yang cukup besar untuk meningkatkan kecerdasan perilaku robot sederhana tanpa memerlukan perangkat keras yang kompleks.

5.2 Saran

Penelitian ini masih memiliki ruang pengembangan agar performa robot line follower dapat ditingkatkan lebih lanjut. Untuk tahap berikutnya, disarankan memperluas jumlah data pelatihan terutama pada kondisi lingkungan yang lebih beragam, seperti tikungan tajam, percabangan, kondisi cahaya berbeda, dan permukaan lintasan yang berbeda warna, sehingga model ANN dapat beradaptasi diberbagai skenario nyata. Dari sisi perangkat keras, penggunaan sensor yang memiliki resolusi lebih tinggi atau tambahan sensor pendukung dapat membantu ANN memperoleh representasi kondisi garis yang lebih akurat. Selain itu, pengujian pada mikrokontroler dengan kemampuan komputasi yang lebih tinggi dapat menjadi alternatif untuk mengakomodasi model ANN yang lebih kompleks apabila dibutuhkan. Terakhir, proses evaluasi performa robot sebaiknya dilakukan secara kuantitatif, seperti menghitung tingkat keberhasilan mengikuti lintasan atau waktu tempuh, sehingga pengaruh penggunaan ANN dapat diukur secara lebih objektif dalam konteks peningkatan kinerja sistem.

Daftar Pustaka

[1] S. Saadatmand, S. Azizi, M. Kavousi, and D. C. Wunsch, "Autonomous Control of a Line Follower Robot Using a Q-Learning Controller," 2020.

[2] S. K. Risandriya and J. A. Tarigan, "OPTIMASI SENSOR PADA ROBOT LINE FOLLOWER DENGAN JENIS LAPANGAN BERBEDA DENGAN METODE NEURAL NETWORK".

[3] M. Education, "Microcontroller based line follower robot 1," vol. 11, no. 3, pp. 2730–2735, 2020.

[4] O. Erbay, "Line Follower Robot with PID Control," no. January, 2024.

[5] H. M. Leal, R. S. Barbosa, and I. S. Jesus, "Control of a Mobile Line-Following Robot Using Neural Networks," pp. 1–26, 2025.

[6] C. Minaya, R. Rosero, M. Zambrano, and P. Catota, "APPLICATION OF MULTILAYER NEURAL NETWORKS FOR CONTROLLING A LINE - FOLLOWING ROBOT IN ROBOTIC COMPETITIONS Abstract :," vol. 18, no. September 2023, 2024, doi: 10.14313/JAMRIS/1.

[7] A. Latif, H. A. Widodo, R. Rahim, and K. Kunal, "Implementation of Line Follower Robot based Microcontroller ATMega32A," no. February, pp. 15–20, 2020, doi: 10.18196/jrc.1316.

[8] R. Beresford, "Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research," vol. 22, pp. 717–727, 2000.

[9] T. M. Mitchell, *Machine Learning*.

[10] Trivusi, "Algoritma Feedforward Neural Network: Pengertian dan Cara Kerjanya." [Online]. Available: https://www.trivusi.web.id/2022/07/algoritma-feedforward-neural-network.html#

[11] W. Pav and J. Campa, "Intelligent Control System for Line Follower Robot : From Theory to implementation on the mechatronics field," no. November 2023, 2024, doi: 10.1109/DASC/PiCom/CBDCom/Cy59711.2023.10361359.

[12] R. A. Sarhan and M. S. Hassan, "Arduino-based implementation of kinematics for a 4 DOF robot manipulator using artificial neural network ARDUINO-BASED IMPLEMENTATION OF KINEMATICS FOR A 4 DOF ROBOT," no. February, 2024, doi: 10.29354/diag/184235.

[13] A. Salim and W. S. Pambudi, "IMPLEMENTASI METODE HYBRID ARTIFICIAL NEURAL NETWORK (ANN) – PID UNTUK PERBAIKAN PROSES BERJALAN PADA PROTOTYPE ROBOT MATERIAL HANDLING," vol. 1, no. 3, pp. 155–164, 2015.

[14] I. Pramana and D. Futra, "Implementasi Algoritma Q Learning Pada Robot Line Follower," pp. 1–6, 2021.

[15] A. E. Muñoz-zavala, J. E. Macías-díaz, and D. Alba-cuéllar, "A

Literature Review on Some Trends in Artificial Neural Networks for Modeling and Simulation with Time Series," pp. 1–45, 2024.