OPTIMASI SENSOR PADA ROBOT *LINE FOLLOWER* DENGAN JENIS LAPANGAN BERBEDA DENGAN METODE *NEURAL NETWORK*

Sumantri K.Risandriya[1], Jecky A Tarigan

*Politeknik Negeri Batam
Mechatronics Engineering Study Program
Parkway Street, Batam Centre, Batam 29461, Indonesia
E-mail: sumantri@polibatam.ac.id

Abstrak

Robot line follower adalah sebuah robot yang dapat bergerak dan berjalan mengikuti sebuah *track* garis. Namun *robot line follower* yang dibuat saat ini masih melakukan *tuningrobot* secara manual. Untuk mengoptimalkan kerja sensor pada *robot line follower*, digunakan metode *Neural Network* atau jaringan saraf tiruan. Dimana setiap data dari sensor dibandingkan, sehingga dapat dikenali perbedaan setiap sampel dari data yang di ambil. Parameter yang digunakan yaitu 4 *input*, 8 *hidden layer* 1, 10 *hidden layer* 2, dan 8 *output*. Dan juga digunakan 1 jenis *track* lapangan dengan warna dasar lapangan berbeda, pada *track* digunakan 2 warna yaitu *leaf green* dan *blue*, dimana antara kedua kondisi lapangan tersebut didapat data yang berbeda. Hasil akhir yang didapatkan berupa robot yang mampu bergerak dan berjalan pada kondisi lapangan yang berbeda dengan tingkat keberhasilan yaitu 80% dari 40 kali percobaan.

Kata kunci :robot *line follower,* sensor garis, *neural network.*

Abstract

Line follower robot is a robot that can move to follow a track line, but line follower robot made still tuning manually. To optimize the sensor in line follower robot, used methods Neural Network. Any data from sensor will be compared, so that it can recognize the difference each sample of data taken. The parameters used are 4 inputs, 8 hidden layer 1, 10 hidden layer 2 and 8 outputs. And used 1 type of track filed by field basis different colors: leaf green and blue, where the field conditions are obtained different data. The final result is robot that can move on a different field conditions with a success rate is 80% of 40 trials.

Key words :line follower robot, line sensor, neural network.

1. Pendahuluan

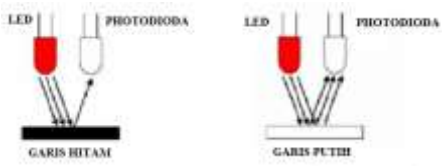
Seiring dengan kemajuan jaman, tidak dapat dipungkiri manusia semakin menginginkan hal yang bersifat automatisasi, terutama pada elektronika dan sistem kontrol. Hal ini ditandai oleh banyaknya bermunculan aplikasi-aplikasi kontrol baik melalui *interface computer* ataupun dalam bentuk modul-modul rangkaian. Salah satunya dalah penerapan sistem control kendali cerdas, sebagai contoh mengoptimalkan kerja sensor pada robot line follower dengan menggunakan metode *neural network*.

Pada umumnya robot *line follower* dalam melakukan tuning pada sensor masih menggunakan cara manual, sehingga bila kondisi track atau kondisi disekitar berubah maka sensor robot perlu di tuning kembali. Dari permasalahan demikian diperlukan metode yang baik agar sensor pada robot tidak pelu di tuning berulang-ulang.

2. Dasar Teori

2.1 Sensor Garis

Dalam sebuah sensor garis terdapat LED (*Light Emmiting Diode*) yang berfungsi sebagai *transmitter* dan *photodiode* yang berfungsi sebagai *receiver*. Ketika LED memancarkan cahaya ke bidang berwarna putih, cahaya akan dipantulkan hampir semuanya oleh bidang berwarna putih tersebut. Sebaliknya, ketika LED memancarkan cahaya ke bidang berwarna gelap atau hitam, maka cahaya akan banyak diserap oleh bidang gelap tersebut, sehingga cahaya yang sampai ke sensor (photodioda atau LDR) sedikit

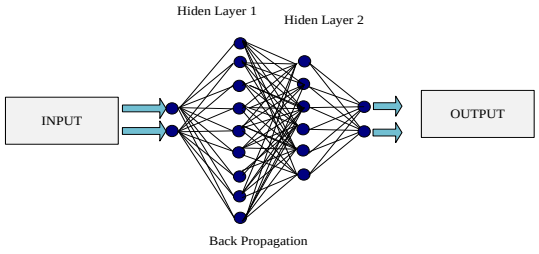


Gambar 1 Mekanisme Pemantulan Cahaya Sensor Garis[1]

Prinsip kerja dari sensor garis yaitu pada saat *photodiode* tidak terkena cahaya, maka nilai resistansi yang didapat akan besar atau dapat diasumsikan tak hingga. Sehingga tidak terdapat arus yang mengalir menuju komparator. Dan pada saat *photodiode* terkena cahaya, maka nilai resistansi yang didapat akan kecil[1]

2.2 NEURAL NETWORK

Jaringan Saraf Tiruan (JST) tau disebut juga *Neural Network* (NN), *Neural Network* dapat digunakan untuk memodelkan hubungan yang kompleks antara input dan output untuk menemukan pola-pola pada data. Pada penelitian ini metode yang digunakan *Training Back-Propagation*, *Training Back-Propagation*menggunakan *error*output untuk mengubah nilai bobot dalam arah mundur (*backward*). Untuk mendapatkan error, tahap perambatan maju (*forward*) harus dikerjakan terlebih dahulu.



Gambar 4 *Multi-Layerperceptron*[2]

Untuk perhitungan menggunakan *Training Back-Propagation* dapat menggunakan algoritma berikut :

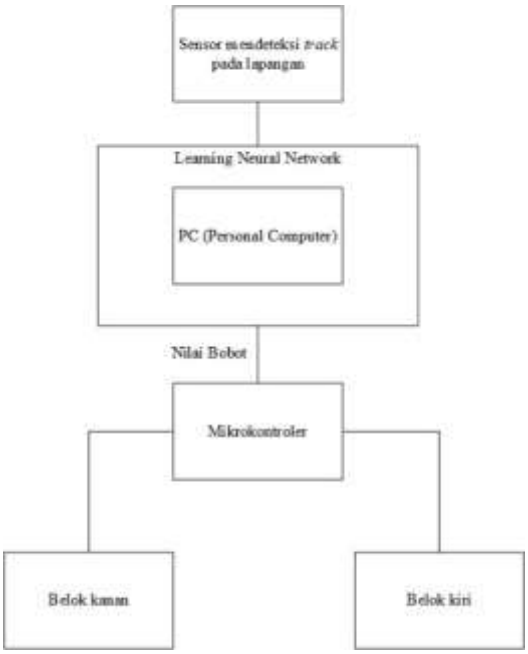
- Step 1 (Inisialisasi)
  Menset semua nilai bobot dan level *threshold* dari bilangan *random* yang terdistribusi secara beragam dalam interval kecil.
- Step 2 (Aktivasi)
  Mengaktifkan jaringan saraf Back-Propagation dengan memakaikan input $X_1(P), X_2(P), \dots, X_n(P)$ dan output yang diinginkan$X_{d.1}(P), X_{d.2}(P), \dots, X_{d.n}(P)$.
  Menghitung output aktual dari *neuron* pada *layer hidden* :
  $Y_i(p) = sigmoid\left[\sum_{i=1}^{n} X_i(P)\ x\ W_{ij}(P) - \theta_i\right]$(1)
  Menghitung *output* aktual dari *neuron* pada *layer output* :
  $Y_i(p) = sigmoid\left[\sum_{j=1}^{m} X_{jk}(P)\ x\ W_{jk}(P) - \theta_k\right]$(2)
- Step 3 (*Training* Bobot)
  Mengupdate bobot dari nilai *error* yang dirambatkan kearah belakang (*backwards*) bersesuaian dengan *neuron output*. Menghitung gradien *error* dari neuron pada *layer ouput* :
  $\delta_k(P) = Y_k(P)\ x\ [1 - Y_k(P)]\ x\ e_k(P)$
  (3)
  $e_k(P) = Y_{d.k}(P) - Y_k(P)$ (4)
  Mengupdate bobot dari *neuron output* :
  $U_{ji}^{(s)}(k+1) = U_{ji}^{(s)}(k) + U^{(s)}\delta_j^{(s)}X_{out,i}^{(s-1)}$ (5)
  Menghitung gradien error dari *neuron* pada *layer hidden* :
  $\delta_j(P) = Y_j(P)\ x\left[1 - Y_j(P)\right]\ x\ \sum_{k=1}^{l}\delta_k(P)\ x\ w_{jk}\ (P)$ (6)
  Mengupdate bobot dari *neuron hidden* :
  $w_{ji}^{(s)}(k+1) = w_{ji}^{(s)}(k) + U^{(s)}\delta_j^{(s)}X_{out,i}^{(s-1)}$ (7)
- Step 4 (Iterasi)
  Naikkan iterasi p dengan 1, kembali ke step 2 dan ulangi proses sampai kriteria error yang terpilih sudah terpenuhi.
- Update Momentum
  Pada update momentum, weight di-update menggunakan persamaan sebagai berikut[2] :
  $w_{ji}^{(s)}(k+1) = w_{ji}^{(s)}(k) + \mu^{(s)}\left[\delta_j^{(s)}(k)x_{out,i}^{(s-1)}(k) + \alpha\delta_j^{(s)}(k - 1)x_{out,i}^{(s-1)}(k-1)\right]$ (8)

Gambar 6 Transkip *Neural Network*

3. Perancangan Sistem

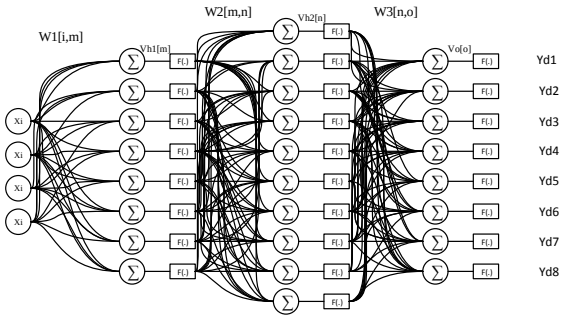
3.1 Rancangan Penelitian

Untuk proses optimasi sensor *robot line follower* pada jenis lapangan berbeda dapat digambarkan pada diagram blok seperti pada gambar 5.



Gambar 5 Diagram Blok Proses Optimasi Sensor Garis pada *Robot Line Follower*

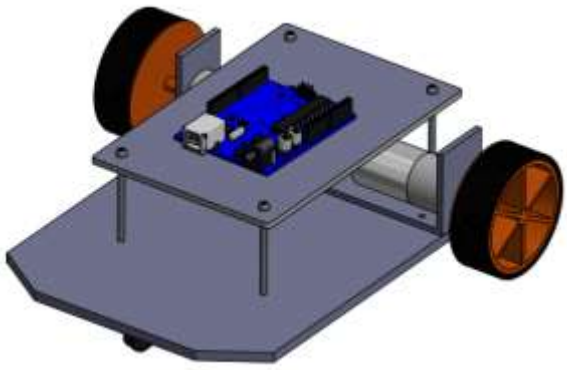
sensor garis pada *robot line follower* mendeteksi *track* pada lapangan. Dimana hasil pembacaan sensor tersebut menjadi masukan mikrokontroler dan data masukan dari sensor pada mikorokontroler tersebut digunakan untuk inputan pada *software* C#. Data yang diterima oleh *software* C# dijadikan sebagai masukan pada *Neural Network.* Hasil data tersebut dilatih pada *Neural Network*, sehingga didapatkan nilai bobot yang sesuai.



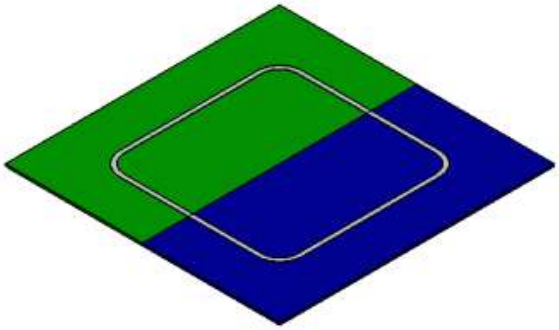
Tabel 1 Target *Neural Network*

No	Kondisi Sensor				Target Output								Ket
1	0	0	0	1	0	0	0	0	0	0	0	1	Pivot Kiri
2	0	0	1	1	0	0	0	0	0	0	1	0	Kiri 1
3	0	0	1	0	0	0	0	0	0	1	0	0	Kiri 2
4	0	1	1	0	0	0	0	0	1	0	0	0	Lurus
5	0	1	0	0	0	0	0	1	0	0	0	0	Kanan 2
6	1	1	0	0	0	0	1	0	0	0	0	0	Kanan 1
7	1	0	0	0	0	1	0	0	0	0	0	0	Pivot Kanan
8	0	0	0	0	1	0	0	0	0	0	0	0	Hitam

Pada gambar 5 merupakan penggunaan metode *Neural Network* dengan menggunakan empat pasang sensor garis. Diharapkan menghasilkan data yang diinginkan. Dan juga menggunakan 2 *hidden layer*, dimana *hidden layer* 1 memiliki 8 *node* dan *hidden layer* 2 memiliki 10 *node*. Hasil keluarannya digunakan untuk menggerakkan motor DC.



Gambar 7 Desain Mekanik *Robot Line Follower*



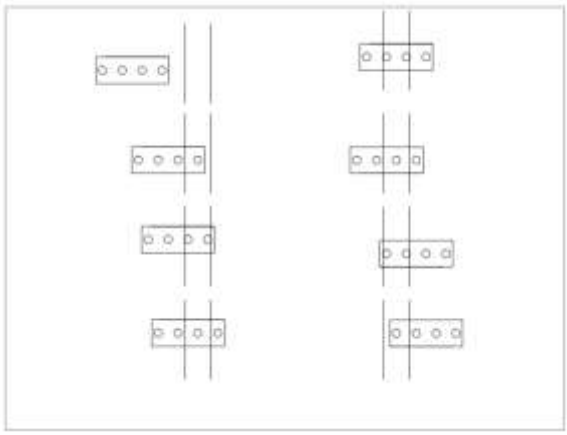
Gambar 8 Desain *Track* Lapangan

Pada gambar 7 merupakan desain *robot line follower* yang terdiri dari *baserobot line follower* dan letak dari mikrokontroler, dimana *base robot line follower* dibuat menggunakan bahan akrilik. Sensor garis diletakkan dibawah *robot line follower* dengan jarak antara sensor garis dengan *track* lapangan yaitu antara 1-3cm, sedangkan jarak antara sensor garis 1 dengan yang lain yaitu antara 1-2cm. Pada gambar 8 merupakan desain *track* lapangan yang akan dilalui oleh *robot line follower*. Dimensi lapangan yaitu 1.2m x 1.2m, dimana lapangan yang digunakan berwarna hijau dan biru dan ukuran garis sebagai *track robot line follower* ini yaitu 2cm dan warna track garis tersebut yaitu berwarna putih. Sehingga cahaya yang akan diterima oleh sensor garis lebih banyak karena sifat dari warna putih yaitu memantulkan cahaya, diharapkan sensor dapat membedakan antara *track* garis sebagai jalur yang harus dilewati dan lapangan sebagai *base*.

4. Hasil dan Analisa

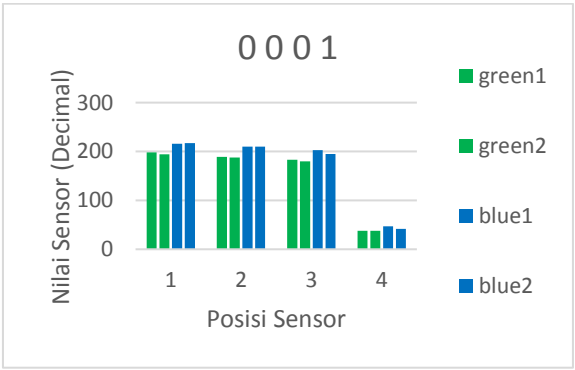
4.1 Data Sensor yang Digunakan Dalam Proses Learning

JumlahData yang digunakan dalam proses *learning* yaitu sebanyak 24 data, dimana data tersebut diambil sesuai dengan kondisi sensor. Terdapat 8 kondisi sensor yang digunakan untuk pengambilan data tersebut, kondisi sensor dapat dilihat pada gambar 9.

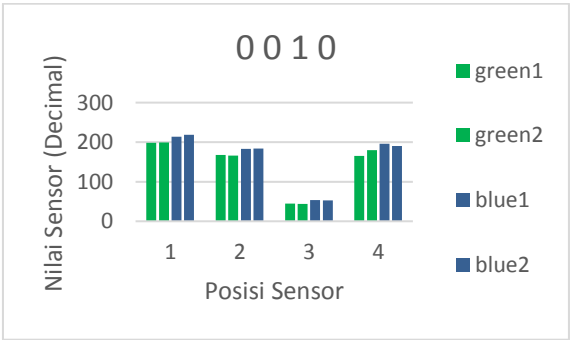


Gambar 9 Kondisi Sensor Dalam Pengambilan Data

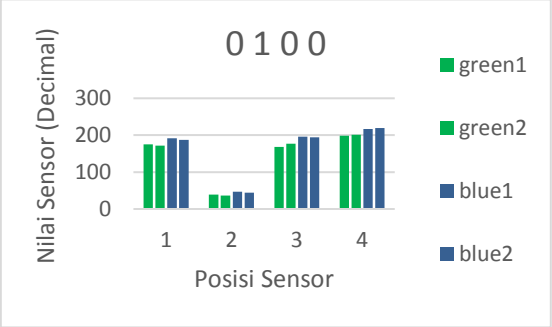
Data pembacaan sensor yaitu data analog dimana *range* data 0 – 1023, *range* data yang didapat dikonversikan menjadi 0 – 255. Terdapat 32 data yang akan dilatih, gambar 10 sampai 14 adalah sebagian data yang akan dilatih. grafik berwarna hijau menunjukkan kondisi data pada lapangan berwarna *leaf green* dan sebaliknya, grafik berwarna biru menunjukkan kondisi data pada lapangan berwarna biru. Pada gambar terdapat nilai 0 dan 1, itu menunjukkan bahwa pada kondisi 1 sensor mengenai track (garis berwarna putih) sedangkan kondisi 0 sensor mengenai *base* lapangan (*leaf green* dan *blue*).



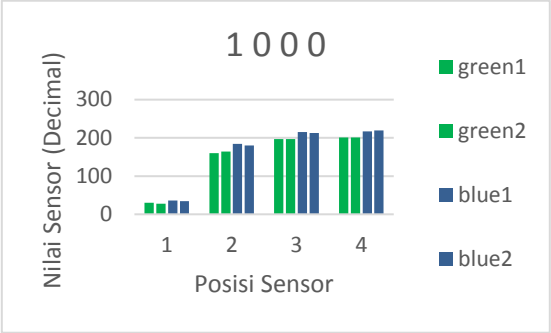
Gambar 10 Data pada Kondisi Sensor 0001



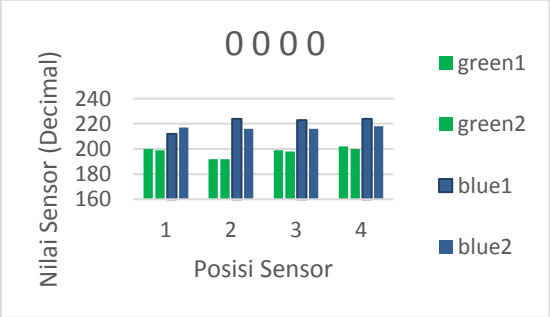
Gambar 11 Data pada Kondisi Sensor 0010



Gambar 12 Data pada Kondisi Sensor 0100

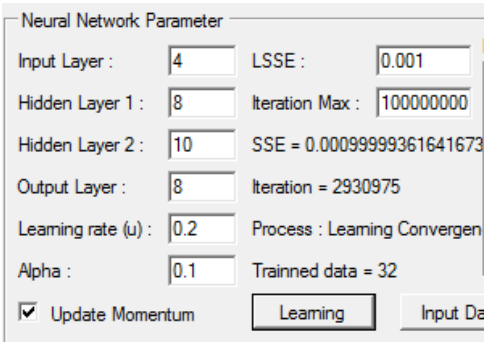


Gambar 13 Data pada Kondisi Sensor 1000



Gambar 14 Data pada Kondisi Sensor 0000

4.2 Parameter yang Digunakan pada Proses Pelatihan



Gambar 18 Parameter pada Proses Pelatihan

Pada gambar 18 dapat dilihat bahwa parameter yang digunakan yaitu 4 *input layer*, 8 *hidden layer* 1, 10 *hidden layer* 2, dan 8 *output layer*. Dan juga data yang dilatih sebanyak 32 data, hasil konvergensi didapat pada iterasi 2930975. Setelah konvergensi maka didapat nilai bobot atau *weight* yang diinginkan dan juga yang akan digunakan pada mikrokontroler.

4.3 Pengujian Alat Secara Keseluruhan

Berikut penyajian data hail uji lapangan ditampilkan dalam bentuk persentasi keberhasilan.

1. Pada *track* lapangan berwarna biru dan hijau dengan arah searah jarum jam.
   Percobaan 1 :
   $$\%\ Keberhasilan = \frac{Jumlah\ Keberhasilan}{Jumlah\ Percobaan}\ x\ 100\%$$
   $$\%\ Keberhasilan = \frac{7}{10}\ x\ 100\% = 70\%$$
   Percobaan 2 :
   $$\%\ Keberhasilan = \frac{Jumlah\ Keberhasilan}{Jumlah\ Percobaan}\ x\ 100\%$$
   $$\%\ Keberhasilan = \frac{10}{10}\ x\ 100\% = 100\%$$
2. Pada *track* lapangan berwarna biru dan hijau dengan arah berlawanan arah jarum jam.
   Percobaan 1 :
   $$\%\ Keberhasilan = \frac{Jumlah\ Keberhasilan}{Jumlah\ Percobaan}\ x\ 100\%$$
   $$\%\ Keberhasilan = \frac{7}{10}\ x\ 100\% = 70\%$$
   Percobaan 2 :
   $$\%\ Keberhasilan = \frac{Jumlah\ Keberhasilan}{Jumlah\ Percobaan}\ x\ 100\%$$
   $$\%\ Keberhasilan = \frac{10}{10}\ x\ 100\% = 100\%$$

Jadi dari semua percobaan yang dilakukan didapatkan persentasi keberhasilan :

$$\%\ Keberhasilan = \frac{70\% + 100\% + 70\% + 100\%}{3} = 80\%$$

5. Kesimpulan

Berdasarkan hasil dan analisa maka dapat disimpulkan bahwa :

1. Metode *neural network* dapat diterapkan pada *robot line follower*.
2. *Robot line follower* dapat bergerak mengikuti *track* garis dengan kondisi warna lapangan yang berbeda yaitu *leaf green* dan *blue*,
3. Dari percobaan yang dilakukan sebanyak 40 kali putaran didapatkan persentasi keberhasilan sebesar 80%.

Daftar Pustaka

[1] Yultrisna ST.,MT, Andi Syofian ST.,MT, "RANCANG BANGUN ROBOT *SOLVING MAZE* DENGAN ALGORITMA *DEPTH FIRST SEARCH*", Vol.15 No.2. Agustus 2013.

[2] Sumantri K. Risandriya. MK-4611 Pratikum Kendali Cerdas, Judul: "PENGANTAR JARINGAN SYARAF TIRUAN – MULTI PERCEPTRON." Mekatronika, Politeknik Negeri Batam, Batam, Januari. 06, 2014.