ISBN: 979-756-061-6

SIMULASI JARINGAN SYARAF TIRUAN BERBASIS METODE *BACKPROPAGATION* SEBAGAI PENGENDALI KECEPATAN MOTOR DC

Romi Wiryadinata[1)], Dwi Ana Ratnawati[2)]
Lab. Pemrograman Informatika Teori[1)], Lab. Software MATLAB T. Elektro[1) 2)],
Fakultas Teknologi Industri Universitas Islam Indonesia
Jl. Kaliurang Km 14.5 Yogyakarta
e-mail: [1)]romi_wiryadinata@yahoo.com, [2)]dwi_ana@fti.uii.ac.id

ABSTRAK

Motor DC dan komputer banyak digunakan dalam kehidupan sehari-sehari, baik di rumah tangga, industri maupun lingkungan pendidikan yang sangat membutuhkan ketelitian dan penggunan yang serba otomatis. Jaringan Syaraf Tiruan merupakan salah satu kendali motor DC yang dapat disimulasikan menggunakan neural network toolbox pada software Matlab 6.5. Dengan menggunakan metode Backpropagation dan fungsi Gradient Descent Momentum diperoleh struktur jaringan yang terbaik, terdiri dari 5 sel neuron lapisan input, 3 sel neuron lapisan tersembunyi dan 1 sel neuron lapisan output. Fungsi aktivasi pada setiap lapisan menggunakan fungsi identitas, dengan learning rate 0.1 dan momentum coefisient 0.9 menghasilkan Mean Square Error (MSE) 0.0070382. Persentase MSE pengujian adalah 1.645 %. Banyaknya jumlah data masukan berpengaruh terhadap banyaknya iterasi dan MSE yang dihasilkan. Penelitian ini juga membuktikan bahwa dasar teori pengaturan kecepatan motor DC metode Ward Leonard tentang penggunaan 2 motor dapat lebih efisien dengan Artificial Intellegence menggunakan Neural Network (Jaringan Syaraf Tiruan).

***Kata kunci**: Jaringan Syaraf Tiruan, Motor DC, MATLAB 6.5*

1. Pendahuluan

1.1 Latar Belakang

Seiring perkembangan jaman peran komputer semakin mendominasi kehidupan. Lebih dari itu, komputer saat ini diharapkan dapat digunakan untuk mengerjakan segala sesuatu yang bisa dikerjakan oleh manusia baik di rumah tangga, industri bahkan di lingkungan pendidikan. Untuk memecahkan masalah dengan komputer, program harus dibuat terlebih dahulu kemudian akan diproses selanjutnya. Tanpa program, komputer hanyalah sebuah kotak besi yang tidak berguna.

Motor DC banyak digunakan dalam kehidupan sehari-hari. Baik dalam dunia industri maupun rumah tangga. Motor DC yang beredar sebenarnya sudah menggunakan bahasa logika sederhana ada yang dikendalikan manual oleh manusia, sebagian sudah ada yang menggunakan *mikrokontroller*, algoritma *fuzzy* maupun algortima dan kendali lainya yang menggunakan bahasa pemrograman yang berbeda. Motor yang beredar di masyarakat akan lebih menghasilkan produk yang bagus dan memiliki tingkat presisi tinggi apabila kesalahan dari faktor manusia dapat diperkecil.

Dari beberapa pengendalian yang menggunakan algoritma *fuzz* dan *mikrokontroller* atau yang lainnya, *error* yang dihasilkan terlalu besar berkisar antara 3-10%.

1.2 Batasan Masalah

Batasan masalah yang dibahas pada penelitian ini adalah:

a. Motor DC dimodelkan dengan persamaan matematis.
b. Mencari arsitektur JST terbaik dari beberapa pelatihan.
c. Pembuatan sistem disimulasikan menggunakan perangkat lunak Matlab 6.5.
d. Pelatihan dan pengujian JST menggunakan fungsi yang terdapat dalam Matlab.

1.3 Tujuan Penelitian

Penelitian ini memiliki beberapa tujuan, yaitu:

a. Merancang dan mensimulasikan sebuah sistem penggerak cerdas dengan algoritma JST metode BP.
b. Mempelajari dan memanfaatkan *toolbox neural network* (NN) dan *simulink* pada Matlab sebagai media pelatihan dan simulasinya.
c. Membuat suatu pelatihan untuk menghilangkan atau memperkecil galat yang terjadi agar sistem JST dapat dikatakan sempurna.

2. Dasar Teori

2.1 Jaringan Syaraf Manusia sebagai Dasar Jaringan Syaraf Tiruan

Beberapa hal yang mendasari kerja Jaringan Syaraf Manusia (JSM), diantaranya mengenai penyimpanan informasi dan daya ingat, dimana bila suatu sinyal tertentu melalui sinapsis secara berulang-ulang, maka sinapsis tersebut menjadi lebih mampu menghantarkan sinyal pada

 ISBN: 979-756-061-6

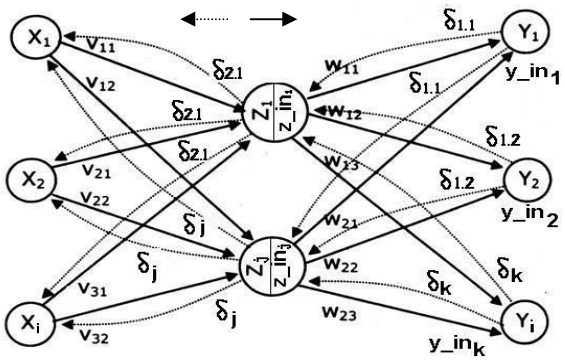
kesempatan berikutnya. Hal ini mendasari adanya proses belajar atau pelatihan (*learning*), jadi JST yang akan digunakan pasti melalui proses pelatihan secara berulang-ulang terlebih dahulu.

Dalam JSM, akson dan dendrit bercabang-cabang sedemikian banyaknya yang menunjukan bahwa adanya sistem paralel dan terdistribusi. Akson dan dendrit pada JSM bercabang-cabang dengan pola yang tidak teratur, sedangkan pada JST, keparalelan dan kedistribusian cabang-cabang itu membentuk pola tertentu.

JST merupakan bagian dari *Artificial Intelligence* (AI) yang berbasis hubungan, karena cara kerjanya melihat pada JSM. Secara garis besar dapat dijelaskan sebagai berikut: beberapa bongkol (baik eksitasi maupun inhibisi) masuk ke suatu neuron, oleh neuron masukan tersebut dijumlahkan, kemudian dibandingkan dengan nilai ambangnya. Hasil penjumlahan baru bisa berarti jika besarnya kecilnya bobot hubungan telah teratur.

2.2 Algoritma *Backpropagation*

JST *backpropagataion* atau rambat balik (JST-BP) adalah metode yang paling sederhana dan mudah dipahami dari metode-metode yang lain. JST-BP akan merubah bobot biasnya untuk mengurangi perbedaan antara *output* jaringan dan target *output*. Setelah pelatihan selesai, dilakukan pengujian terhadap jaringan yang telah dilatih. Pembelajaran algoritma jaringan syaraf membutuhkan perambatan maju dan diikuti dengan perambatan mundur. Keduanya dilakukan untuk semua pola pelatihan.



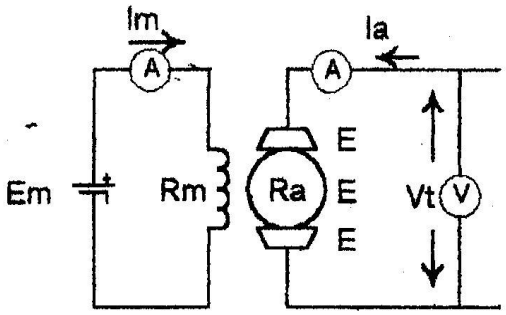
Gambar 1. Arsitektur algoritma *backpropagation*

2.3 Motor DC

Motor DC adalah sistem mesin yang berfungsi mengubah tenaga listrik arus searah menjadi tenaga gerak atau mekanis. Motor DC hampir dapat dijumpai di setiap peralatan baik rumah tangga, kendaraan bahkan dalam dunia industri sekalipun, dari yang beukuran mikro sampai motor yang memiliki kekuatan ribuan daya kuda.

2.3.1 Karakteristik Motor DC

Pada motor *shunt* eksitasi terpisah, bertambahnya kopel arus jangkar (Ia) mengakibatkan kecepatan (n) menurun. Pada motor seri, bertambahnya kopel (arus) akan menyebabkan bertambahnya harga fluks (ϕ), karena fluks pada motor seri merupakan merupakan fungsi Ia.



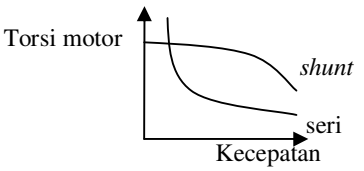
Gambar 2. Rangkaian ekivalen motor DC

$$E_a = V_t - I_a R_a, \quad (1)$$

$$E_a = C\, n\, \phi, \quad (2)$$

$$n = {}^{Vt - Ia\,Ra}/_{C\,\phi}, \quad (3)$$

Untuk harga Ia = 0, harga fluks juga nol sehingga dari persamaan 3, diperoleh harga n menuju tak terhingga. Sedangkan untuk harga Ia yang cukup besar, harga n akan mendekati nol. Dengan demikian karakteristik kecepatan-kopel untuk motor *shunt* dan seri dapat digambarkan sebagai berikut:



Gambar 3. Karakteristik kecepatan-kopel motor

2.3.2 Pengaturan Kecepatan Motor DC

Tiga parameter yang biasa diatur adalah:

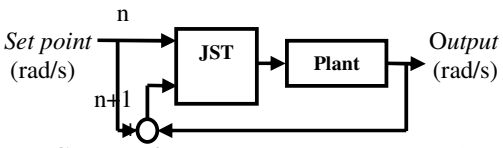
a. Medan *shunt* (ϕ), dengan menyisipkan tahanan variabel yang dipasang seri terhadap kumparan medan (motor *shunt*), maka dapat diatur arus medan dan fluksnya. Rugi panas yang ditimbulkan sangat kecil pengaruhnya. Karena besarnya fluks yang dicapai oleh kumparan medan terbatas, kecepatan yang diaturpun akan terbatas.
b. Tegangan (V_t), dikenal dengan metode *Ward Leonard*. Menghasilkan suatu pengaturan kecepatan yang sangat halus dan banyak dipakai untuk lift, mesin bubut dan lain-lain. Satu-satunya kerugian dalam sistem ini adalah biaya untuk penambahan generator dan penggerak awal.

ISBN: 979-756-061-6

c. Tahanan (R_a), dengan menyisipkan tahanan variabel terhadap tahanan jangkar. Cara ini jarang dipakai, karena penambahan tahanan seri terhadap tahanan jangkar menimbulkan rugi panas yang cukup besar.

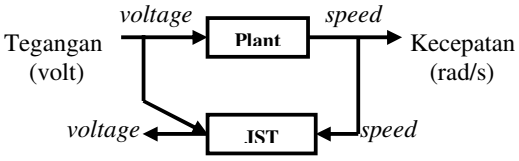
3. Perancangan Sistem

Proses belajar JST dilakukan secara *on-line/continue*, sehingga JST memerlukan hasil pengendaliannya (kecepatan yang dihasilkan motor) untuk memperbaiki tanggapan motor. Dalam perancangan sistem, masukan JST adalah berupa *set point* kecepatan, sedangkan keluarannya yang juga berfungsi sebagai masukan motor adalah tegangan DC. Sebagai keluaran motor dan sebagai hasil akhir dari sistem adalah kecepatan model motor. Untuk mengetahui lebih jelas perancangan sistem ini dapat dilihat pada diagram blok berikut:



Gambar 4. Blok diagram perancangan sistem

Pelatihan dari sistem pengendalian dirancang dengan menggunakan metode *inverse*, dimana masukan dari *plant* adalah sebagai target atau keluaran dari JST, sehingga skenario keluarannya akan digunakan kembali sebagai masukan. Karena pada pelatihan menggunakan metode *inverse*, maka masukan dan keluaran dari sistem kendali yang sebenarnya akan dibalik pada saat pelatihan. Pada saat pelatihan masukan dari JST adalah keluaran dari motor, yaitu kecepatan atau putaran dari motor, sedangkan keluaran atau target dari JST adalah merupakan masukan motor, yaitu tegangan.



Gambar 5. Model *inverse* pelatihan

Kemampuan dari JST akan dipergunakan untuk mengidentifikasikan motor. Selanjutnya hasil proses identifikasi digunakan pada proses pengendalian kecepatan motor. Perangkat lunak yang digunakan dalam perancangan sistem adalah Matlab 6.5 *release* 13, karena memiliki bahasa tingkat tinggi dan dapat digunakan untuk komputasi teknik, penghitungan, visualisasi dan pemrograman. Selain itu juga memiliki *neural network* (NN) *toolbox,* sehingga memudahkan dalam perancangan program JST maupun pensimulasian dari sistem yang telah dilatih. Beberapa kegunaan lain dari Matlab di antaranya adalah untuk pengembangan algoritma, pemodelan, simulasi dan pembuatan antarmuka GUI (*Graphical User Interface*).

4. Analisis dan Pembahasan

Berikut ini adalah tabel perbandingan yang didapat dari keadaan motor *real* dengan tegangan medan tetap (100 volt).

Tabel 1. Perbandingan kecepatan dan tegangan pada motor sebenarnya

Tegangan (volt)	*Kecepatan (rpm)*
150	**1913.3**
125	**1657**
100	**131.9**
75	**980.2**
50	**664**
25	**311.8**

Pada saat tegangan jangkar 150 volt kecepatan maksimum mencapai 1913.3 rpm, tetapi pada *data board* pada motor sebenarnya, kecepatan maksimum saat tegangan jangkar 150 volt adalah 1750 rpm. Hal ini banyak terjadi pada keadaan motor sebenarnya, yang disebabkan karena usia motor yang cukup lama dan penggunaan yang sering dilakukan, sehingga menyebabkan perubahan pada beberapa piranti pada motor yang sudah tidak sesuai dengan standarisasi pada saat motor diproduksi.

Dengan mengunakan data yang sama pada motor sebenarnya, data *input* dan *output* dari hasil simulasi disimpan kedalam *workspace* Matlab untuk dijadikan sebagai masukan dan target pada pelatihan JST sebagai pengendali motor DC. Pelatihan dengan menggunakan *for-while loops* kurang mendapatkan hasil yang lebih maksimum, disebabkan karena data *input* dan target JST terlalu banyak, kurang lebih sebanyak 150676 data *input* dan 150676 data *output.* Sebagai perbandingan, untuk melakukan 1000 iterasi dengan 1 HL dan 7 sel neuron menggunakan *for-while loops,* membutuhkan waktu kurang lebih selama 18 jam. Berbeda dengan pelatihan menggunakan fungsi *newff* yang disediakan oleh Matlab. Dengan menggunakan struktur JST yang sama, 1000 iterasi dapat dilakukan hanya dalam hitungan menit. Pelatihan dan pengujian JST menggunakan Matlab akan lebih cepat jika semua data *input, output* dan bobot-bias dijadikan kedalam bentuk perhitungan matrik seperti yang terdapat pada fungsi *newff.*

Hasil pelatihan terbaik adalah adalah pada tabel nomor 10. Pelatihan berhenti ketika iterasi yang ditentukan sudah tecapai dengan *Mean Square Error* (MSE) 0.0070382. MSE pada tabel

ISBN: 979-756-061-6

nomor 15 lebih kecil 2e-7 dari MSE pada tabel nomor 10. Tetapi jumlah iterasi 2 kali lebih besar dari iterasi pada tabel nomor 10, sehingga untuk pengujian jaringan akan digunakan struktur pada tabel nomor 10.

Pada tabel nomor 5, pelatihan berhenti karena *gradient* sudah mencapai target, artinya MSE yang dihasilkan sudah mencapai nilai yang paling minimum untuk arsitektur JST sebagai pengendali motor DC dengan 1 lapisan tersembunyi. Nilai *gradient* menggunakan nilai *default* fungsi *newff* yaitu 1e-10.

Nilai *gradient* yang dihasilkan dan ditampilkan akan selalu dipengaruhi oleh perubahan nilai MSE. Penentuan nilai *momentum coefisient* (MC) akan berpengaruh langsung kepada perubahan bobot.

Tabel 2. Hasil pelatihan terbaik dari masing-masing pengelompokan

No	*HL*	*Neuron*	*LR*	*MC*	*F.aktivasi*	*Iterasi*	*MSE*	*Ket.*
1	1	7–1	0.7	0.6	*logsig – logsig*	1000	0.00806	iterasi
2	1	7–1	0.6	0.8	*logsig – purelin*	1000	0.00707	iterasi
3	1	7–1	0.4	0.6	*logsig – tansig*	1000	0.00744	iterasi
4	1	7–1	0.8	0.9	*purelin – logsig*	1000	0.00805	iterasi
5	1	7–1	0.3	0.7	*purelin – purelin*	140	0.00705	*gradient*
6	1	7–1	0.6	0.5	*purelin – tansig*	78	0.00876	*gradient*
7	1	7–1	0.7	0.6	*tansig – logsig*	1000	0.00765	iterasi
8	1	7–1	0.4	0.7	*tansig – purelin*	1000	0.00717	iterasi
9	1	7–1	0.4	0.7	*tansig – tansig*	1000	0.00714	iterasi
10	**2**	**5–3–1**	**0.1**	**0.9**	***purelin – purelin – purelin***	**150**	**0.00703**	**iterasi**
11	2	5–3–1	0.1	0.9	*logsig – logsig – logsig*	500	0.01120	iterasi
12	2	5–3–1	0.1	0.9	*purelin – logsig – logsig*	500	0.00910	iterasi
13	2	5–3–1	0.5	0.9	*purelin – purelin – logsig*	300	0.00806	iterasi
14	2	5–3–1	0.5	0.9	*logsig – purelin – purelin*	300	0.00920	iterasi
15	4	7–4–3–2–1	0.3	0.9	*purelin – purelin – purelin purelin – purelin*	300	0.00703	iterasi

Purelin = Fungsi identitas atau linear,
Logsig = Fungsi Sigmoid biner,
Tansig = Fungsi Sigmoid Bipolar.

Fungsi aktivasi identitas dapat menghasilkan MSE hampir mendekati target, yang disebabkan karena *input* dan target dari JST memiliki nilai yang sebanding, hal ini sesuai dengan fungsi aktivasi identitas dimana masukan fungsi sama dengan keluarannya. Jumlah lapisan dan sel neuron pada masing-masing lapisan tersembunyi tidak berpengaruh besar terhadap MSE, kecuali jika variasi dari nilai *learning rate* (LR) dan MC yang digunakan sesuai dengan arsitektur JST. Tetapi hampir semua pelatihan yang menggunakan fungsi aktivasi *sigmoid* bipolar pada *hidden layer* (HL) dan lapisan keluaran tidak dapat mencapai target iterasi dan MSE, disebabkan karena fungsi aktivasi *sigmoid* bipolar memiliki nilai *range* antara 1 sampai -1. Sedangkan pada pelatihan, nilai *input* dan target sudah di normalisasi agar menghasikan kecepatan yang lebih cepat, sehingga target memiliki nilai antara 0 sampai 1.

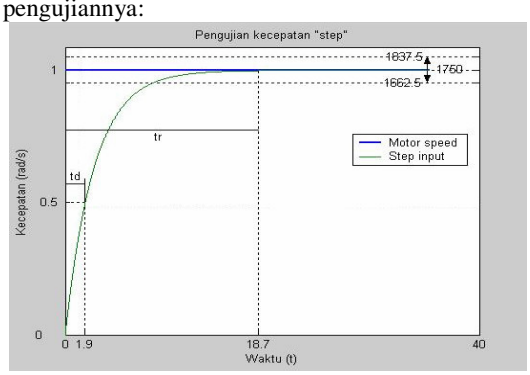
Nilai LR dan MC akan berpengaruh terhadap perubahan MSE pada setiap iterasi. Semakin besar nilai LR, akan semakin cepat pelatihan mendekati nilai *error* minimum, tetapi menghasilkan perubahan MSE yang tidak stabil. Jika nilai LR digunakan terlalu kecil, maka akan menyebabkan pelatihan lebih lama mendekati nilai error minimum yang ditentukan dan iterasi semakin besar. Berbeda dengan nilai MC, semakin kecil nilai MC yang digunakan maka semakin banyak iterasi yang dibutuhkan untuk mencapai *error* minimum. Sehingga nilai yang digunakan untuk pelatihan tidak terlalu besar dan tidak terlalu kecil, sesuai dengan variasi nilai antara LR dan MC.

Penentuan jumlah target iterasi (*epoch*) dilihat dari struktur pelatihan jaringan. Jika jaringan memiliki HL dan jumlah neuron yang banyak, maka target iterasi di set tidak terlalu besar agar pelatihan tidak menggunakan memori pada *personal computer* (PC) terlalu banyak. Semakin banyak jumlah lapisan dan jumlah sel neuron pada masing-masing lapisan, semakin banyak komputasi, semakin besar memori PC yang digunakan dan akan semakin lama waktu yang ditempuh untuk mencapai *error* minimum. Dari tabel diatas maka struktur jaringan yang akan digunakan adalah strukur jaringan pada tabel nomor 10. Dimana struktur jaringan terdiri dari 2 sel neuron *input*. *Input* pertama adalah *set point*, sel neuron kedua adalah perubahan dari kecepatan yang dihasilkan motor. Lapisan *input* (v) terdiri dari 5 sel neuron, sedangkan lapisan tersembunyi terdiri dari 2 lapisan. Lapisan tersembunyi pertama (w) terdiri dari 3 sel neuron dan sesuai dengan target sistem

ISBN: 979-756-061-6

JST, maka HL kedua atau lapisan output (w_out) terdiri dari 1 sel neuron. Fungsi aktivasi yang digunakan pada setiap lapisannya adalah fungsi identitas.

Struktur terbaik jaringan kemudian di uji dengan menggunakan masukan *step*. Perlu diketahui waktu (t) adalah dalam satuan detik Matlab, bukan dalam *real time*. Berikut ini adalah grafik hasil pengujiannya:

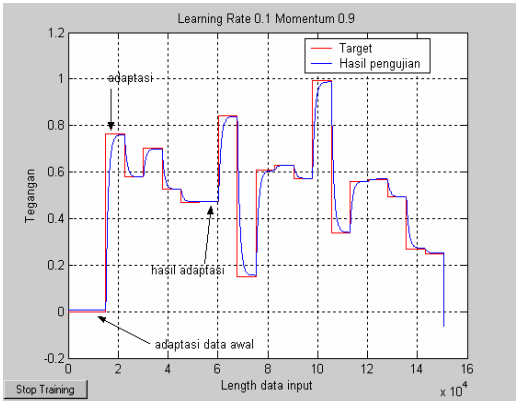


Gambar 6. Hasil pengujian menggunakan *step input*

Karakteristik *respon transien* dari JST sebagai pengendali kecepatan motor DC dengan menggunakan *step input* adalah sebagai berikut :

a. Waktu tunda (td) adalah waktu yang diperlukan oleh tanggapan untuk mencapai setengah (50%) dari nilai akhirnya yaitu selama 1.9 detik.
b. Waktu naik (tr) adalah waktu yang diperlukan oleh tanggapan untuk naik dari 0% menjadi 100% dari nilai akhir yaitu selama 18.7 detik.
c. Maksimum o*vershoot* (mp) adalah nilai puncak kurva tanggapan diukur dari satuan waktu, digunakan untuk mengukur kestabilan relatif dari sistem. Pada grafik tidak terlihat adanya *overshoot*, disebabkan karena pelatihan menggunakan LR yang kecil dan juga disebabkan karena karakteristik dari motor DC dimana kecepatan berbanding terbalik dengan torsi.
d. Waktu puncak adalah waktu yang diperlukan tanggapan untuk mencapai puncak atau maksimum *overshoot*. Karena tidak ada *overshoot*, maka waktu puncak (tp) juga tidak ada.
e. Waktu turun (ts) adalah waktu yang diperlukan untuk menanggapi kurva agar dapat mencapai dan tetap berada dalam persentase nilai akhir tertentu dan biasanya digunakan batasan 5%. Seperti telihat pada gambar 4.16 diatas, grafik kecepatan yang dihasilkan sudah stabil, sehingga waktu turun (ts) sama dengan waktu naik (tr).

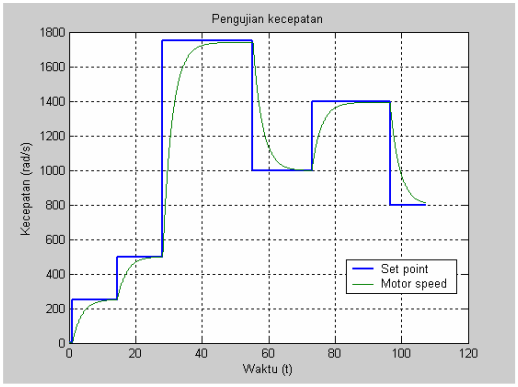
Kemudian pengujian jaringan dilakukan dengan data *offline* dan data *online*. Pengujian data *offline* dilakukan dengan data *input* menggunakan data masukan (tegangan) yang digunakan juga saat pelatihan. Sedangkan pengujian *online* dilakukan menggunakan *set point* Berikut ini adalah grafik hasil dari pengujian *offline*.



Gambar 7. Hasil pengujian *offline*

Grafik hasil dari pengujian (biru) sudah hampir mengikuti dari grafik target tegangan pelatihan (merah). Pada data awal, grafik pengujian menunjukan proses adaptasi yang akan diperbaiki untuk data berikutnya. Proses adaptasi akan terus dilakukan pada setiap awal perubahan nilai dari tegangan. Hasil pengujian tersebut membuktikan bahwa, pelatihan jaringan sudah cukup baik dan akan digunakan lebih lanjut dalam pengujian *online* menggunakan *toolbox simulink* pada Matlab dan pengujian menggunakan GUI (*Graphical User Interface*).

Pengujian *online* dilakukan dengan memberikan *set point* kecepatan (*input* pertama) dan menghasilkan keluaran jaringan berupa tegangan, tegangan inilah yang akan memberi masukan pada motor untuk menghasilkan keluaran berupa kecepatan motor. Kemudian kecepatan motor akan di-*feedback* menjadi *input* jaringan kedua. Hasil dari pengujian *online* dapat dilihat pada gambar berikut ini.



Gambar 8. Hasil pengujian *online*

Set point yang berupa kecepatan di masukan melalui blok *step*. Kecepatan dari motor mampu mengikuti kacepatan dari *set point*, hanya pada kecepatan-kecepatan tertentu motor tidak dapat

ISBN: 979-756-061-6

mengikuti, tetapi dengan selisih yang cukup kecil. Sesuai dengan karakteristik dari motor DC (gambar 3), kecepatan yang dihasilkan motor DC tidak bisa langsung mengikuti disebabkan karena kecepatan berbanding terbalik dengan torsi motor dan torsi motor saat keadaan awal akan lebih besar.

Berikut ini adalah tabel perbandingan hasil pengujian menggunakan *simulink,* dimana nilai *input* kecepatan dipilih atau ditentukan secara acak.

Tabel 3. Perbandingan kecepatan *set point* dengan kecepatan motor

No	*Masukan*	*Keluaran*	*Selisih*
1	**250**	**253.4**	**3.4**
2	**500**	**500.04**	**0.04**
3	**1750**	**1740.3**	**10.3**
4	**1000**	**996.9**	**3.1**
5	**1400**	**1393.4**	**6.6**
6	**800**	**798.6**	**1.4**

Diperoleh rata-rata selisih kecepatan sebesar 4.14 rad/s. Hasil penelitian ini membuktikan bahwa simulasi JST sebagai kendali kecepatan motor DC sudah cukup baik, dengan persentase MSE kecepatan pengujian sebesar 1.645% atau tingkat keberhasilan mencapai 98.355.

$$selisih\ \ rata - rata = \frac{\sum selisih}{banyak\ \ \ data} \quad (4)$$

$$MSE\ selisih = \frac{\sum (selisih)^2}{banyak\ data} \times \frac{100}{nilai\ data\ \max} \% \quad (5)$$

Persentase ini belum mencapai nilai minimum karena struktur yang digunakan saat pengujian adalah hasil pelatihan yang berhenti disebabkan Karena MSE minimum pelatihan belum mencapai target MSE. Hal ini juga disebabkan keterbatasan komputer yang digunakan saat pelatihan dan simulasi, sehingga dengan data masukan yang banyak, perlu dicoba menggunakan komputer dengan tingkat proses komputasi yang lebih tinggi atau dengan menggunakan metode pembelajaran yang lebih cepat agar dapat mencapai MSE target yang sekecil mungkin.

5. Kesimpulan

a. Pelatihan dan pengujian akan lebih cepat jika data yang dihitung dalam jaringan menggunakan operasi matriks.
b. Banyaknya data yang dijadikan sebagai data pelatihan, berpengaruh terhadap lamanya waktu iterasi untuk mencapai taget *error* minimum, jumlah iterasi akan semakin banyak dan nilai MSE yang dihasilkan.
c. Struktur terbaik JST untuk sistem kendali kecepatan motor DC terdiri dari 5 sel neuron lapisan *input*. Lapisan tersembunyi terdiri dari 2 lapisan, dimana HL pertama memiliki 3 sel neuron, lapisan tersembunyi kedua terdiri dari 1 sel neuron. (*Mean Square Error*) MSE yang dihasilkan adalah 0.007382 dengan fungsi aktivasi setiap lapisan menggunakan fungsi *purelin* (fungsi identitas).
d. Persentase MSE untuk selisih kecepatan pengujian adalah sebesar 1.645 %.
e. Membuktikan bahwa dasar teori tentang pengaturan kecepatan motor DC metode *Ward Leonard* tentang penggunaan 2 motor dapat lebih efisien dengan *Artificial Intellegence* menggunakan *Neural Network.*

6. Saran

a. Jaringan dilakukan dengan metode yang berbeda, agar menghasilkan nilai *Mean Square Error* yang lebih kecil lagi.
b. Dilakukan perbandingan simulasi sistem jaringan menggunakan *S-Function* dan dengan blok-blok terpisah.
c. Mengganti dengan model motor yang lain, tetapi dengan struktur jaringan syaraf yang sama untuk membuktikan apakah jaringan mampu beradaptasi dengan data motor yang berbeda.
d. Dilakukan aplikasi terhadap motor DC *real* menggunakan *xPC target* dengan *interface serial port* RS 232.
e. Diadakan penelitian lanjutan yang lebih mendetail dengan proses aplikasi sesuai metode *Ward Leonard* untuk mempertegas hasil dari penelitian ini.

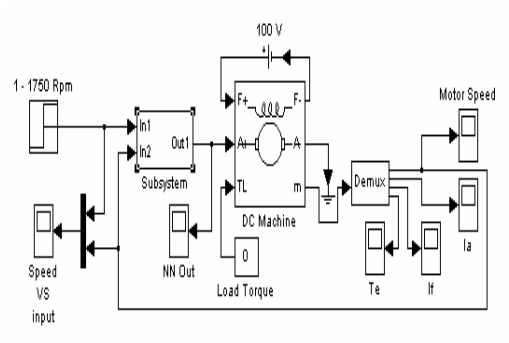
Daftar Pustaka

[1] Brooks/Cole, *Simulations of Machines.*
[2] Fausett, Laurance, *Fundamental Of Neural Network*, 1994, Prentice Hall International Edition.
[3] Floyd, Thomas L., *Electronics Fundamentals*, 1995, Prentice Hall International Edition
[4] Harvey, Robert L., *Neural Network Principles*, 1993, Prentice Hall International Edition.
[5] Kung, S.Y., *Digital Neural Networks*, 1993, Prentice Hall International Edition.
[6] Kuo, Benjamin C., *Automatic Control System,* edisi bahasa indonesia jilid 1, 1995, Prentice Hall International Edition.
[7] Kusumadewi, Sri, *Artificial Intelligence*, 2003, Graha Ilmu.
[8] Lin, Chin-Theng & Lee, C.S George, *Neural Fuzzy Sistems*, 1996, Prentice Hall International Edition.
[9] Ogata, Katsuhiko, *Teknik Kontrol Automatik*, jilid 1, 1997, Erlangga.
[10] Suwandi, Adang, dkk, *Sistem Penggerak Cerdas Berbasis Jaringan Syaraf Tiruan,* Tabloid Elektron, No. 49, Th XVIII.

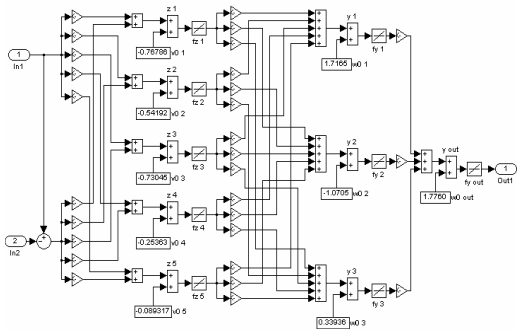
ISBN: 979-756-061-6

[11] Theraja, B.L., *Electronics & Telecomunication Engineering (objective type)*, 1982, S. Chand & Company. Ltd.

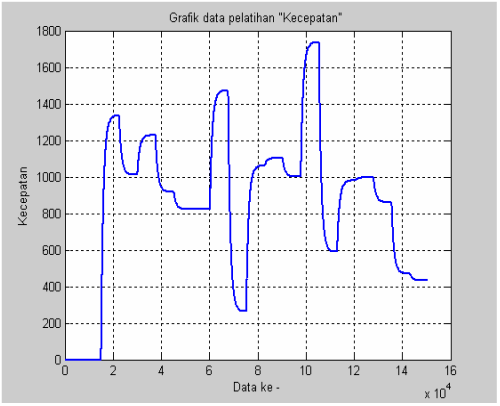
Lampiran

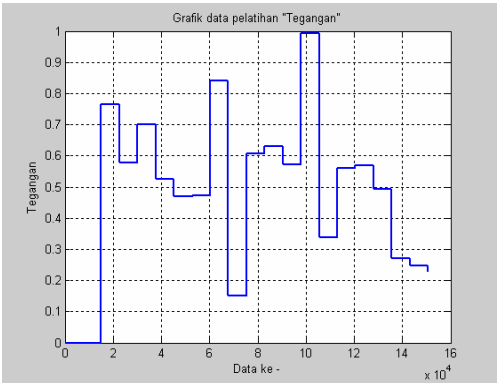


Rangkaian utama simulasi



Rangkaian JST simulasi





Grafik data *input* (*speed*) dan *output* (*voltage*) pelatihan