Perancangan Dan Implementasi Sistem Pengaturan Kecepat*an* Motor Bldc Menggunakan Kontroler Pi Berbasiskan *Neural-Fuzzy* Hibrida Adaptif

Agung Setyadi Wicaksono, Rushdianto Effendie A. K., dan Eka Iskandar
Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember (ITS)
Jl. Arief Rahman Hakim, Surabaya 60111
E-mail: agungsetyadiw@gmail.com, rusdhi@elect-eng.its.ac.id, iskandar@elect-eng.its.ac.id

***Abstrak*—Perkembangan di bidang otomotif mengantarkan kita pada kendaraan tanpa menggunakan bahan bakar minyak. Mobil listrik menjadi inovasi terbaru dengan tujuan utama untuk melepaskan ketergantungan pada bahan bakar minyak. Penelitian yang telah ada memaparkan bahwa motor listrik yang sesuai untuk menggerakkan mobil listrik adalah motor *Brushless Direct Current* (BLDC). Beberapa keunggulan motor BLDC antara lain adalah suara halus, ukuran kompak, torsi besar, efisiensi tinggi, memiliki umur pakai yang panjang, dan mudah dikontrol. Performa dan kecepatan motor BLDC dapat terganggu apabila bekerja pada kondisi berbeban. Oleh karena itu, dibutuhkan pengaturan kecepatan menggunakan sebuah kontroler yang dapat menjaga kecepatan motor BLDC sesuai *set-point* meskipun sedang beroperasi pada kondisi berbeban.Kontroler yang digunakan untuk mengatur kecepatan motor BLDC adalah kontroler Proposional Integral (PI) berbasiskan *Neural-Fuzzy* Hibrida Adaptif. Kontroler PI dipilih karena dapat mengeliminasi *steady-state error*. Sedangkan *Neural-Fuzzy* Hibrida Adaptif merupakan kombinasi antara *Fuzzy* dan *Neural-Network*. *Fuzzy* digunakan untuk penentuan parameter kontroler PI. Parameter kontroler PI didapatkan dari *Neural-Network*. Karakteristik respon terhadap hasil implementasi memiliki *settling time* 20 detik, *overshoot* sebesar 1,1%, dan *time constant* 7,7 detik.**

Kata Kunci— Brushless DC, Hybrid Adaptive Neural-Fuzzy, Propotional and Integral (PI) Controller.

I. PENDAHULUAN

Perkembangan di bidang otomotif mengantarkan kita pada kendaraan tanpa menggunakan bahan bakar minyak. Mobil listrik menjadi inovasi terbaru dengan tujuan utama untuk melepaskan ketergantungan pada bahan bakar minyak [1].

Penelitian yang telah ada memaparkan bahwa motor listrik yang sesuai untuk menggerakkan mobil listrik adalah motor *Brushless Direct Current* (BLDC). Motor BLDC dipilih karena memiliki beberapa keunggulan apabila dibandingkan dengan motor listrik yang lain. Beberapa keunggulan yang menjadi pertimbangan pada pemilihan motor BLDC antara lain adalah suara halus, ukuran kompak, torsi besar, efisiensi tinggi, memiliki umur pakai yang panjang, dan mudah dikontrol. Motor BLDC beroperasi tanpa menggunakan sikat sehingga rugi gesekan pada sikat dapat dihilangkan [1].

Performa dan kecepatan motor BLDC dapat terganggu apabila bekerja pada kondisi berbeban. Oleh karena itu, dibutuhkan pengaturan kecepatan menggunakan sebuah kontroler yang dapat menjaga kecepatan motor BLDC sesuai *set-point* meskipun sedang beroperasi pada kondisi berbeban [2].

Kontroler yang digunakan untuk mengatur kecepatan motor BLDC adalah kontroler Proposional Integral (PI) berbasiskan *Neural-Fuzzy* Hibrida Adaptif. Kontroler PI dipilih karena dapat mengeliminasi *steady-state error*. Sedangkan *Neural-Fuzzy* Hibrida Adaptif merupakan kombinasi antara *Fuzzy Logic* dan *Neural Network*. Sedangkan *Neural-Fuzzy* Hibrida Adaptif merupakan kombinasi antara *Fuzzy* dan *Neural Network*. *Fuzzy* digunakan untuk penentuan parameter kontroler PI. Parameter kontroler PI didapatkan dari *Neural-Network*.

II. TEORI PENUNJANG

A. BLDC [2]

Motor BLDC merupakan pilihan tepat untuk aplikasi yang membutuhkan keandalan tinggi, efisiensi tinggi, dan rasio *power-to-volume* tinggi. Secara umum, motor BLDC dianggap sebagai motor dengan performa tinggi yang mampu menghasilkan torsi yang besar pada *range* kecepatan yang besar. Motor BLDC adalah turunan dari motor DC yang paling umum digunakan, yaitu motor DC dengan sikat dan mereka memiliki kurva karakteristik torsi dan kecepatan yang sama. Perbedaan utama motor BLDC dan DC adalah penggunaan sikat. Motor BLDC tidak memiliki sikat dan harus terkomutasi secara elektronik.

Komutasi merupakan perubahan fase arus motor pada waktu yang tepat untuk menghasilkan torsi rotasional. Dalam motor DC dengan sikat, motor memiliki komutator fisik yang digerakkan dengan sikat untuk memindahkan rotor. Dalam motor BLDC, kekuatan arus listrik magnet permanen menyebabkan motor untuk bergerak, sehingga komutator fisik tidak diperlukan.

Motor BLDC sangat handal karena tidak memiliki sikat yang dapat aus dan harus diganti. Ketika dioperasikan dalam kondisi optimal, usia motor dapat lebih dari 10000 jam. Untuk aplikasi jangka panjang, hal ini dapat menjadi keuntungan yang besar. Setiap kali motor rusak atau perlu diganti, *plant* atau bagian dari *plant* harus dimatikan. Hal ini membutuhkan waktu dan uang, tergantung pada berapa lama waktu yang dibutuhkan untuk mengganti komponen yang aus dan rusak agar *plant* dapat berjalan seperti semula. Meskipun motor BLDC memakan biaya lebih dari motor DC dengan

sikat, hal ini akan sepadan seiring dengan banyaknya waktu dan uang yang dihabiskan motor DC dengan sikat.

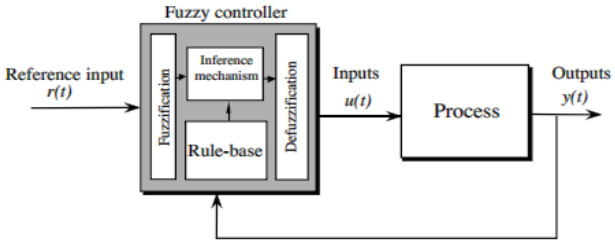
Motor yang digunakan adalah motor *Air Conditioner* (AC) Daikin *Inverter*. Motor Daikin *Inverter* ditunjukkan pada Gambar 1.

Gambar 1 Motor Daikin *Inverter*

B. Fuzzy Logic [3]

Dasar dari *fuzzy logic* adalah himpunan *fuzzy*. Zadeh memperkenalkan konsep himpunan *fuzzy* sebagai perluasan dari himpunan konvensional. Suatu himpunan *fuzzy* adalah koleksi dari bilangan real yang memiliki keanggotaan parsial dalam himpunan.

Logika *fuzzy* sudah digunakan dalam dalam bidang kontrol. Secara umum, sistem *fuzzy* terdiri dari beberapa komponen, yaitu *fuzzification*, *fuzzy rule base*, *fuzzy inference engine*, dan *defuzzifier* seperti yang ditunjukkan pada Gambar 2.



Gambar 2 Fuzzy Controller

Fuzzy rule base berisi pernyataan-pernyataan logika *fuzzy*, yang berbentuk pernyataan IF-THEN. Bentuk umum pernyataan *fuzzy* ditunjukkan pada Persamaan 1.

$$IF\ x_1\ is\ A_1^1\ and\ \dots and\ x_n\ is\ A_n^1\ THEN\ y\ is\ B^1 \quad (1)$$

A_1^l dan B^l adalah himpunan *fuzzy*, sedangkan $x = (x_1, x_2, \dots, x_n)^T$ dan y adalah *input* dan *output* dari variabel *fuzzy*.*Fuzzy inference engine* merupakan suatu modul pembuat keputusan dalam kontroler *fuzzy* yang berfungsi untuk menentukan suatu kesimpulan berdasarkan pada seberapa besar pengaruh setiap *rule* dalam *rule base* berdasarkan pada nilai *input* yang masuk.

Fuzzifier digunakan untuk memetakan nilai variabel di dunia nyata kedalam himpunan *fuzzy*. Pemetaan nilai variabel dilakukan dengan menggunakan fungsi keanggotaan. Formulasi *singleton*, *gaussian*, dan *triangular fuzzyfier* ditunjukkan pada Persamaan 2, 3, dan 4.

$$\mu A'(x) = \begin{cases} 1\ if\ x = x* \\ 0\ otherwise \end{cases} \quad (2)$$

$$\mu A'(x) = e^{-\left(\frac{x_1 - x_1^*}{a_1}\right)} * \dots . * e^{-\left(\frac{x_n - x_n^*}{a_n}\right)} \quad (3)$$

$$\mu A'(x) = \begin{cases} \left(1 - e^{-\left(\frac{x_1 - x_1^*}{b_1}\right)} * \dots . * e^{-\left(\frac{x_n - x_n^*}{b_n}\right)} \text{ if } |x_n - x_n^*| \le b_i, i = 1,2, \dots, n\right) \\ 0 \text{ otherwise} \end{cases} \quad (4)$$

Defuzzifier mengembalikan hasil perhitungan *fuzzy* menjadi variabel yang sesuai di dunia nyata. Sama dengan fuzzifier, defuzzifier juga menggunakan fungsi keanggotaan untuk memetakan nilai himpunan *fuzzy* menjadi variabel nyata. Beberapa metode *defuzzifier* adalah:

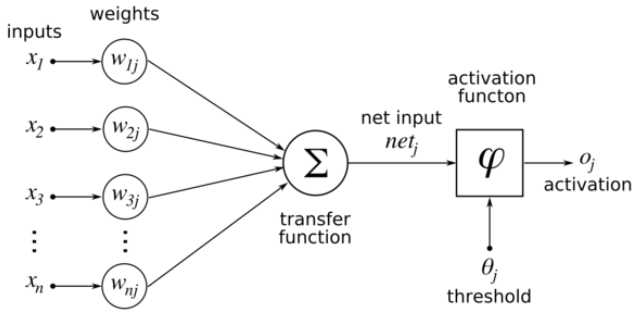
1) *Center of gravity defuzzifier*. *Center of gravity* dinyatakan dengan y*, menunjukan pusat area yang diliputi oleh fungsi keanggotaan.
2) *Center average defuzzifier*. *Center average* menunjukan *weight average* dari titik tengah (*center*) masing-masing fungsi keanggotaan.
3) *Maximum defuzzifier*. *Maximum defuzzifier* memilih nilai tertinggi sebagai y*. Terdapat 3 pilihan, yaitu *smallest of maxima*, *largest of maxima* atau *mean of maxima*.

C. Jaringan Syaraf Tiruan (JST) [4]

JST adalah suatu program komputer yang dibuat berdasarkan cara kerja otak manusia. JST melakukan proses pelatihan yang meniru kerja otak manusia. Proses komputasi pada JST diilhami dari struktur dan cara kerja otak manusia. Pada jaringan syaraf otak manusia, informasi disalurkan dari satu *neuron* ke *neuron* lainnya. Sementara pada JST proses penyaluran informasi dari satu *neuron* ke *neuron* lainnya diimplementasikan pada program komputer. Proses pelatihan JST umumnya menggunakan metode pelatihan *backpropagation* yang sudah banyak diterapkan pada semua proses pelatihan yang sederhana sampai yang rumit.

Metode pelatihan *backpropagation* termasuk ke dalam metode pelatihan terawasi (*supervisory learning*). *Supervisory learning* adalah metode pelatihan yang memasukan target ke *output* dalam data untuk proses pelatihan. Metode pelatihan *backpropagation* telah banyak diaplikasikan dalam berbagai bidang, antara lain: bidang finansial, pengenalan pola tulis tangan, sistem kontrol, dan lain sebagainya. Metode *backpropagation* banyak diaplikasikan dalam berbagai proses karena metode *backpropagation* didasarkan pada interkoneksi yang sederhana. Apabila *output* JST tidak sesuai dengan *output* yang diinginkan, maka metode *backpropagation* akan memperbaiki bobot (*weight*) yang ada pada lapisan tersembunyi (*hidden layer*) untuk mencapai *output* JST yang sesuai dengan target *output*.

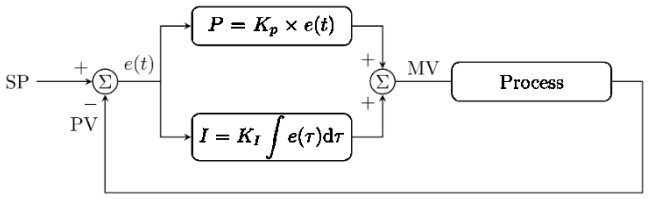
Pada dasarnya JST akan diberikan pola *input* sebagai pola pelatihan maka pola akan menuju ke unit-unit lapisan tersembunyi dan akan diteruskan ke lapisan *output*. Struktur JST ditunjukkan pada Gambar 3.



Gambar 3 Struktur JST

D. *Kontroler PI [2*]

Kontroler PI berguna menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya umpan balik pada sistem tesebut. Kontroler PI adalah kontroler konvensional yang banyak dipakai dalam dunia industri. Blok diagram dan persamaan kontroler PI ditunjukkan pada Gambar 4 dan Persamaan 5.



Gambar 4 Blok Diagram Kontroler PI

$$mv(t) = \left(K_P e(t) + \frac{1}{T_i} \int_0^t e(t)dt \right) \quad (5)$$

Komponen kontroler PI terdiri dari dua jenis yaitu proporsional dan integratif. Komponen proposional dan integral dapat dipakai bersamaan maupun sendiri-sendiri tergantung dari respon yang kita inginkan terhadap suatu *plant*.

Apabila pada kontroler proporsional nilai G(s) = Kp maka Kp berlaku sebagai *gain* (penguat) saja tanpa memberikan efek dinamik kepada kinerja kontroler. Penggunaan kontroler proporsional memiliki berbagai keterbatasan karena sifat kontrol yang tidak dinamik. Walaupun demikian dalam aplikasi-aplikasi dasar yang sederhana kontroler proposional mampu memperbaiki respon transien khususnya *rise time* dan *settling time*. Kontroler proporsional memiliki keluaran yang sebanding dengan besarnya sinyal kesalahan (selisih antara besaran yang diinginkan dengan *output*).

Ciri-ciri kontroler proporsional:

1) Apabila nilai Kp kecil, kontroler proporsional hanya mampu melakukan koreksi kesalahan yang kecil, sehingga akan menghasilkan respon sistem yang lambat (menambah *rise time*)
2) Apabila nilai Kp dinaikkan, respon sistem akan semakin cepat mencapai keadaan *steady state* (mengurangi *rise time*)
3) Namun, apabila nilai Kp diperbesar sehingga mencapai harga yang berlebihan, akan mengakibatkan sistem bekerja tidak stabil atau respon sistem akan berosilasi
4) Apabila nilai Kp dapat di-*set* sedemikian sehingga mengurangi *error steady state*, tetapi tidak menghilangkannya.

Kontroler integral berfungsi untuk menghasilkan respon sistem yang memiliki *error steady state* nol. Apabila sebuah kontroler tidak memiliki unsur integrator, kontroler proporsional tidak mampu menjamin memiliki keluaran sistem dengan *error steady state* nol.

Ketika nilai *error steady state* mendekati nol maka efek kontrol integral semakin kecil. Kontroler integral dapat menghilangkan *error steady-state*, namun pemilihan Ki yang tidak tepat dapat menyebabkan ketidakstabilan sistem. Pemilihan Ki yang sangat tinggi dapat menyebabkan *output* berosilasi karena menambah orde sistem.

Keluaran kontroler integral merupakan hasil penjumlahan yang terus menerus dari perubahan *input*. Ketika sinyal kesalahan tidak mengalami perubahan, maka kontroler integral menjaga keadaan *output* seperti sebelum terjadinya perubahan *input*. Sinyal keluaran kontroler integral merupakan luas bidang yang dibentuk oleh kurva *error.*

Ciri-ciri kontroler proporsional:

1) Apabila nilai Kp kecil, kontroler proporsional hanya mampu melakukan koreksi kesalahan yang kecil, sehingga akan menghasilkan respon sistem yang lambat (menambah *rise time*)
2) Apabila nilai Kp dinaikkan, respon sistem akan semakin cepat mencapai keadaan *steady state* (mengurangi *rise time*)
3) Namun, apabila nilai Kp diperbesar sehingga mencapai harga yang berlebihan, akan mengakibatkan sistem bekerja tidak stabil atau respon sistem akan berosilasi
4) Apabila nilai Kp dapat di-*set* sedemikian sehingga mengurangi *error steady state*, tetapi tidak menghilangkannya.

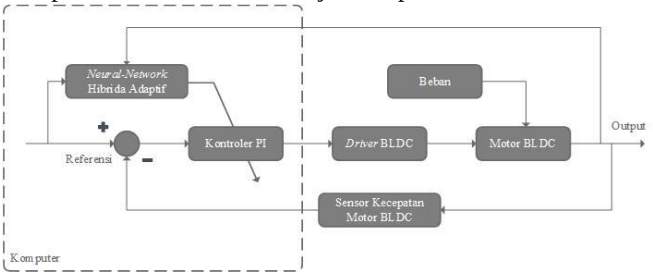
III. PEMODELAN DAN PERANCANGAN

A. *BLDC*

Penulis merancang sistem pengaturan kecepatan motor BLDC menggunakan kontroler PI berbasiskan *Neural-Fuzzy* Hibrida Adaptif. Hibrida merupakan kombinasi antara *fuzzy* dan *neural-network* sedangkan adaptif merupakan sifat sistem kendali dimana parameter-parameter sistem kendali dapat menyesuaikan terhadap perubahan kondisi beban. Kontroler *fuzzy* pada sistem pengaturan kecepatan motor BLDC berfungsi sebagai pengatur nilai Kp dan Ki pada kontroler PI, sehingga nilai Kp dan Ki menjadi fungsi keanggotaan pada *output* kontroler *fuzzy*. *Input* kontroler *fuzzy* berupa nilai tegangan beban. Nilai Kp dan Ki didapatkan dari proses *learning neural-network*.

Plant menggunakan sistem kendali *negative feedback* yang terdiri dari komponen utama yaitu motor BLDC, *driver*, dan rem elektromagnetik. *Driver* berfungsi sebagai aktuator yang menghubungkan antara kontroler dengan *plant* sedangkan rem elektromagnetik berfungsi untuk memberikan efek pembebanan pada motor BLDC. Sinyal kontrol dari kontroler berupa sinyal PWM yang digunakan sebagai *input driver* untuk mengatur kecepatan motor BLDC. Pemorograman kontroler dilakukan di MATLAB R2013a sedangkan Arduino bertindak sebagai *interface* antara komputer dengan *driver* motor BLDC. Sensor yang digunakan dalam sistem pengaturan kecepatan motor BLDC adalah sensor kecepatan

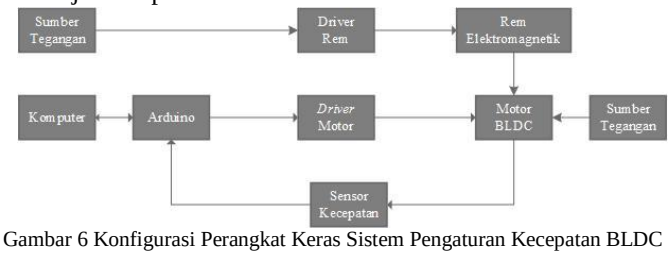
yang digunakan sebagai *negative feedback* untuk mengukur kecepatan motor secara aktual. Blok diagram pengaturan kecepatan motor BLDC ditunjukkan pada Gambar 5.



Gambar 5 Blok Diagram Pengaturan Kecepatan Motor BLDC

Perancangan perangkat keras ada dua jenis, yaitu perancangan mekanik dan elektronik. Perancangan mekanik merupakan perancangan komponen utama pada *plant*, yaitu berupa motor BLDC dan rem elektromagnetik sedangkan perancangan elektronik merupakan perancangan pada kontroler, *driver* rem, dan rangkaian sensor kecepatan pada *plant*.

Arduino menerima data dari *sensor* kecepatan dan mengirimkan data pada komputer kemudian mengirimkan sinyal kontrol ke motor BLDC melaui Arduino Konfigurasi perangkat keras sistem pengaturan kecepatan BLDC ditunjukkan pada Gambar 6

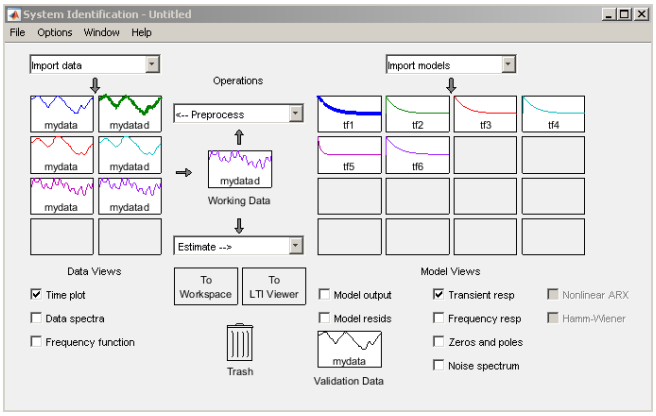


Gambar 6 Konfigurasi Perangkat Keras Sistem Pengaturan Kecepatan BLDC

B. *Identifikasi Sistem*

Identifikasi pemodelan sistem pengaturan kecepatan motor BLDC digunakan identifikasi dinamis. Model matematika *plant* didapatkan melalui *toolbox matlab system identification* dengan cara meng-*input*-kan data *input-output*, melakukan *preprocessing*, dan memilih bentuk model yang diinginkan. Dalam proses identifikasi, data *input-output* yang ada dimasukkan ke dalam aplikasi dengan memilih "*time domain data"* saat menekan akan timbul tulisan "*import data"*. Setelah itu dilakukan *preprocessing*.

Data yang telah melalui tahap *preprocess* kemudian dipilih sebagai *working data* sekaligus *validation data* dengan cara *drag-and-drop* kotak yang berisi data tersebut ke kotak *working data* dan *validation data*. Kemudian dilakukan identifikasi model dengan memilih "*transfer function*" *se*telah meng-klik *popout* bertuliskan "*estimate*". Tampilan dari *system identification toolbox* pada MATLAB ditunjukkan pada Gambar 7.



Gambar 7 Tampilan System Identification Toobox MATLAB

Fungsi alih yang didapatkan setelah dilakukan identifikasi menggunakan *system identification toolbox* ditunjukkan pada Tabel 1.

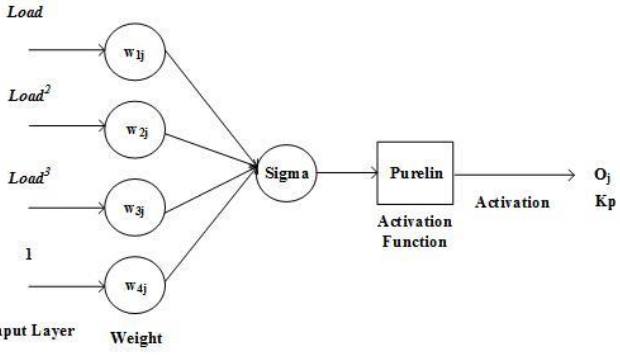
Tabel 1.
Fungsi Alih *Plant* Motor *BLDC*.

Beban	Tegangan *input* beban	Fungsi Alih	MSE
Minimal	16 Volt	$\frac{428{,}8s + 1279}{s^2 + 2{,}351s + 0{,}7318}$	376,6
Nominal	20 Volt	$\frac{536{,}2s + 1969}{s^2 + 3{,}461s + 1{,}185}$	346,9
Maksimal	24 Volt	$\frac{460{,}3s + 1396}{s^2 + 2{,}47s + 0{,}8689}$	271

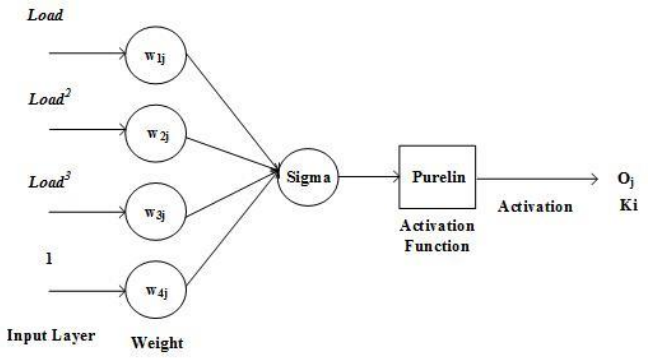
C. *Perancangan Neural-Fuzzy Hibrida Adaptif*

Setelah fungsi alih dari *plant* didapatkan kemudian melakukan perancangan kontroler untuk masing-masing pembebanan. Kontroler digunakan untuk mengembalikan respon ke nilai *set point* meskipun motor BLDC diberi beban. Tahapan desain kontroler meliputi perancangan kontroler PI yang didapatkan dari *learning Neural-Network* dan perancangan kontroler *fuzzy* PI.

Neural-Network digunakan sebagai *tuning* parameter PI untuk mengatur kecepatan motor BLDC.



Gambar 8 Struktur Neural-Network untuk Kp



Gambar 9 Struktur Neural-Network untuk Ki

Algoritma pembelajaran *neural-network* yang dipakai adalah *feed forward*. *Input layer neural-network* berupa nilai tegangan beban. Persamaan *input layer neural-network* ditunjukkan pada Persamaan 6.

$$O_j^1 = x(j)(j = 1,2,\dots,M) \tag{6}$$

M meruapakan jumlah variabel *input* pada *input layer*, pada *neural network* digunakan 4 variabel *input*. Variabel *input* dari *input layer* kemudian menuju *output layer*.

$$net^1 = \sum_{j=0}^{M} w_j^1 O_j^1 \tag{7}$$

$$O^2 = purelin(net^1) \tag{8}$$

Untuk merivisi bobot *neural-network* dipakai algoritma *gradient steepest descent* sehingga dipenuhi kriteria *error* kuadarat tiap saat minimum. Formulasi kriteria *error* kuadrat tiap saat minimum ditunjukkan pada Persamaan 9.

$$J = \frac{1}{2}e^2 \tag{9}$$

Error berupa *error* Kp dan *error* Ki. Untuk mendapatkan *error* Kp dan *error* Ki, nilai *error* sinyal kontrol harus diketahui terlebih dahulu. *Error* sinyal kontrol ditunjukkan Persamaan 10.

$$e(u) = (em(k) - e(k)) * \frac{B}{K} \tag{10}$$

Error model, $em(k)$, merupakan nilai *error* yang harus dituju oleh nilai *error* sesungguhnya. $e(u)$ merupakan *error* sinyal kontrol *plant* pengaturan kecepatan motor BLDC. Kemudian, didapatakan nilai *error* Kp dan Ki yang ditunjukkan pada Persamaan 11 dan 12.

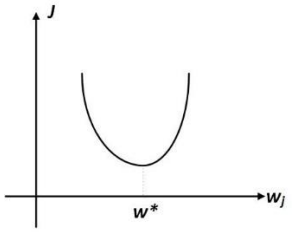
$$ekp = \frac{e.Kp}{u}e(u) \tag{11}$$

$$eki = \frac{e.Ki}{u}e(u) \tag{12}$$

Kriteria *error* kuadrat akan minimum apabila memenuhi formulasi yang ditunjukkan pada Persamaan 13.

$$\frac{\partial J}{\partial w_j} = 0 \tag{13}$$

Karena kriteria *error* kuadarat tiap minimum, *J*, merupakan fungsi kuadrat dari w_j maka dapat digambarkan pada Gambar 10.



Gambar 10 Fungsi J terhadap w_j

Kemudian didapatkan persamaan $\frac{\partial J}{\partial w_j}$ yang ditunjukkan pada Persamaan 14.

$$\frac{\partial J}{\partial w_j} = \frac{\partial J}{\partial e}\frac{\partial e}{\partial O_j^2}\frac{\partial O_j^2}{\partial net^1}\frac{\partial net^1}{\partial w_j}$$
$$\frac{\partial J}{\partial w_j} = -ef'(O_j^2)x(j) \tag{14}$$

Kemudian didapatkan persamaan *gradient steepest descent* yang ditunjukkan pada Persamaan 15.

$$w_j(k+1) = w_j(k) + \Delta w * e * f'(O_j^2)x(j) \tag{15}$$

Tahap perancangan kontroler *fuzzy* PI memiliki berberapa tahap yaitu penentuan fungsi keanggotaan dan bentuk fungsi keanggotaan kemudian membuat *rule base* dan menentukan fungsi keanggotaan pada *output*.

Kontroler *fuzzy* pada sistem pengaturan kecepatan motor BLDC berfungsi sebagai pengatur nilai Kp dan Ki pada kontroler PI, sehingga nilai Kp dan Ki menjadi fungsi keanggotaan pada *output* kontroler *fuzzy*. Pada *input fuzzy* memiliki 1 *input* yaitu besar beban pengereman yang diberikan.

Kontroler *fuzzy* PI digunakan untuk mengeluarkan nilai Kp dan Ki sesuai dengan kondisi pembebanan maka dalam *rule base* memiliki 3 pernyataan di setiap nilai Kp dan Ki. Misalkan dipilih representasi fungsi keanggotaan *input* sebagai berikut:

Beban Minimal = BK
Beban Nominal = BS
Beban Maksimal = BB

Maka *rule base* untuk kontroler *fuzzy* akan terlihat seperti berikut.

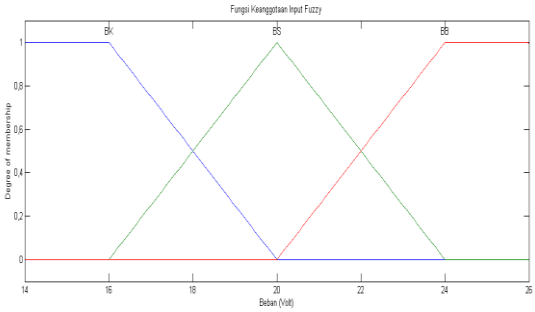
If X = BK *Then* K_P = 2,497369583
If X = BS *Then* K_P = 7,911902938
If X = BB *Then* K_p = 14,50763256

Pernyataan diatas merupakan *rule base* untuk pencarian nilai K_p. Apabila semakin besar pembebanan maka semakin besar nilai K_p yang digunakan. Kemudian menentukan *rule base* untuk Ki.

If X = BK *Then* K_i = 1,892985981
If X = BS *Then* K_i = 2,20533539
If X = BB *Then* K_i = 7,721486311

Pernyataan diatas merupakan *rule base* untuk pencarian nilai Ki. Apabila semakin besar pembebanan maka semakin besar nilai Ki yang digunakan.

Fungsi keanggotaan untuk *input* pada kontroler *fuzzy* PI digunakan untuk mengubah nilai Kp dan Ki pada setiap pembebanan, maka *input* fungsi keanggotaan berupa nilai tegangan pembebanan. Karena sistem pengaturan kecepatan motor BLDC memiliki pembebanan minimal, nominal, dan maksimal maka pada fungsi keanggotaan memiliki 3 anggota. Berikut ini fungsi keanggotaan dari *input kontroler fuzzy*.



Gambar 11 Fungsi Keanggotaan Input

Pada Gambar 11 dapat dilihat bahwa jenis fungsi keanggotaan yang dipakai adalah jenis *triangular*. Fungsi keanggotaan *input* kontroler *fuzzy* terdiri dari 3 *fuzzy set* yaitu BK, BS, dan BB.

Pada fungsi keanggotaan *output* terdapat 3 anggota pada setiap Kp dan Ki. Bentuk dari fungsi keanggotaan hanya berupa konstanta atau *singleton* dari nilai Kp dan Ki. Proses defuzzifikasi menggunakan metode *center average* yaitu nilai Kp dan Ki yang telah didapatkan dikalikan dengan nilai bobot pada tiap *rule*-nya kemudian hasil kali antara *weight* dengan nilai Kp dan Ki dijumlahkan dan dibagi dengan jumlah total bobot.

D. Perancangan Kontroler PI

Parameter kontroler PI didapatkan dengan *tuning neuro fuzzy*. Blok diagram dan persamaan kontroler PI ditunjukkan pada Persamaan 16.

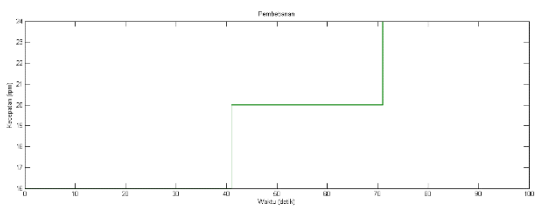
$$mv(t) = \left(K_P e(t) + \frac{1}{T_i} \int_0^t e(t)dt \right) \qquad (16)$$

Komponen kontroler PI terdiri dari dua jenis yaitu proporsional dan integratif. Komponen proposional dan integral dapat dipakai bersamaan maupun sendiri-sendiri tergantung dari respon yang kita inginkan terhadap suatu *plant*.

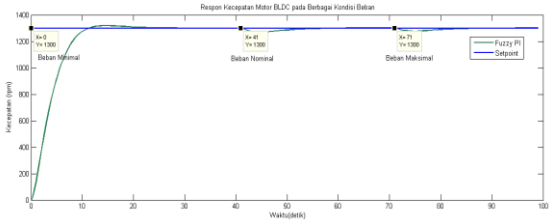
IV. PENGUJIAN DAN ANALISIS

A. Simulasi Pengujian Kontroler PI berbasiskan Neural-Fuzzy Hibrida Adaptif

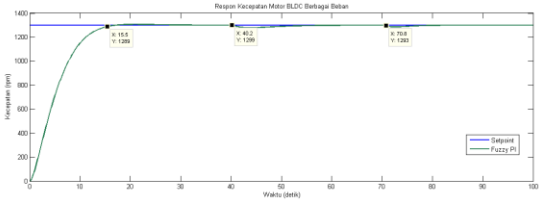
Pembebanan dan hasil respon simulasi dengan kontroler PI berbasiskan *Neural-Fuzzy* Hibrida Adaptif dapat dilihat pada Gambar 12, 13 dan 14. *Setpoint* yang diberikan sebesar 1300 RPM dan beban yang digunakan adalah beban minimal pada 0-40 detik, beban nominal pada 41-70 detik, kemudian beban maksimal pada 71-99 detik. Karakteristik respon hasil simulasi memiliki *settling time* 11 detik, *overshoot* sebesar 1,5%, dan *time constant* sebesar 5 detik.



Gambar 12 Pembebanan pada Motor BLDC



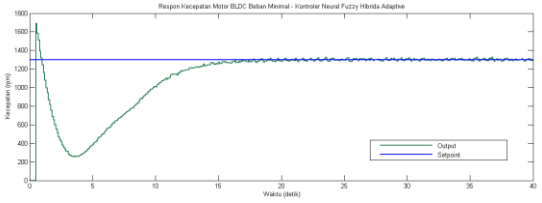
Gambar 13 Respon Kecepatan Motor BLDC pada berbagai Beban



Gambar 14 Respon Kecepatan Motor BLDC pada berbagai Beban 2

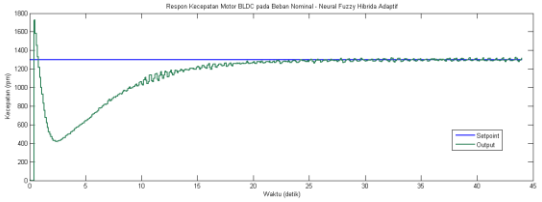
B. Implementasi Kontroler PI berbasiskan Neural-Fuzzy Hibrida Adaptif

Pada implementasi sistem diberikan *setpoint* berupa sinyal *step* yang bernilai 1300 rpm. Pengujian dilakukan pada setiap beban dengan *setpoint* yang sama. Setelah itu respon motor BLDC dianalisis dengan mencari nilai *settling time*, *overshoot*, dan *time constant*. Respon kecepatan motor BLDC ketika beban minimal, nominal, maksimal, dan berbagai beban ditunjukkan pada Gambar 15, 16, 17, dan 18.



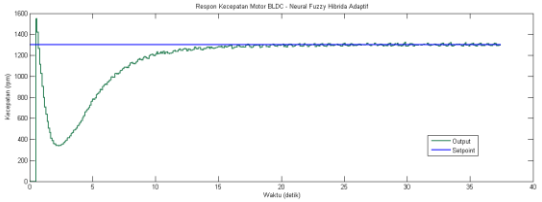
Gambar 15 Implementasi Respon Kecepatan Motor BLDC Beban Minimal

Setpoint kecepatan motor BLDC yang diberikan sebesar 1300 RPM dan beban yang digunakan adalah beban minimal dari 0-40 detik. Karakteristik respon hasil simulasi memiliki *settling time* 15 detik dan memiliki *overshoot* sebesar 1,5%, dan *time constant* 8 detik.



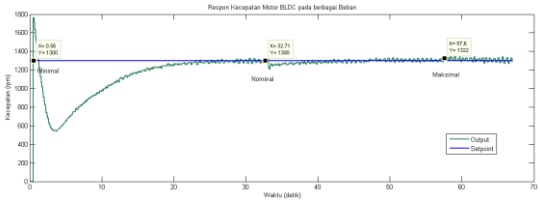
Gambar 16 Implementasi Respon Kecepatan Motor BLDC Beban Nominal

Setpoint kecepatan motor BLDC yang diberikan sebesar 1300 RPM dan beban yang digunakan adalah beban nominal dari 0-40 detik. Karakteristik respon hasil simulasi memiliki *settling time* 17,7 detik dan memiliki *overshoot* sebesar 1,3%, dan *time constant* 9 detik.



Gambar 17 Implementasi Respon Kecepatan Motor BLDC Beban Maksimal

Setpoint kecepatan motor BLDC yang diberikan sebesar 1300 RPM dan beban yang digunakan adalah beban maksimal dari 0-40 detik. Karakteristik respon hasil simulasi memiliki *settling time* 13 detik dan memiliki *overshoot* sebesar 1,1%, dan *time constant* 5 detik.



Gambar 18 Implementasi Respon Kecepatan Motor BLDC pada berbagai BebaSn

Setpoint kecepatan motor BLDC yang diberikan sebesar 1300 RPM pada berbagai beban. Karakteristik respon hasil simulasi memiliki *settling time* 20 detik dan memiliki *overshoot* sebesar 1,1%, dan *time constant* 7,7 detik.

V. KESIMPULAN

Dari analisis yang telah dilakukan terhadap hasil simulasi maka dapat disimpulkan bahwa pengaturan kecepatan menggunakan kontroler PI berbasiskan *Neural Fuzzy* Hibrida Adaptif menghasilkan respon pada berbagai beban dengan karakteristik respon hasil simulasi memiliki *settling time* 11 detik, *overshoot* sebesar 1,5%, dan *time constant* sebesar 5 detik sedangkan terhadap hasil implementasi memiliki karakteristik respon *settling time* 20 detik, *overshoot* sebesar 1,1%, dan *time constant* 7,7 detik.

DAFTAR PUSTAKA

[1] M. Safrurriza, "Desain Sistem Pengaturan Kecepatan Motor Arus Searah Tanpa Sikat Menggunakan PID Mutiobjektif Berdasarkan Algoritma Genetika", *Tugas Akhir*, Jurusan Teknik Elektro, Institut Teknologi Sepuluh Nopember, Surabaya, 2016.

[2] Guntur P., "Perancangan Kontrol Kecepatan Motor Arus Searah Tanpa Sikat Menggunakan Sliding Mode Berbasis PID", *Tugas Akhir*, Jurusan Teknik Elektro, Institut Teknologi Sepuluh Nopember, Surabaya, 2016.

[3] Fairuzza D., "Pengaturan Kecepatan Motor Brushless DC Menggunakan Kontroler Fuzzy Berbasis Linear Quadratic Regulator", *Tugas Akhir*, Jurusan Teknik Elektro, Institut Teknologi Sepuluh Nopember, Surabaya, 2016.

[4] Mansouri, Mahdi, *"*Hybrid Adaptive Neuro Fuzzy Tuned PI*", 4th Power Electronics, Drive Systems & Technologies Conference*, Februari, 2013.