SISTEM KENDALI KECEPATAN MOTOR BLDC MENGGUNAKAN ALGORITMA *HYBRID PID FUZZY*

Roedy Kristiyono[1], Oyas Wahyunggoro[2], Prapto Nugroho[3]
1) Jurusan Teknik Elektro –ATW Surakarta
Email : roedy_kristiyono@ymail.com
2) Jurusan Teknik Eektro dan Teknologi Informasi – UGM
Email : oyas@ugm.ac.id
3) Jurusan Teknik Elektro dan Teknologi Informasi – UGM
Email : pnugroho@jteti.gadjahmada.edu

Abstract

BLDC motors are widely used in many industrial applications because of its high efficiency, high torque and low noise volume. BLDC motor speed control is a complex process. But complexity is made comparable to the performance of the BLDC motor is high. Conventional PID control proved to be able to show good performance in controlling the specific loading on the plant. But any changes in the load of the plant, conventional PID control should be re-set the parameters kp, ki and kd to target steady state according to the desired set point. The purpose of this study is to design equipment for the control of BLDC motor that can automatically tune the PID parameters by fuzzy logic. In the present study used a RISC AVR microcontroller as the control center. While the software is used for algorithm programming PID control and PID-fuzzy hybrid control in C language. In order to tune the proper PID parameters in real time, then made a two-level control system. The first level to determine the parameters of PID by finding the minimum and maximum value of kp, ki and kd the reaction curve method. The second level designing fuzzy systems in order to automatically tune PID reinforcement, then formulated into a combination of 49 fuzzy if-then rules to get the value of kp, ki and kd right of errors and changes in the value of delta error. Testing changes in set point and load changes resulting response characteristics of conventional PID control systems with an average value that is the rise time (tr) 0.025 seconds, the preset time (ts) 0.1625 seconds, amounting to 15.98% overshoot. While control Hybrid PID Fuzzy resulting average value of the rise time (tr) 0.0025 seconds, the preset time (ts) 0.057 seconds, overshoot at 5.42%.

Keywords: *BLDC Motor, Brushless DC Motor, Control, PID, Fuzzy, Auto-tuning*

1. PENDAHULUAN

Sistem kendali PID paling banyak digunakan dalam pengendalian di industri. Keberhasilan pengendali PID tergantung ketepatan dalam menentukan konstanta (penguatan) PID[1]. Dalam suatu sistem kendali kita mengenal adanya beberapa macam aksi kendali, yaitu aksi kendali proporsional, aksi kendali integral, dan aksi kendali derivatif. Masing-masing aksi kendali ini mempunyai keunggulan tertentu, dimana aksi kendali proporsional mempunyai keunggulan *risetime* yang cepat, aksi kendali integral mempunyai keunggulan untuk memperkecil *error*, dan aksi kendali derivatif keunggulannya untuk memperkecil *derror* atau meredam *overshoot/undershoot*. Agar mendapatkan suatu keluaran yang tinggi dan *error* yang kecil, maka kita dapat gabungkan kendali-kendali tersebut menjadi aksi PID. Pada makalah ini sistem kendali yang digunakan adalah sistem kendali PID digital.

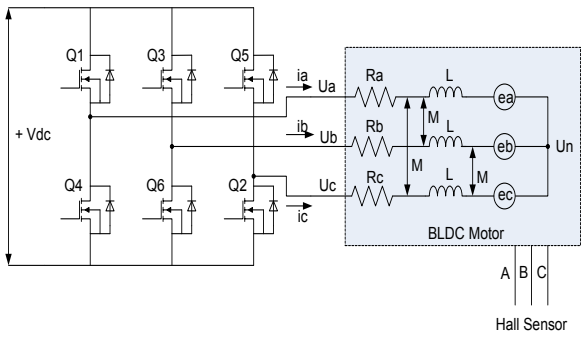
Seiring dengan perkembangan sistem berbasis pengetahuan, penalaan konstanta PID ini dapat ditentukan dengan menganalisis tanggapan suatu sistem, hasil analisis ini dibentuk dalam sejumlah aturan. Dengan mengkombinasikan aturan, pengaturan ini menjadi sebuah sistem *fuzzy* sebagai salah satu sistem berbasis pengetahuan sehingga konstanta PID dapat ditala secara waktu nyata. Untuk mencapai

hal tersebut sistem kendali dibentuk menjadi sistem kendali dua aras[2]. Aras pertama adalah sistem kendali PID konvensional. Aras kedua adalah sistem *fuzzy* yang menala konstanta PID secara waktu nyata.

II. SISTEM KENDALI KECEPATAN MOTOR BLDC MENGGUNAKAN ALGORITMA HYBRID PID-FUZZY

a. *Brushless DC Motor*

Brushless motor DC (Motor BLDC) semakin banyak digunakan dalam aplikasi motor karena mereka memiliki banyak keuntungan. Motor BLDC tidak memiliki sikat sehingga tidak banyak atau bahkan tidak memerlukan perawatan. Tidak menghasilkan suara atau kebisingan dibandingkan dengan motor dc *universal* yang memakai sikat. Juga memiliki berat yang baik untuk rasio ukuran daya. Motor tersebut memiliki *rotor inersia* yang kecil. Kumparan yang melekat pada *stator*. Komutasi ini dikendalikan secara elektronik. Komutasi disediakan baik oleh sensor posisi (*hall sensor*) atau dengan gulungan *Back Electomotive Force.* Dalam mode sensor, motor BLDC biasanya terdiri dari tiga bagian utama: *stator*, *rotor* dan *hall sensor*



Gbr 1. Motor Drive dan ekuivalen BLDC Motor

TABEL 1
Motor BLDC dengan arah putar jarum jam

Hall Sensor			EMF			Gate Logic					
A	B	C	A	B	C	Q1	Q2	Q3	Q4	Q5	Q6
0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	-1	1	0	0	0	1	1	0
0	1	0	-1	1	0	0	1	1	0	0	0
0	1	1	-1	0	1	0	1	0	0	1	0
1	0	0	1	0	-1	1	0	0	0	0	1
1	0	1	1	-1	0	1	0	0	1	0	0
1	1	0	0	1	-1	0	0	1	0	0	1
1	1	1	0	0	0	0	0	0	0	0	0

b. Kendali Proposional Integral Derivatif (PID)

Karaktristik yang umum digunakan dalam pengaturan suatu sistem antara lain meliputi stabilitas, akurasi, kecepatan respon dan sensitivitas[3]. Dalam aksi kendali proposional, *output* dari sistem kendali akan sebanding dengan *input* nya. Sinyal *output* merupakan penguatan dari sinyal kesalahan dengan faktor tertentu, faktor penguatan ini merupakan konstanta proposional dari sistem, yang dinyatakan dengan K_p, dimana K_p menpunyai tanggapan yang tinggi/cepat. Dalam kendali integral, *output* nya selalu berubah selama terjadi penyimpangan, dan kecepatan perubahan *ouput* nya tersebut sebanding dengan penyimpangannya, konstanta dinyatakan dengan K_i, dimana K_i ini mempunyai sensitivitas yang tinggi, yaitu dengan cara mereduksi *error* yang dihasilkan dari sinyal *feedback*. Semakin besar nilai K_i, maka sensitivitasnya akan semakin tinggi, tetapi waktu yang dibutuhkan untuk mencapai kestabilan lebih cepat, demikian pula sebaliknya. Sedangkan kendali derivatif (turunan) bekerja berdasarkan laju perubahan simpangan, sehingga jenis kendalier ini selalu digunakan bersama-sama dengan kendalier proposional dan integral. Konstanta dinyatakan dengan K_d, dimana K_d ini mempengaruhi kestabilan dari sistem, karena aksi kendali ini mampu mereduksi *error*. Dengan penggabungan aksi kendali PID ini maka diharapkan akan mendapat suatu tanggapan yang mempunyai tingkat kestabilan yang tinggi.

PID digital merupakan proses dari suatu program yang dijalankan dengan menggunakan komputer, dimana kita memasukkan suatu nilai *Setting Point* (SP) dan *Preset Value* (PV) yang kemudian data tersebut diproses sehingga *error* yang didapatkan sama dengan 0, atau nilai *Setting Point = Preset Value*[4].

Untuk dapat mengimplementasikan sistem kendali PID[4] pada komputer, PID harus diubah ke dalam persamaan diskrit :

$$V_o = Kp + Ki \int e\,dt + Kd\; d\frac{de}{dt} \qquad (1)$$

Kemudian diturunkan

$$\frac{dVo}{dt} = Kp\frac{de}{dt} + Ki\frac{d}{dt}\left(\int e dt\right) + Kd\frac{d^2e}{dt^2} \qquad (2)$$

$$\frac{dVo}{dt} = Kp\frac{de}{dt} + Kie + Kd\frac{d}{dt}\left(\frac{de}{dt}\right) \qquad (3)$$

Dikalikan dengan Ts, maka:

$$\frac{\Delta Vo}{Ts} = Kp\frac{\Delta e}{Ts} + Kie + Kd\frac{\Delta}{Ts}\left(\frac{\Delta e}{Ts}\right) \qquad (4)$$

$$\Delta Vo = Kp\Delta e + KieTs + Kd\Delta\left(\frac{\Delta e}{Ts}\right) \qquad (5)$$

Harga *ΔVo* merupakan harga perubahan *output* yang didapat dari *output* sekarang dikurangi dengan *output* sebelumnya.

$$\Delta Vo = V_{on} - V_{on-1} \quad (6)$$
$$\Delta e = e_n - e_{n-1} \quad (7)$$

Sehingga persamaan, menjadi :

$$V_o - V_{on-1} = Kp(e_n - e_{n-1}) + Kie_nTs + \frac{Kd}{Ts}(\Delta e_n - \Delta e_{n-1}) \quad (8)$$

Pada kondisi akhir, perubahan Δ pada *error* sebelumnya dapat di distribusikan menjadi,

$$\Delta e_n = e_n - e_{n-1} \quad (9)$$
$$\Delta e_{n-1} = e_{n-1} - e_{n-2} \quad (10)$$

Disubtitusikan kedalam persamaan, maka

$$V_o = V_{on-1} + Kp(e_n - e_{n-1}) + Kie_nTs + \frac{Kd}{Ts}[(e_n - 2e_{n-1} + e_{n-2})] \quad (11)$$

Hasil akhir dai persamaan PID menjadi:

$$V_o = V_{on-1} + Kp(e_n - e_{n-1}) + Kie_nTs + \frac{Kd}{Ts}[(e_n - 2e_{n-1} + e_{n-2})] \quad (12)$$

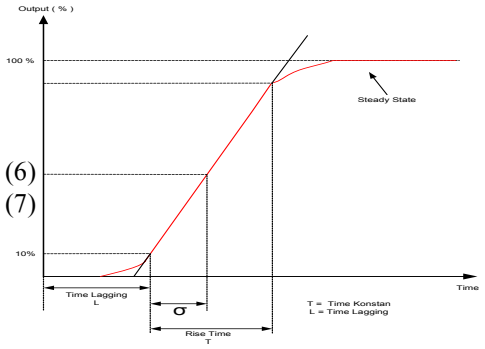
dengan,
V_o = *Output*
V_{on-1} = *Output* sebelumnya
K_p = Konstanta Proposional
K_i = Konstanta Integral
K_d = Konstanta Derivatif
e_n = *error* sekarang
e_{n-1} = *error* 1 kali sebelumnya
e_{n-2} = *error* 2 kali sebelumnya
Ts = *time sampling*

Untuk mendapatkan harga K_p, $T_i = 1/K_i$, T_d (Kd) maka ditentukan dengan kurva proses reaksi (Gbr. 2), dimana sistem dijalankan secara *open loop*[5]
Perhitungan *auto tuning* PID menurut *Ziegler Nichols* dapat dicari dengan persamaan :

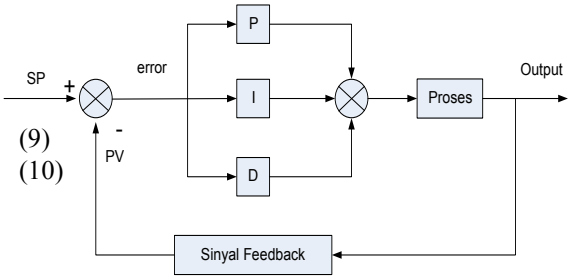
$$K_p = 1{,}2\frac{T}{L} \quad (13)$$

$$K_i = Kp.\frac{1}{Ti} = Kp.2L \quad (14)$$

$$K_d = Kp.Td = Kp.\frac{L}{2} \quad (15)$$



Gbr 2. Kurva Proses Reaksi



Gbr 3. Kendali PID

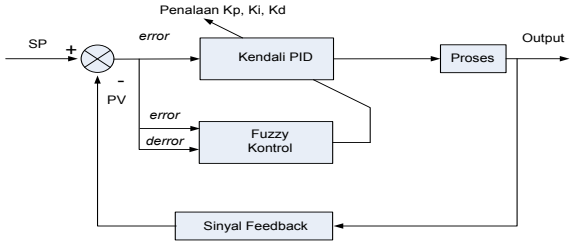
c. Kendali *Hybrid PID-Fuzzy*

Teknologi telah berkembang, dimana tidak lagi memakai cara konvensional untuk mendapatkan suatu hasil yang diinginkan dengan memakai persamaan matematika. Tetapi bisa menerapkan suatu kemampuan manusia untuk mengendalikan sesuatu, yaitu dengan aturan-aturan. *If – then Rules* adalah suatu pengendalian yang akan mengikuti pendekatan secara bahasa, sistem ini disebut dengan sistem kendali logika *fuzzy*. Dimana sistem kendali logika *fuzzy* ini tidak memiliki ketergantungan pada variabel-variabel proses kendali[6]. Sistem ini banyak dikembangkan dalam bidang teknik kendali, terutama untuk sistem nonlinear dan dinamis. Pada aplikasi industri yang membutuhkan suatu sistem kendali dengan kecepatan tinggi dan data *output* yang akurat maka pemakaian aksi kendali PID mungkin masih dianggap kurang memuaskan. Karena jika menggunakan aksi kendali PID, jika kendalier di set sangat sensitif, maka *overshoot/undershoot* yang dihasilkan akan sangat peka, sehingga osilasi yang ditimbukan akan lebih tinggi, sedangkan bila kendalier diset kurang peka maka terjadinya *overshoot/undershoot* dapat diperkecil, tetapi waktu yang dibutuhkan akan semakin lama, dan ini akan menjadi suatu masalah dalam suatu proses industri.

Sistem kendali *hybrid PID-Fuzzy* dalam penelitian ini merupakan sistem *Fuzzy Logic*

untuk menala penguatan kendali PID. Kendali ini kemudian dicoba untuk diimplementasikan pada sistem untuk mengendalikan *plant* yaitu kecepatan motor BLDC dengan pembebanan tertentu.

Sistem utama adalah kendali PID, sedangkan logika *Fuzzy* disini berfungsi untuk memperbaiki tanggapan dan *recovery time* dengan cara menala nilai Kp, Ki dan Kd secara otomatis terhadap pembebanan seperti terlihat pada gbr 4.

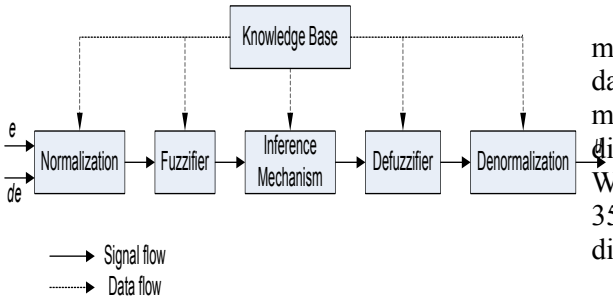


Gbr 4. Kendali *PID tuning Fuzzy*

d. **Desain Fuzzy Logic Controller (FLC)**

Diagram blok FLC dengan dua *input* (e_1, e_2) dan satu *output* (u) ditunjukkan pada Gbr 5. *Error* dihitung dengan mengurangi kecepatan referensi rotor dengan kecepatan motor yang sebenarnya, sebagai berikut:

$$e_1[n] = \omega_{ref}[n] - \omega_r[n] \quad (16)$$



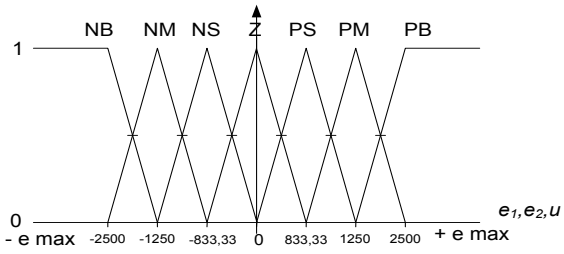
Gbr 5.Blok diagram *Fuzzy Logic Controller*

dimana $e_1[n]$ adalah *error*, $\omega_{ref}[n]$ adalah kecepatan referensi, dan $\omega_r[n]$ adalah kecepatan motor yang sebenarnya. Perubahan dalam kesalahan dihitung dengan persamaan (17), di mana $e_1[n-1]$ adalah nilai *error* sebelumnya.

$$e_2[n] = e_1[n] - e_1[n-1] \quad (17)$$

Dalam sistem kendali logika *fuzzy*, dua parameter normalisasi (N_{e1} , N_{e2}) , untuk *input* dan satu parameter *denormalization* (N_u) didefinisikan untuk *output*. Dalam proses normalisasi, nilai-nilai masukan yang ditingkatkan antara (-1, +1) dan dalam proses *denormalization*, nilai *output* dari kendalier *fuzzy* dikonversi ke nilai tergantung pada terminal elemen kendali.

Nilai-nilai kabur diperoleh dari mekanisme *inferensi fuzzy* harus dikonversi ke nilai output *crips* (u) dengan proses *defuzzifier*. Untuk tujuan ini, fungsi keanggotaan segitiga *fuzzy* didefinisikan untuk setiap masukan dan nilai output dengan tujuh *cluster*. Gbr 6, menggambarkan fungsi keanggotaan yang digunakan untuk *fuzzify* dua nilai masukan (e_1, e_2) dan *output defuziffy* (u) dari kendalier *fuzzy*. Selama tujuh kelompok di fungsi keanggotaan, tujuh variabel linguistik didefinisikan sebagai: *Negatif Big* (NB), *Negatif Medium* (NM), *Negative Small* (NS), *Zero* (Z), *Positif Sm*all(PS), *Positif Medium* (PM), dan *Positif Big* (PB).

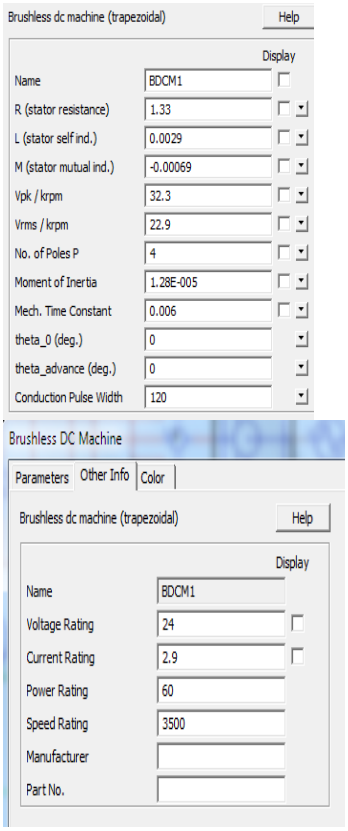


Gbr 6. Fungsi Keanggotaan untuk *e* dan *de*

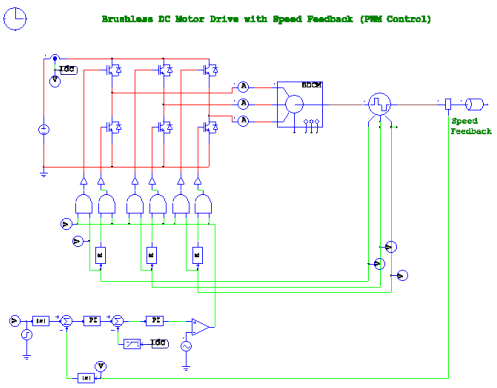
III. METODE PENELITIAN DAN HASIL

Langkah awal yang dilakukan adalah mempelajari *plant* yang akan digunakan dalam sistem kendali ini. Pada gbr.7 merupakan spesifikasi Motor BLDC yang digunakan sebagai *plant*, dengan daya 60 Watt, tegangan 24 Volt, arus 2.9 Ampere, 3500 rpm, serta data parameter motor diperoleh melalui pengujian dilaboratorium.

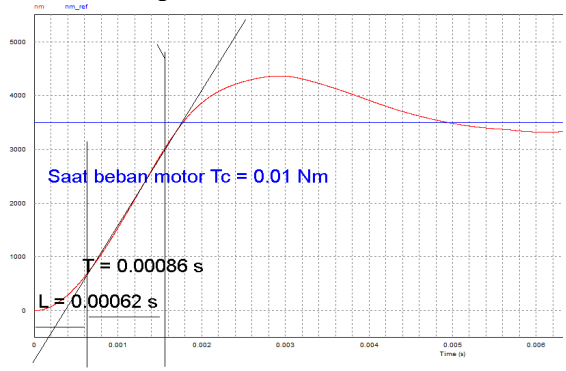
Kemudian dengan simulasi rangkaian percobaan dengan menggunakan program simulator PSIM, motor BLDC dilakukan percobaan pembebanan untuk mendapatkan hasil *dead time* dan *delay time* dengan metode kurva proses reaksi. Dimana dari hasil kurva reaksi ini nanti akan diperoleh nilai-nilai k_p, k_i dan k_d maksimal dan minimal dengan metode *Ziegler Nichols.*



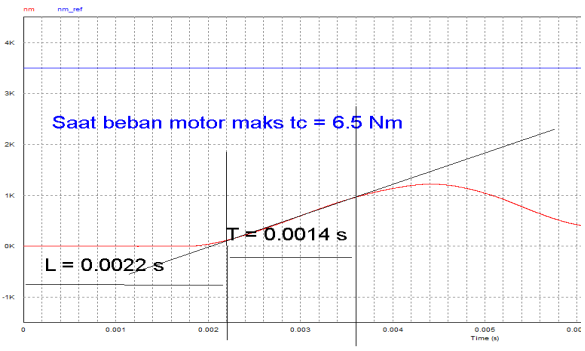
Gbr 7. Data Parameter Motor



Gbr 8. Rangkaian Percobaan Simulator PSIM



Gbr 9. Motor BLDC dengan beban torque 0.01Nm



Gbr 10. Motor BLDC dengan beban torque 6.5Nm

Dengan hasil pada gbr 9 dan 10, dimasukan pada persamaan (18), (19) dan (20)

$$K_p' = \left[\frac{K_p - K_{p\,min}}{K_{p\,max} - K_{p\,min}}\right] \quad (18)$$

$$K_d' = \left[\frac{K_{pd} - K_{d\,min}}{K_{d\,max} - K_{d\,min}}\right] \quad (19)$$

$$K_i = K_p / \alpha T_d \quad (20)$$

Sehingga diperoleh persamaan (21), (22), (23):

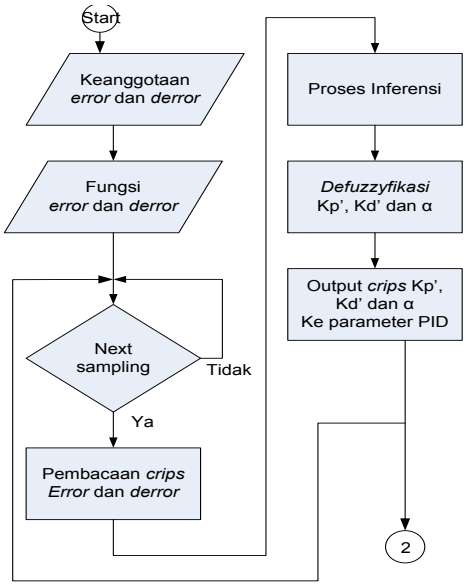
$$K_p = \text{K'p}\ (0{,}368) + 1.1316 \quad (21)$$

$$K_d = \text{K'd}\ (0.00048) + 0.00035 \quad (22)$$

$$K_i = 1{,}1316 / \alpha\ .0.00031 \quad (23)$$

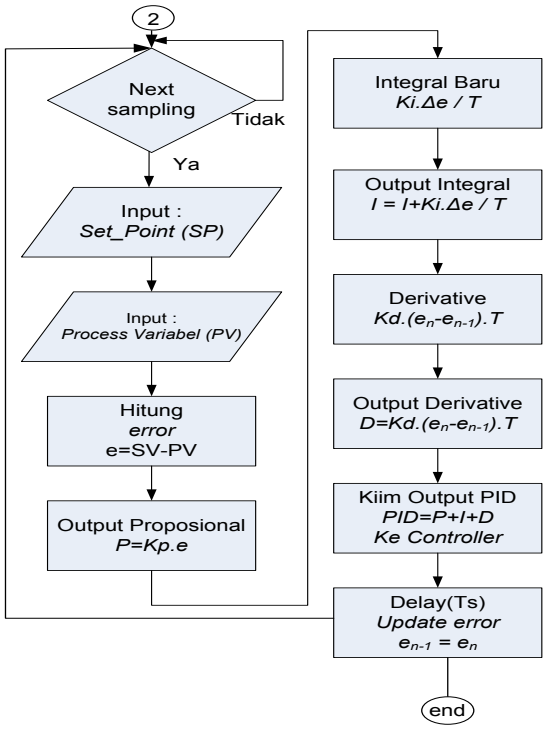
Dari persamaan diatas nilai K_p , K_d dan K_i akan berubah-ubah sesuai dengan pembacaan hasil *defuzyfikasi* ketiga parameter Kp', Kd' dan α.

Sedangkan pembacaan nilai Kp', Kd' dan α adalah seperti diagram alir gbr 11. Sehingga dapat direalisasikan pada rutin *fuzzy* kedalam bahasa pemrograman mikrokendalier. Tahap pertama adalah membentuk definisi keanggotaan *error* dan *derror* kedalam struktur *array*, kemudian membentuk fungsi keanggotaan tiap variabel keanggotaan *error* dan *derror*.



Gbr 11. *Diagram alir* Kp', Kd' dan α

Dari diagram alir gbr 11, terjadi eksekusi program di setiap waktu *sampling* maka besar nilai *crips error* dan *derror* akan dimasukkan kedalam fungsi keanggotaannya. Hasil dari pembacaan fungsi keanggotaan dilakukan proses *inferensi*, selanjutnya dilakukan proses defuzzyfikasi untuk menentukan nilai Kp', Kd' dan α. Nilai hasil defuzyfikasi merupakan nilai tegas yang nantinya digunakan oleh kendali PID sebagai variabel penalaannya.

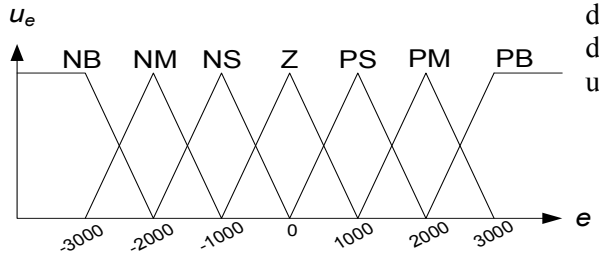


Gbr 12. Diagram Alir Kendali PID

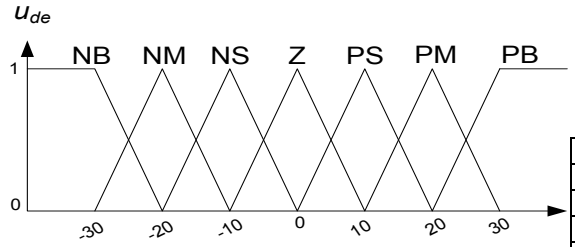
Dari diagram alir gbr.12 dapat direalisasi kedalam bahasa pemrograman mikrokendalier sebagai berikut : Ketika terjadi periode waktu *sampling*, sistem akan menunggu apakah ada penekanan *keypad* jika ada, kemudian memasukkan nilai *set_point* (SP) melalui penekanan *keypad*. Kemudian besaran *variabel process* (PV) akan selalu dibaca nilainya dan digunakan sebagai nilai pengurang terhadap SP untuk mendapatkan harga *error*. Untuk menghasilkan keluaran *proporsional* maka nilai *error* dikalikan dengan penguatan *proporsional*. Sedangkan harga keluaran *integral* merupakan besarnya akumulasi *error* dikalikan dengan penguatan *integral*. Dan harga keluaran *derivatif* adalah harga perubahan *error* dikalikan dengan penguatan *derivatif*. Setelah ketiga parameter PID didapatkan kemudian akan diumpankan ke kendali. Proses ini terjadi selama satu perioda *sampling* sebesar waktu tunda (ts). Selanjutnya harga *error* selama satu perioda *sampling* perlu diperbarui untuk perhitungan nilai *derivatif* berikutnya.

Nilai *error* dibagi kedalam tujuh aras (NB, NM, NS, Z, PS, PM, PB), sedangkan nilai perubahan *derror* juga dibagi kedalam tujuh aras (DNB, DNM, DNS, DZ, DPS, DPM, DPB). Huruf pertama N, P dan D berarti *negative, positive* dan *delta*, sedangkan huruf kedua B, M, S dan Z berarti *big, medium, small* dan *zero*.

Pada penelitian menggunakan fungsi bentuk segitiga sebab bentuk fungsi segitiga lebih mudah diterapkan dalam pembuatan program. Didalam rutin program *fuzzy* nilai *set point* dikalikan dengan 10



(a)



(b)

Gbr 13. (a) Keanggotan input *error* (b)Keanggotaan Input *derror*

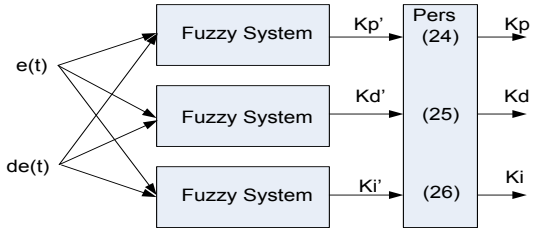
dikurangi nilai dari kecepatan motor melalui masukan ADC 10 bit. Nilai masukan ADC digunakan sebagai nilai pengurang terhadap *set point* untuk menghasilkan besarnya *error*. Sehingga rentang kecepatan motor yang dapat dibaca sebesar 100 sampai 3000. Nilai *error* yang terbaca dapat dipetakan menurut fungsi keanggotaan input *derror* dan perubahan *derror* seperti gbr 13.

$$K_p' = \frac{\sum_{l=1}^{49} Y_p \; U_A\left(e(t)\right) U_B\left(de(t)\right)}{\sum_{l=1}^{49} \quad U_A\left(e(t)\right) U_B\left(de(t)\right)} \tag{24}$$

$$K_d' = \frac{\sum_{l=1}^{49} Y_d \; U_A\left(e(t)\right) U_B\left(de(t)\right)}{\sum_{l=1}^{49} \quad U_A\left(e(t)\right) U_B\left(de(t)\right)} \tag{25}$$

$$K_\alpha' = \frac{\sum_{l=1}^{49} Y_\alpha \; U_A\left(e(t)\right) U_B\left(de(t)\right)}{\sum_{l=1}^{49} \quad U_A\left(e(t)\right) U_B\left(de(t)\right)} \tag{26}$$

Dari aturan pada gbr 14, maka dapat ditentukan aturan ketiga konstanta PID. Selengkapnya sesuai persamaan (24), (25), dan (26) pernyataan *fuzzy* tersebut dapat ditabelkan pada tabel 2, tabel 3 dan tabel 4, yang kemudian akan di masukkan sebagai data pemrograman mikrokendalier yang digunakan system kendali *hybrid PID-Fuzzy* untuk mengatur BLDC motor.



Gbr 14. Fuzzy system penala kendali PID

Tabel 2 Aturan untuk Kp'

	DNB	DNM	DNS	DZ	DPS	DPM	DPB
NB	B	B	B	B	B	B	B
NM	S	B	B	B	B	B	S
NS	S	S	B	B	B	S	S
Z	S	S	S	B	S	S	S
PS	S	S	B	B	B	S	S
PM	S	B	B	B	B	B	S
PB	B	B	B	B	B	B	B

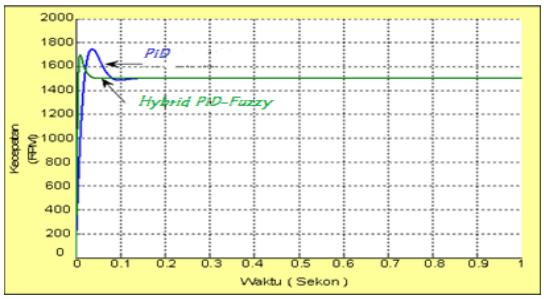
Tabel 3 Aturan untuk Kd'

	DNB	DNM	DNS	DZ	DPS	DPM	DPB
NB	S	S	S	S	S	S	S
NM	B	B	S	S	S	B	B
NS	B	B	B	S	B	B	B
Z	B	B	S	S	S	B	B
PS	B	B	B	S	B	B	B
PM	B	B	S	S	S	B	B
PB	S	S	S	S	S	S	S

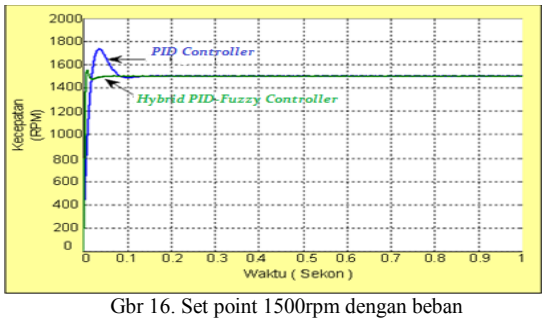
Tabel 4 Aturan untuk α

	DNB	DNM	DNS	DZ	DPS	DPM	DPB
NB	2	2	2	2	2	2	2
NM	3	3	2	2	2	3	3
NS	4	3	3	2	3	3	4
Z	5	4	3	3	3	4	5
PS	4	3	3	2	3	3	4
PM	3	3	2	2	2	3	3
PB	2	2	2	2	2	2	2

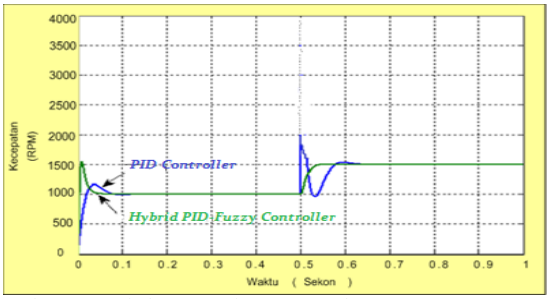
Gbr 15-20 merupakan gambar data-data pengukuran hasil uji dilakukan dengan memperlakukan BLDC Motor pada variasi kecepatan 1000 rpm dan 1500 rpm dengan tanpa beban serta dengan beban 0.5kg.



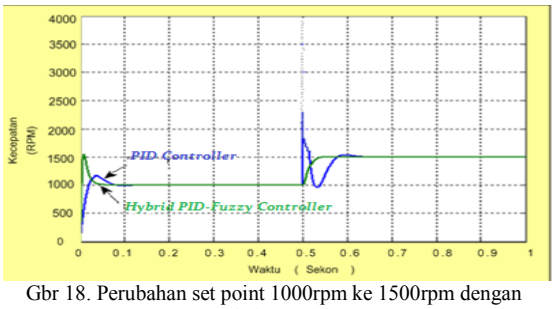
Gbr 15. Set point 1500 rpm tanpa beban



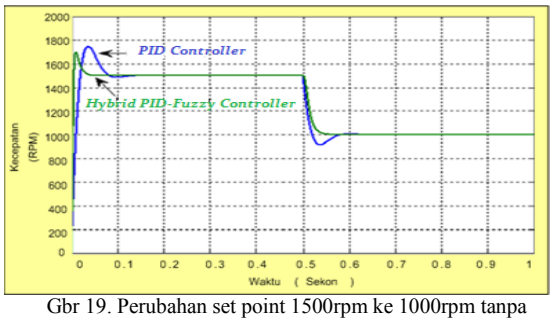
Gbr 16. Set point 1500rpm dengan beban



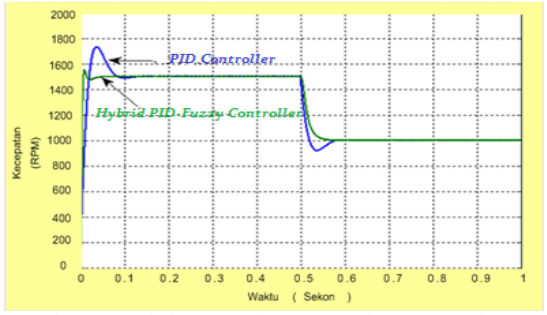
Gbr 17. Perubahan set point 1000rpm ke 1500rpm tanpa beban



Gbr 18. Perubahan set point 1000rpm ke 1500rpm dengan beban



Gbr 19. Perubahan set point 1500rpm ke 1000rpm tanpa beban



Gbr 20. Perubahan set point 1500rpm ke 1000rpm dengan beban

Tabel 5 menunjukkan hasil uji percobaan yang dilakukan dengan mengubah-ubah set point (kecepatan) dari 500 rpm sampai 3500rpm. Juga dilakukan pula uji percobaan dengan mengubah set point secara tiba-tiba dalam selang waktu 0,5 detik dari set point 1000 rpm menuju ke 1500 rpm tanpa beban dan dengan beban 0,5 kg, juga dilakukan uji perubahan set point dari 1500 rpm menuju 1000 rpm dengan selang waktu 0,5 detik.

Tabel 5.
Data hasil uji percobaan

Kecepatan	**Kendali PID**			**Kendali *hybrid PID-Fuzzy***		
Set Point	Rise Time (sec)	Over-shoot (%)	Settling Time (sec)	Rise Time (sec)	Over-shoot (%)	Settling Time (sec)
500 tanpa beban	0.025	7.50	0.175	0.0025	7.00	0.05
500 dengan beban	0.025	7.50	0.25	0.0025	5.50	0.075
1000 tanpa beban	0.025	12.50	0.175	0.0025	12.00	0.05
1000 dengan beban	0.025	12.50	0.25	0.0025	10.50	0.075
1500 tanpa beban	0.025	17.50	0.175	0.0025	17.00	0.05
1500 dengan beban	0.025	17.50	0.25	0.0025	15.50	0.075
2000 tanpa beban	0.025	22.50	0.175	0.0025	22.00	0.05
2000 dengan beban	0.025	22.50	0.25	0.0025	20.50	0.075
2500 tanpa beban	0.025	27.50	0.175	0.0025	27.00	0.05
2500 dengan beban	0.025	27.50	0.25	0.0025	25.50	0.075
3000 tanpa beban	0.025	32.50	0.175	0.0025	32.00	0.05
3000 dengan beban	0.025	32.50	0.25	0.0025	30.50	0.075
3500 tanpa beban	0.025	37.50	0.175	0.0025	37.00	0.05
3500 dengan beban	0.025	37.50	0.25	0.0025	35.50	0.075
1000 – 1500 tanpa beban	0.025	20.00	0.1375	0.0025	0	0.05
1000 – 1500 dengan beban	0.025	22.50	0.150	0.0025	0	0.044
1500 – 1000 tanpa beban	0.025	9.20	0.1875	0.0025	0	0.05
1500 – 1000 dengan beban	0.025	9.20	0.075	0.0025	0	0.075

IV. KESIMPULAN

Dari hasil penelitian dan pembahasan, dapat disimpulkan bahwa proses kendali dengan sistem penalaan parameter kendali PID dengan logika *Fuzzy* yang diaplikasikan untuk mengatur Motor BLDC dapat memperbaiki kinerja kendali PID konvensional. Pengujian perubahan *set point* dan perubahan beban, dihasilkan karakteristik tanggapan sistem kendali PID konvensional dengan nilai rata-rata yaitu waktu kenaikan (tr) 0.025 detik, waktu penetapan (ts) 0.1625 detik, *overshoot* sebesar 15.98%. Sedangkan kendali *hybrid PID Fuzzy* dihasilkan nilai rata-rata waktu kenaikan (tr) 0.0025 detik, waktu penetapan (ts) 0.057 detik, *overshoot* sebesar 5.42%.

Dapat disimpulkan bahwa kendali *hybrid PID Fuzzy* mampu meningkatkan kinerja dari kendali PID konvensional.

V. REFERENSI

[1] Gunterus, Frans. 1994, Falsafah Dasar: Sistem Pengendalian Proses, jakarta. PT. Elex Media Komputindo..

[2] Wang, L. X. 1997, A Course in Fuzzy Systems and Control, New Jersey: Prentice-Hall International. Inc: pp. 257-263.

[3] M. Depenbrok, IEEE Trans, On Power Electronics 3 (1988) 420.

[4] J.M Jacob, Industrial Control Electronics Application and Design, Prentise Hall Inc. Englewood Cliffs, New Jersey, 1988.

[5] Y. S. Lai, Proceedings of the IEEE PES Winter Meeting, 1999, p.47

[6] C.T Lin, C.S Lee, Neural Fuzzy Systems, Prentice Hall Inc, Englewood Cliffs, New Jersey, 1996

UCAPAN TERIMAKASIH

Terimakasih kepada Jurusan Teknik Elektro Akademi Teknologi Warga Surakarta yang telah memberi fasilitas dalam penelitian ini. Kepada Wiyono dan Budi Nugroho yang telah banyak membantu dan memberi dukungan.