

**DESAIN OPTIMAL
KONTROLER PROPORSIONAL INTEGRAL
MOTOR DC MENGGUNAKAN
ALGORITME *PARTICLE SWARM OPTIMIZATION***

Andarini Asri¹, Muhammad Ruswandi Djalal², dan Rahmat²

¹Jurusan Teknik Elektro, Politeknik Negeri Ujung Pandang

²Jurusan Teknik Energi, Politeknik Negeri Ujung Pandang

Jalan Perintis Kemerdekaan Km.10, Tamalanrea, Makassar 90245

E-mail: ¹andarinasri@gmail.com, ²wandi@poliupg.ac.id

ABSTRACT

In this paper, we propose a method to control DC motors using Proportional-Integral (PI), in which PI parameters are determined using the intelligent Particle Swarm Optimization (PSO) algorithm. PSO tuning are done using PSIM software, with 5 swarm used to find effective parameters K_p and K_i . The value of K_p is 1.447, and K_i is 4.271. Using three parameters that is overshoot, rise time, and settling time, it can be concluded that PSO is better and faster than Ziegler Nichols method. With setting speed value 1000 rpm, overshoot using Ziegler Nichols method reach 1200 rpm whereas for the PSO only up to 1000 rpm. The rise time of PSO is better than Ziegler Nichols because the rise time is smaller. Settling time of PSO is less than or faster than Ziegler Nichols. The overall results show that PSO gives a better speed response than Ziegler Nichols method.

Keywords: Particle Swarm Optimization, Proportional-Integral, Ziegler-Nichols, settling time, rise time

ABSTRAK

Dalam penelitian ini diusulkan satu metode untuk mengatur kecepatan motor DC dengan kontroler Proporsional-Integral (PI) di mana parameter PI diperoleh dengan menggunakan algoritme cerdas Particle Swarm Optimization (PSO). Penalaan PSO dilakukan dengan menggunakan perangkat lunak PSIM, dengan 5 swarm digunakan untuk mengetahui parameter K_p dan K_i yang efektif. Dari hasil simulasi diperoleh nilai K_p sebesar 1,447 dan K_i sebesar 4,271. Bila menggunakan tiga parameter penilaian yaitu overshoot, risetime, dan settling time, dapat disimpulkan bahwa PSO lebih baik dan lebih cepat dari pada metode Ziegler Nichols. Pada setting kecepatan 1000 rpm, overshoot pada metode Ziegler Nichols mencapai 1200 rpm, sedangkan pada PSO hanya sampai 1000 rpm. Rise time pada PSO lebih baik dari pada Ziegler Nichols karena nilainya lebih kecil. Settling time pada PSO

membutuhkan waktu lebih sedikit atau lebih cepat dari Ziegler Nichols. Dari hasil keseluruhan diperoleh respons kecepatan dengan metode PSO lebih baik dibandingkan dengan metode Ziegler Nichols.

Kata kunci: *Particle Swarm Optimization, Proporsional-Integral, Ziegler-Nichols, settling time, rise time*

1. PENDAHULUAN

Sistem kontrol proporsional, integral, derivatif (PID) dengan struktur sederhana, stabilitas yang baik, dan ketahanan yang kuat sudah banyak digunakan dalam industri [1]. Motor DC telah banyak digunakan dalam aplikasi industri. Masalah utama yang sering dibahas dalam kontrol PID adalah penyesuaian parameter. Salah satu metode yang sering digunakan adalah metode konvensional *trial-error*, yaitu menala parameter PID dengan cara coba-coba, namun dengan metode ini sulit untuk menyesuaikan parameter, sehingga dibutuhkan waktu lama [2], serta tingkat akurasi kontrol tidak baik. Dalam beberapa tahun terakhir, para peneliti telah mengembangkan beberapa metode cerdas (*artificial intelligent*) untuk penentuan parameter PID motor DC. Perilaku hewan, antara lain lebah, semut, dan tawon dipelajari dan dibuatkan algoritme berdasarkan perilaku hewan tersebut, selanjutnya algoritme tersebut diaplikasikan untuk menyelesaikan berbagai masalah optimasi.

Beberapa metode optimasi berbasis metode konvensional maupun metode cerdas telah banyak digunakan untuk mengoptimasi parameter PID motor DC, diantaranya *Artificial Bee Colony* [1], *Genetic Algorithm* [3], *Evolutionary Algorithm* [4], *Bio-Inspired Algorithm* [5], *Bacterial Foraging* [6], *Tabu Search* [7], *Fuzzy Logic* [8], [9], *Cuckoo Search* [10] dan *Flower* [11].

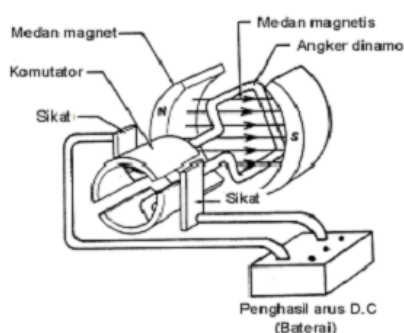
Parameter PID terkadang tidak dapat bekerja dengan baik pada *plant* tertentu karena ada suatu kombinasi yang optimal antara P, I, dan D; dengan kata lain, kontroler PID tidak dapat bekerja dengan optimal pada semua *plant* [12] - [14]. Pada penelitian ini akan diusulkan salah satu metode cerdas berbasis *Particle Swarm Optimization* (PSO) untuk optimasi parameter proporsional dan integral (PI) dari kontroler motor DC.

2. KAJIAN PUSTAKA

2.1 Pengertian Motor DC

Motor DC memerlukan suplai tegangan yang searah pada kumparan medan untuk diubah menjadi energi mekanik. Bagian utama motor DC adalah stator dan rotor di mana kumparan medan pada motor dc disebut stator (bagian yang tidak berputar) dan kumparan jangkar disebut rotor (bagian yang berputar). Pada Gambar 1 diperlihatkan motor paling sederhana dengan kumparan satu lilitan yang bisa berputar bebas di antara kutub-kutub magnet permanen.

Catu tegangan DC dari baterai menuju ke lilitan melalui sikat yang menyentuh komutator dua segmen yang terhubung dengan dua ujung lilitan. Kumparan satu lilitan yang merupakan komponen yang berputar di antara medan magnet disebut angker dinamo dapat dilihat pada Gambar 1.



Gambar 1 Motor DC Sederhana

2.2 Prinsip Arah Putaran Motor

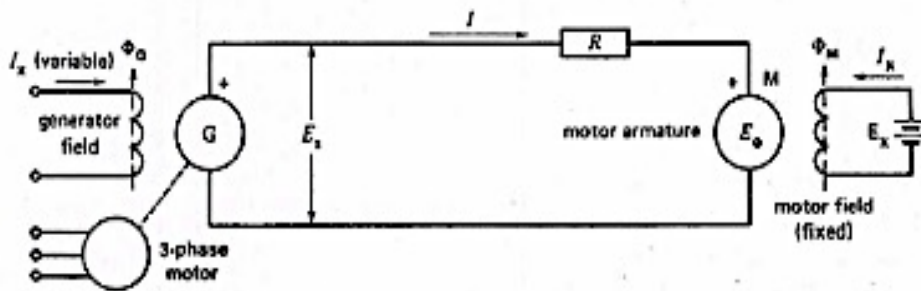
Untuk menentukan arah putaran motor digunakan kaidah Fleming tangan kiri. Kutub-kutub magnet akan menghasilkan medan magnet dengan arah dari kutub utara ke kutub selatan. Jika medan magnet memotong sebuah kawat penghantar yang dialiri arus searah dengan empat jari, maka akan timbul gerak searah ibu jari. Gaya ini disebut gaya Lorentz. Prinsip motor adalah aliran arus pada penghantar yang berada di dalam pengaruh medan magnet akan menghasilkan gerakan. Besarnya gaya pada penghantar akan bertambah besar jika arus yang melalui penghantar bertambah besar.

2.3 Pengaturan Kecepatan *Armature*

Berdasarkan Persamaan (1) di bawah ini

$$n = \frac{60.E_s}{Z\phi} \quad (1)$$

jika flux Φ tetap dijaga konstan dan Z adalah impedansi motor, kecepatan (n) berubah berdasarkan *armature voltage* (E_s). Jika E_s naik atau turun maka kecepatan motor akan naik atau turun.



Gambar 2 Rangkaian Motor Mengatur Kecepatan pada Belitan Jangkar (*Armature*)

Pada Gambar 2 diperlihatkan contoh rangkaian pengaturan kecepatan motor AC. Dapat dilihat bahwa E_s dapat divariasikan dengan menghubungkan motor *armature* M ke variabel eksitasi generator G yang berbeda. *Field excitation* dari motor dijaga tetap konstan, tetapi arus generator I_x bisa divariasikan dari nol sampai maksimum dan bahkan sebaliknya. Oleh sebab itu tegangan keluaran generator E_s bisa divariasikan dari nol sampai maksimum, baik dalam polaritas positif maupun negatif. Hal ini menyebabkan kecepatan motor dapat divariasikan dari nol sampai maksimum dalam dua arah. Metode pengontrolan kecepatan ini dikenal sebagai sistem *Ward-Leonard* dan dapat ditemukan di pabrik baja (*steel mills*), lift bertingkat, pertambangan, dan pabrik kertas.

Dalam instalasi modern, generator sering digantikan dengan *high-power electronic converter* yang mengubah daya AC menjadi DC. Sistem Ward-Leonard merupakan cara menerapkan suatu variabel DC ke *armature* dari motor DC. Hal tersebut benar-benar dapat memaksa motor untuk mengembangkan torsi dan

kecepatan yang dibutuhkan oleh beban. Contohnya, jika E_s dibuat sedikit lebih tinggi dari E_o motor maka arus akan mengalir dengan arah sesuai dengan gambar di atas, dan motor mengembangkan torsi yang positif. *Armature* dari motor menyerap daya karena I mengalir ke terminal positif. Selanjutnya jika kita mengurangi E_s dengan mengurangi eksitasi Φ_G , segera setelah E_s menjadi kurang dari E_o , arus I berbalik. Hasilnya, torsi motor berbalik dan *armature* dari motor menghantarkan daya ke generator G . Akibatnya, motor DC mendadak menjadi generator dan generator G mendadak menjadi motor. Motor dapat dipaksa untuk melambat dengan mengurangi E_s . Saat menerima daya listrik, generator beroperasi sebagai motor, mengendalikan motor AC-nya sendiri sebagai *asynchronous generator*. Hasilnya, daya AC diumpankan kembali ke rangkaian yang mensuplai motor AC.

2.4 Tuning Parameter PI Menggunakan Algoritme PSO

Particle Swarm Optimization didasarkan pada perilaku sebuah kawanan serangga, seperti semut, rayap, lebah atau burung. Algoritme PSO meniru perilaku sosial organisme ini. Perilaku sosial terdiri dari tindakan individu dan pengaruh dari individu-individu lain dalam suatu kelompok. Kata partikel menunjukkan, misalnya, seekor burung dalam kawanan burung. Setiap individu atau partikel berperilaku secara terdistribusi dengan cara menggunakan kecerdasannya (*intelligence*) sendiri dan juga dipengaruhi perilaku kelompok kolektifnya. Jika satu partikel atau seekor burung menemukan jalan yang tepat atau pendek menuju ke sumber makanan, sisa kelompok yang lain juga akan dapat segera mengikuti jalan tersebut meskipun lokasi mereka jauh di kelompok tersebut.

Metode optimasi yang didasarkan pada *swarm intelligence* ini disebut algoritme *behaviorally inspired* sebagai alternatif dari algoritme genetika, yang sering disebut *evolution-based procedures*. Dalam konteks optimasi multivariabel, kawanan diasumsikan mempunyai ukuran tertentu atau tetap dengan setiap partikel posisi awalnya terletak di suatu lokasi yang acak dalam ruang multidimensi. Setiap partikel diasumsikan memiliki dua karakteristik yaitu posisi dan kecepatan. Setiap partikel bergerak dalam ruang/*space* tertentu dan mengingat posisi terbaik yang

pernah dilalui atau ditemukan terhadap sumber makanan atau nilai fungsi objektif. Setiap partikel menyampaikan informasi atau posisi terbaiknya kepada partikel yang lain dan menyesuaikan posisi dan kecepatan masing-masing berdasarkan informasi yang diterima mengenai posisi yang terbaik tersebut. Sebagai contoh, misalnya perilaku burung-burung dalam kawanan burung. Meskipun setiap burung mempunyai keterbatasan dalam hal kecerdasan, biasanya ia akan mengikuti kebiasaan (*rule*) seperti berikut:

1. Seekor burung tidak berada terlalu dekat dengan burung yang lain.
2. Burung tersebut akan mengarahkan terbangnya ke arah rata-rata keseluruhan burung.
3. Akan memposisikan diri dengan rata-rata posisi burung yang lain dengan menjaga sehingga jarak antar burung dalam kawanan itu tidak terlalu jauh.

Dengan demikian perilaku kawanan burung akan didasarkan pada kombinasi dari tiga faktor simpel berikut: 1) kohesi - terbang bersama; 2) separasi - jangan terlalu dekat; dan 3) penyesuaian (*alignment*) - mengikuti arah bersama.

Particle Swarm Optimization (PSO) dikembangkan dengan berdasarkan pada model berikut:

1. Ketika seekor burung mendekati target atau makanan (atau minimum atau maksimum suatu fungsi tujuan), secara cepat mengirim informasi kepada burung-burung yang lain dalam kawanan tertentu.
2. Burung yang lain akan mengikuti arah menuju ke makanan tetapi tidak secara langsung.
3. Ada komponen yang tergantung pada pikiran setiap burung, yaitu memorinya tentang apa yang sudah dilewati pada waktu sebelumnya.

Model ini akan disimulasikan dalam ruang dengan dimensi tertentu dengan sejumlah iterasi sehingga di setiap iterasi, posisi partikel akan semakin mengarah ke target yang dituju, dengan kata lain meminimalkan (*minimization*) atau memaksimalkan (*maximization*) fungsi. Ini dilakukan hingga maksimum iterasi dicapai atau bisa juga digunakan kriteria penghentian yang lain.

3. METODE PENELITIAN

3.1 Pengaturan Kecepatan Motor DC

Motor DC dapat diatur kecepatannya dengan cara mengatur arus yang masuk ke kumparan jangkar yaitu dengan cara memberi resistansi, bisa juga dengan mengatur tegangan DC yang masuk. Dalam penelitian ini, pengaturan kecepatan motor dilakukan dengan cara mengatur tegangan yang masuk ke kumparan jangkar dengan menggunakan konverter DC-DC tipe *buck-boost*.

Kumparan medan diberi sumber DC yang nilai tegangannya tetap 100 volt, kemudian kumparan jangkar diberi sumber DC yang dapat diatur tegangannya oleh konverter DC-DC. Pada Gambar 2 diperlihatkan gambar rangkaian yang digunakan pada penelitian ini. Sistem yang digunakan adalah *close loop*. Motor diberi sensor putaran, kemudian kecepatan putaran diatur 1000 rpm menggunakan kontrol PI.

3.2 Implementasi PSO

Misalkan terdapat fungsi berikut

$$\min f(x) \tag{2}$$

dengan $X(B) \leq X \leq X(A)$. $X(B)$ adalah batas bawah dan $X(A)$ adalah batas atas dari X . Prosedur PSO dijabarkan di bawah ini.

1. Ukuran kelompok atau kawanan (jumlah partikel) diasumsikan sebagai N . Untuk mengurangi jumlah evaluasi fungsi yang diperlukan untuk menemukan solusi, sebaiknya ukuran N tidak terlalu besar, tetapi juga tidak terlalu kecil, agar ada banyak kemungkinan posisi menuju solusi terbaik atau optimal. Jika terlalu kecil sedikit kemungkinan menemukan posisi partikel yang baik, jika terlalu besar juga akan membuat perhitungan menjadi panjang. Biasanya digunakan ukuran kawanan 20 sampai 30 partikel.
2. Populasi awal X dibangkitkan pada rentang $X(B)$ dan $X(A)$ secara acak sehingga didapat X_1, X_2, \dots, X_N . Setelah itu, untuk mudahnya, partikel j dan kecepatannya pada iterasi I dinotasikan sebagai $X_1(0), X_2(0), \dots, X_N(0)$. Vektor $X_j(0)$,

($j=1,2,\dots,N$) disebut partikel atau vektor koordinat dari partikel, contohnya kromosom pada algoritme genetika. Evaluasi dari fungsi tujuan untuk setiap partikel dan nyatakan dengan

$$f[X_1(0)], f[X_2(0)], \dots f[X_N(0)] \quad (3)$$

3. Kecepatan dari semua partikel dihitung. Semua partikel bergerak menuju titik optimal dengan suatu kecepatan. Awalnya semua kecepatan dari partikel diasumsikan sama dengan nol. Set iterasi $i=1$.
4. Pada iterasi ke-I, dicari dua parameter penting untuk setiap partikel j yaitu:
 - a. nilai terbaik sejauh ini dari $X_j(i)$ (koordinat partikel j pada iterasi i) dan nyatakan sebagai $P_{best,j}$ dengan nilai fungsi obyektif paling rendah (kasus *minimization*), $f[X_j(i)]$, yang ditemui sebuah partikel j pada semua iterasi sebelumnya. Nilai terbaik untuk semua partikel $X_j(i)$ yang ditemukan sampai iterasi ke-I, G_{best} , dengan nilai fungsi tujuan paling kecil atau minimum di antara semua partikel untuk semua iterasi sebelumnya, $f[X_j(i)]$.
 - b. Kecepatan partikel j pada iterasi ke I dihitung dengan rumus sebagai berikut

$$V_{ji}=V_{ji-1}+c_1r_1P_{best,j}-x_{ji-1}+c_2r_2G_{best}-x_{ji-1} \quad (4)$$

dengan $j=1, 2, 3 \dots N$. Nilai c_1 dan c_2 masing-masing adalah *learning rates* untuk kemampuan individu (*cognitive*) dan pengaruh sosial (*group*), serta r_1 dan r_2 bilangan acak yang berdistribusi uniformal dalam interval 0 dan 1. Jadi parameter c_1 dan c_2 menunjukkan bobot dari memori (*position*) sebuah partikel terhadap memori dari kelompok (*swarm*). Nilai dari c_1 dan c_2 biasanya adalah 2 sehingga perkalian c_1r_1 dan c_2r_2 memastikan bahwa partikel-partikel akan mendekati target sekitar setengah selisihnya. Hitung posisi atau koordinat partikel j pada iterasi ke-I dengan cara

$$X_j(i) = X_j(i - 1) + V_j(i) \quad (5)$$

dengan $j=1, 2, 3 \dots N$. Evaluasi nilai fungsi tujuan untuk setiap partikel dan nyatakan sebagai $f[X_1(i)], f[X_2(i)], \dots, f[X_N(i)]$.

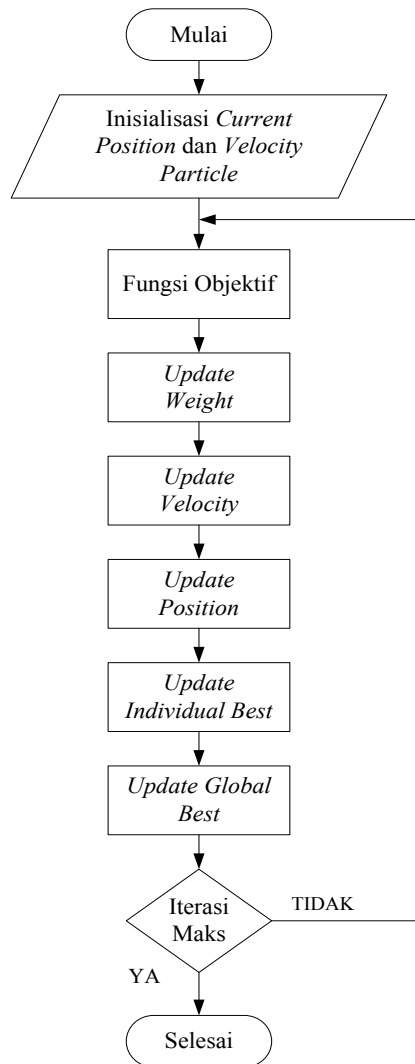
5. Memeriksa apakah solusi yang sekarang sudah konvergen. Jika posisi semua partikel menuju ke satu nilai yang sama, maka ini disebut konvergen. Jika belum konvergen maka langkah 4 diulang dengan memperbarui iterasi $i = i + 1$, dengan cara menghitung nilai baru dari $P_{best,j}$ dan G_{best} . Proses iterasi ini dilanjutkan sampai semua partikel menuju ke satu titik solusi yang sama. Biasanya akan ditentukan dengan kriteria penghentian (*stopping criteria*), misalnya jumlah selisih solusi sekarang dengan solusi sebelumnya sudah sangat kecil.

3.3 Implementasi *Particle Swarm Optimization* untuk *Tuning PI*

Berikut ini adalah langkah-langkah PSO untuk *tuning PI*:

1. Menentukan parameter awal dari PSO seperti $c = 0,2$ dan $w = 0,05$.
2. Menentukan nilai K_p dan K_i dengan menggunakan metode *Ziegler Nichols*.
3. Menentukan banyaknya *swarm* ($n=5$).
4. Menentukan posisi awal dari tiap *swarm* (sumbu $x=K_p$, sumbu $y=K_i$).
5. Mencoba nilai posisi (K_p dan K_i) tiap-tiap *swarm* yang telah dipilih dan dilihat responsnya.
6. Mengambil hasil nilai *Overshoot (OV)*, *Rise Time (RT)*, dan *Settling Time (ST)* dari masing-masing *swarm* yang telah dipilih.
7. Menentukan *objective function* dari tiap-tiap *swarm* menggunakan rumus
 - a.
$$\frac{OV}{X1} + \frac{RT}{X2} + \frac{ST}{X3} \tag{6}$$
 - b. Nilai yang paling kecil merupakan *swarm* yang paling optimal. Di mana P_{best} adalah nilai paling optimal di *swarm* itu sendiri, dan G_{best} adalah nilai paling optimal dari keseluruhan *swarm*.
8. Memeriksa apakah solusi yang sekarang sudah konvergen. Jika posisi *overshoot*, *rise time*, dan *settling time* semakin kecil, maka solusi hampir konvergen. Jika belum maka diulang dengan memperbarui iterasi $i = i + 1$, dengan cara menghitung nilai baru dari P_{best} dan G_{best} .

Gambar 3 menunjukkan *flowchart* algoritme *Particle Swarm Optimization* dalam melakukan proses komputasi menyelesaikan problem optimasi global.

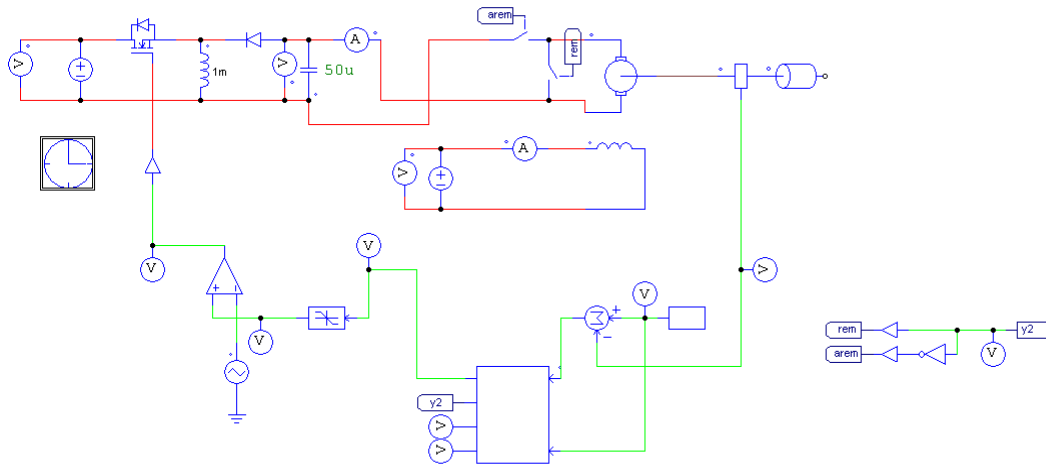


Gambar 3 *Flowchart* Algoritme PSO

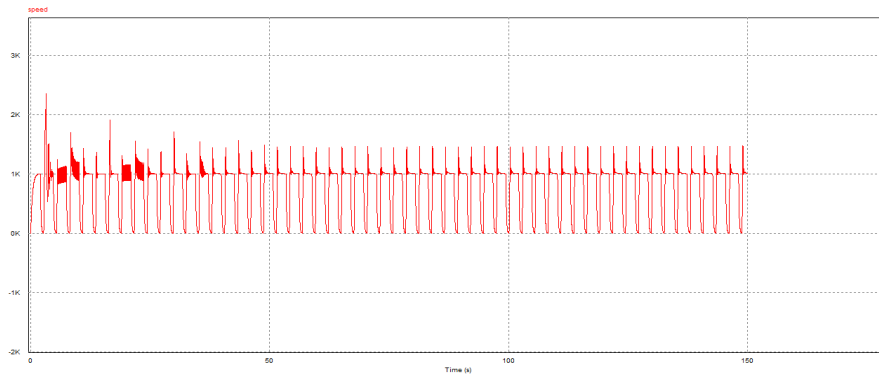
4. HASIL DAN PEMBAHASAN

Gambar 4 menunjukkan desain kontroler kecepatan motor DC dengan Proporsional-Integral (PI). Parameter PI tersebut ditala (*tune*) menggunakan algoritme cerdas berbasis *Particle Swarm Optimization* (PSO). Tujuan optimasi parameter PI adalah agar kontroler lebih optimal dalam menstabilkan respons

kecepatan motor DC. Beberapa parameter kontrol akan ditinjau untuk melihat keberhasilan metode yang diusulkan, seperti respons *overshoot*, *rise time*, dan *settling time*.



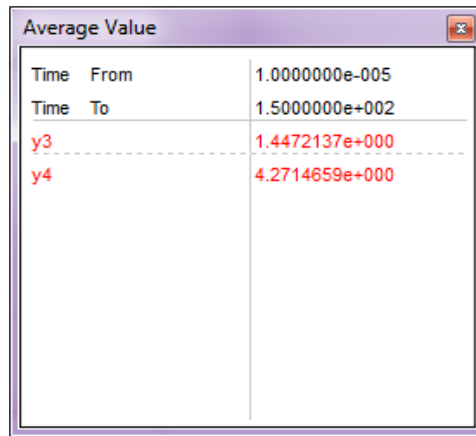
Gambar 4 Simulasi Kontrol Kecepatan Motor Menggunakan PI dan PSO pada Power Simulator (PSIM)



Gambar 5 Respons Kecepatan (n) vs Waktu (Time) dengan Kontroler PI yang Ditala Menggunakan PSO

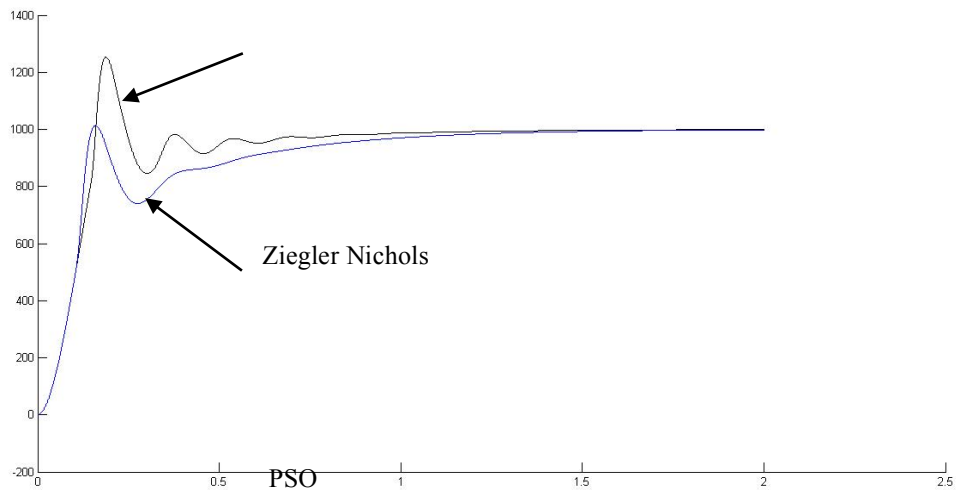
Gambar 5 menunjukkan respon kecepatan kontroler PI dengan penalaan PSO yang dilakukan menggunakan *software* Power Simulator (PSIM). Seperti yang telah dibahas sebelumnya, bahwa ada 5 *swarm* yang digunakan untuk mencari nilai efektif parameter K_p dan K_i . Gambar 6 dan 7 menunjukkan proses penalaan parameter PI

1000 rpm. Dengan menggunakan Ziegler Nichols diperoleh respons *overshoot* sebesar 1270 rpm yang melebihi *set point* dari sistem yaitu 1000 rpm.



Time	From	To
	1.0000000e-005	1.5000000e+002
y3		1.4472137e+000
y4		4.2714659e+000

Gambar 8 Hasil Penalaan Parameter K_p (y3) dan K_i (y4)



Gambar 9 Perbandingan Respons Kecepatan Motor Menggunakan Ziegler Nichols dan PSO

Nilai *settling time* dengan menggunakan PSO juga menunjukkan hasil yang lebih baik, di mana pada 0,5 detik sudah mencapai *steady state*, dibanding dengan Ziegler Nichols yang mencapai 0,8 detik. Dengan mengambil tiga acuan parameter

yaitu *overshoot*, *rise time* dan *settling time*, maka dapat disimpulkan bahwa PSO lebih baik dan lebih cepat daripada metode Ziegler Nichols. *Rise time* adalah posisi ketika 90% dari *setting* kecepatan tercapai, yang dalam penelitian ini *setting* kecepatan adalah 1000 rpm. Untuk *rise time*, PSO lebih baik daripada Ziegler Nichols karena *rise time*-nya lebih kecil daripada Ziegler Nichols. *Settling time* adalah waktu yang diperlukan oleh sistem untuk mencapai *steady state*. Untuk *settling time*, dari gambar terlihat bahwa PSO membutuhkan waktu yang lebih sedikit atau lebih cepat dibandingkan dengan Ziegler Nichols.

5. KESIMPULAN

1. Dari hasil penalaan dengan metode PSO diperoleh nilai K_p sebesar 1,447 dan K_i sebesar 4,271.
2. Berdasarkan parameter *overshoot*, *risetime*, dan *settling time*, dapat disimpulkan bahwa respons kecepatan motor dengan metode PSO lebih baik dan lebih cepat daripada metode Ziegler Nichols. Pada *setting* kecepatan 1000 rpm, *overshoot* menggunakan metode Ziegler Nichols mencapai 1200 rpm sedangkan dengan metode PSO hanya 1000 rpm. *Rise time* pada PSO lebih baik dari pada Ziegler Nichols karena nilainya lebih kecil. *Settling time* pada PSO yaitu 0,5 detik lebih cepat dari Ziegler Nichols yang besarnya 0,8 detik.

DAFTAR PUSTAKA

- [1] Wudai Liao, Yingyue Hu, Haiquan Wang. "Optimization of PID control for DC motor based on artificial bee colony algorithm," dalam *Proc. of the 2014 International Conference on Advanced Mechatronics System*, 2014, hlm. 23-27.
- [2] M.R.Djalal, D. Ajiatmo, A. Imran, I. Robandi. "Desain optimal kontroler PID motor DC menggunakan cuckoo search algorithm," dalam *Prosiding Seminar Nasional Teknologi Informasi dan Aplikasinya (SENTIA)*, 2015.
- [3] D. Chen, K. Fang, Q. Chen. "Application of genetic algorithm in PID parameters optimization." *Microcomputer Information*, Vol. 23, No.3, hlm. 35-36, 2007.

- [4] H. He, F. Qian. "The PID parameter tuning based on immune evolutionary algorithm." *Microcomputer Information*, Vol. 27, No. 5, hlm. 1174-1176, 2007.
- [5] Nitish Katal, Sanjay Kr. Singh. "Optimal Tuning of PID controller for DC motor using bio-inspired algorithms." *International Journal of Computer Applications*, Vol. 56, No. 2, Oktober 2012.
- [6] Bharat Bhushan, Madhusudan Singh. "Adaptive control of DC motor using bacterial foraging algorithm." *Applied Soft Computing*, Vol. 11, hlm. 4913-4920, 2011.
- [7] Anant Oonsivilai, Padej Pao-La-Or. "Optimum PID controller tuning for AVR system using adaptive tabu search," dalam *Proceeding of 12th WSEAS International Conference on COMPUTERS*, 2008, hlm. 987 – 992.
- [8] Ashu Ahuja, Sanjev Kumar Aggarwal. "Design of fractional order PID controller for DC motor using evolutionary optimization techniques." *WSEAS Transactions on Systems and Control*, Vol. 9, hlm. 171-182, 2014.
- [9] Umesh Kumar Bansal, Rakesh Narvey. "Speed control of DC motor using fuzzy PID controller." *Advance in Electronic and Electric Engineering*, Vol. 3, No. 3, hlm. 1209-1220, 2013.
- [10] K. Premkumar, B.V. Manikandan. "Fuzzy PID supervised online ANFIS based speed controller for brushless dc motor." *Neurocomputing*, Vol. 157, hlm. 76-90, Juni 2015.
- [11] D. Lastomo, M.R. Djalal, Widodo, I. Robandi. "Optimization of PID controller design for DC motor based on flower pollination algorithm," dalam *The 2015 International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM 2015)*, 2015.
- [12] M. A. A. Mutalib. "Speed Control of DC Motor Using PI Controller." Project Report Faculty of Electrical Engineering, Universiti Malaysia, Pahang, 2008.
- [13] R. G. Kanojiya dan P. Meshram. "Optimal tuning of PI controller for speed control of DC motor drive using particle swarm optimization," dalam *International Conference on Advances in Power Conversion and Energy Technologies (APCET)*, 2012, hlm. 1-6.

- [14] K. Sundareswaran dan M. Vasu, "Genetic tuning of PI controller for speed control of DC motor drive," dalam *Proceedings of IEEE International Conference on Industrial Technology*, 2000, hlm. 521-525.