Implementasi Metode PSO-LDW untuk Optimasi Kontroler PID Pada Plant Orde Tinggi

Alrijadjis[1)]
[1)]Politeknik Elektronika Negeri Surabaya – ITS
Kampus ITS, Sukolilo, Surabaya 60111, email: alrijadjis@eepis-its.edu

Abstrak

Kontroler PID adalah kontroler yang paling populer abad ini karena efektifitasnya luar biasa, implementasinya sederhana dan aplikasinya sangat luas. Namun, dalam prakteknya tuning kontroler PID merupakan suatu persoalan tersendiri yang tidak mudah dan umumnya masih dilakukan secara manual yang memerlukan cukup waktu, terlebih untuk sistem atau plant orde tinggi. Makalah ini membahas teknik tuning kontroler PID menggunakan metode PSO-LDW (Particle Swarm Optimization-Linear Decreased Weight), yaitu metode PSO yang dimodifikasi (Modified PSO) , dimana parameter bobot inersia dibuat menurun secara linear. Metode PSO-LDW memiliki beberapa keunggulan dibandingkan metode PSO standar, yaitu konvergensinya lebih cepat dan total performansi indeksnya lebih baik. Sistem atau plant orde tinggi dimodelkan dalam Simulink dan algoritma PSO-LDW diimplemenntasikan dalam MATLAB. Hasil simulasi menunjukkan bahwa metode PSO-LDW menghasilkan performansi sistem yang lebih baik dibandingkan dengan metode PSO standar maupun metode sebelumnya..

Kata Kunci*: PSO, PSO-LDW, sistem orde tinggi, kontroler PID, kontroler modern, komputasi cerdas*

1. PENDAHULUAN

Kontroler PID termasuk kontroler konvensional atau klasik yang masih populer di industri hingga saat ini karena efektifitasnya luar biasa, implementasinya mudah dan aplikasinya sangat luas [1].

Seiring dengan perkembangan teknologi, mulai dikembangkan kontroler modern, seperti *nonlinear control*, *optimal control*, *variable structure control*, *adaptive control*, *neural-network* (NN), *fuzzy logic controller* (FLC), dan sebagainya. Tetapi, kontroler modern ini masih kalah popular dengan PID karena berbasis teori yang kompleks, sulit implementasinya, sulit dijadikan *general purpose*, *user friendly*, dan *smart controller* [2].

Masalah utama kontroler PID adalah *tuning*, yaitu menentukan nilai parameter Kp, Ki dan Kd agar diperoleh performansi sistem yang optimal. Umumnya proses *tuning* masih dilakukan secara manual, *trial and error* sehingga memakan waktu.

Perkembangan proses *tuning* PID diawali dengan metode *tuning* konvensional, seperti Ziegler-Nichols, Cohen-Coon, Astrom-Hagglund, Poulin-Pomerleau, dan sebagainya. Tetapi, metode-metode ini kurang optimal karena menggunakan asumsi sistem memiliki dinamika minimum, *linear* (LTI) dan *no-disturbance*. Pada kenyataannya proses industri adalah *nonlinear*, *time-varying* dan kompleks.

Salah satu kontroler modern, yaitu FLC yang juga mulai banyak diimplementasikan, mempunyai potensi besar dalam memecahkan persoalan kontrol yang rumit. Tetapi, FLC memiliki keterbatasan, yaitu : desain rumit karena berbasis pengalaman operator, sulit menentukan kestabilan sistem karena berbasis aturan (*rule-based*), aturan yang salah atau *out-dated* dapat mengakibatkan ketidakstabilan, jika ada perubahan sistem atau *plant* maka perlu modifikasi aturan dan umumnya memakan waktu [2].

Selanjutnya, Yu mengembangkan metode LQR (*Linear Quadratic Regulator*) untuk *tuning* PID dengan hasil yang cukup optimal, tetapi perlu kalkulasi matematika dan penyelesaian yang rumit [3]. Perkembangan teknologi komputasi melahirkan metode komputasi cerdas [*genetic algorithm* (GA), *ant colony optimization* (ACO), *particle swarm optimization* (PSO), *simulated annealing* (SA), *bacterial foraging optimization* (BFO)] yang membuka jalan baru bagi metode *tuning* modern. Metode *tuning* modern ini dirancang untuk sistem yang kompleks, *nonlinear*, *time-varying* seperti yang banyak ditemui di proses industri [4], [5].

Lin memperkenalkan kontroler PID berbasis GA untuk BLDC motor [8]. GA adalah algoritma optimasi stokastik yang diinspirasi oleh mekanisme seleksi alam dan evolusi genetika. Dengan metode GA dapat menyelesaikan persoalan optimasi yang kompleks, tetapi pada penelitian akhir-akhir ini terjadi penurunan performansi GA untuk kasus dimana parameter-parameter yang akan dioptimasi memiliki korelasi yang kuat. Sehingga operasi *crossover* dan mutasi tidak mampu menghitung *fitness* karena stuktur kromosom yang hampir sama [9], [10], [11].

PSO mulai diperkenalkan oleh Dr. Kennedy dan Dr. Eberhart pada tahun 1995 dan merupakan salah satu algoritma *heuristic modern* yang diinspirasi oleh perilaku organisme, seperti kawanan burung atau ikan. Secara umum PSO memiliki karakteristik : konsepnya sederhana, mudah implementasinya dan efisien dalam komputasi. PSO merupakan metode berbasis populasi seperti GA, tetapi konsep dasarnya adalah kerjasama, bukan persaingan [6], [7].

Pillay menerapkan metode PSO standar untuk optimasi kontroler PID dengan plant orde tinggi dan hasilnya lebih baik dibandingkan dengan metode lainnya [14].

Pada makalah ini membahas *tuning* kontroler PID berbasis PSO-LDW (yaitu pengembangan dari metode PSO standar) pada sistem atau *plant* orde tinggi. Hasilnya akan dibandingkan dengan beberapa metode sebelumnya.

2. PARTICLE SWARM OPTIMIZATION (PSO)

PSO adalah salah satu teknik optimasi dan termasuk jenis teknik komputasi evolusi. Metode ini memiliki *robust* yang bagus untuk memecahkan persoalan yang mempunyai karakteristik *nonlinear* dan *nondifferentiability, multiple optima*, dimensi besar melalui adaptasi yang diturunkan dari teori psychologi-sosial [6],[11]. Metode ini diinspirasi oleh dinamika gerak kawanan burung atau ikan dalam mencari makanan. Mereka bergerak secara bersamaan dalam suatu kelompok dan bukan tiap individu. Mereka menggunakan konsep kerjasama dimana tiap informasi di-*sharing* dalam kelompok. Konsep PSO sangat berbeda dengan GA. Konsep GA adalah persaingan berdasarkan seleksi alam, sehingga menggunakan operator evolusi seperti mutasi dan *crossover*.

Dalam prosesnya metode PSO dipengaruhi oleh sifat individu dan kelompok dalam mendapatkan solusi optimal. Sebagai pengganti operator evolusi, misalnya untuk persoalan optimasi d-variabel, akan disebar kawanan partikel (misalnya sebanyak n-partikel) dalam ruang d-dimensi secara acak. Masing-masing partikel merupakan kandidat solusi dan mempunyai nilai *fitness* tertentu. Kemudian tiap partikel akan bergerak dengan kecepatan tertentu yang dipengaruhi oleh pengalaman terbang sendiri dan pengalaman terbang partikel lainnya. Sebagai contoh, partikel ke-i dinyatakan sebagai $x_i = \left(x_{i,1}, x_{i,2}, ... x_{i,d}\right)$ dalam ruang d-dimensi. Posisi terbaik sebelumnya dari partikel ke-i disimpan dan dinyatakan sebagai : $pbest_i = \left(pbest_{i,1}, pbest_{i,2}, ..., pbest_{i,d}\right)$.

Indeks partikel terbaik dinatara semua partikel dalam kawanan atau *group* dinyatakan sebagai $gbest_d$. Kecepatan patikel ke-i dinyatakan sebagai : $v_i = \left(v_{i,1}, v_{i,2}, ...v_{i,d}\right)$. Modifikasi kecepatan dan posisi tiap partikel dapat dihitung menggunakan kecepatan saat ini dan jarak dari $pbest_{i,d}$ ke $gbest_d$ seperti ditunjukkan persamaan berikut ini :

$$v_{i,m}^{(t+1)} = w.v_{i,m}^{(t)} + c_1 * R * \left(pbest_{i,m} - x_{i,m}^{(t)}\right) + c_2 * R * \left(gbest_m - x_{i,m}^{(t)}\right) \quad (1)$$

$$x_{i,m}^{(t+1)} = x_{i,m}^{(t)} + v_{i,m}^{(t+1)} \quad (2)$$

$$i = 1,2,...,n$$

$$m = 1,2,...,d$$

di mana :

- n : jumlah partikel dalam kelompok
- d : dimensi
- t : penunjuk iterasi (generasi)
- $v_{i,m}^{(t)}$: kecepatan partikel ke-i pada iterasi ke t
- w : faktor bobot inersia
- c_1, c_2 : konstanta akeselerasi (*learning rate*)
- R : bilangan random (0 – 1)
- $x_{i,d}^{(t)}$: posisi saat ini dari partikel ke-i pada iterasi ke-t
- $pbest_i$: posisi terbaik sebelumnya dari partikel ke-i
- gbest : partikel terbaik diantara semua partikel dalam satu kelompok/populasi

3. METODE PSO-LDW

Pada metode PSO standar nilai bobot inersia dibuat konstan, sehingga untuk beberapa kasus metode PSO standar menjadi kurang efisien. Shi dan Eberhart melakukan modifikasi terhadap nilai bobot inersia dengan pertimbangan pada saat awal iterasi bobot inersia di-set dengan nilai yang cukup besar untuk memperluas daerah pencarian dan menghindari terjebak di local optimum, kemudian pada iterasi terakhir bobot inersia di-set cukup kecil untuk mendapatkan hasil akhir yang akurat [15].

Pada makalah ini, seting untuk bobot inersia dinyatakan sebagai berikut :

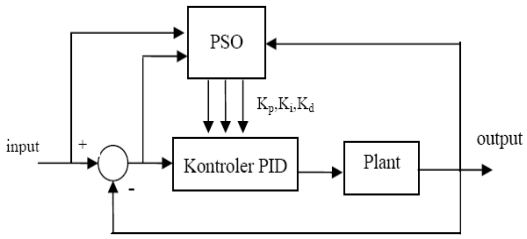
$$w = w_2 - \frac{w_1}{iter_{\max}} \cdot iter \quad (3)$$

di mana :

- w_2 : nilai bobot awal
- w_1 : nilai bobot akhir
- iter : iterasi yang berjalan
- $iter_{max}$: iterasi maksimum

4. DESAIN KONTROLER MPSO-PID

Kontroler MPSO-PID untuk *plant* ditunjukkan pada gambar 1.



Gambar 1 *Struktur kontrolerPSO-PID*

Di mana indeks performansi IAE yang dipakai untuk mengestimasi parameter-parameter PID diberikan sebagai berikut:

$$IAE = \int_0^T |e(t)| dt \qquad (4)$$

Konsep utama *tuning* kontroler PID secara *on line* adalah *tuning* parameter PD tiap *sampling* waktu. Fungsi obyektif atau *fitness function* yang akan dioptimasi dinyatakan sebagai berikut :

$$J(i) = \alpha.IAE(i) + \beta.|O(i)| \qquad (5)$$

di mana :

α, β : faktor *improvement*

O : *overshoot*

Flowcart kontroler PSO-PID ditunjukkan pada gambar 2.

5. ANALISIS NUMERIK

Sebagai contoh representasi sistem orde tinggi adalah [14]:

$$G(s) = \frac{s^2 - 30s + 300}{s^4 + 31s^3 + 330s^2 + 300s} \qquad (6)$$

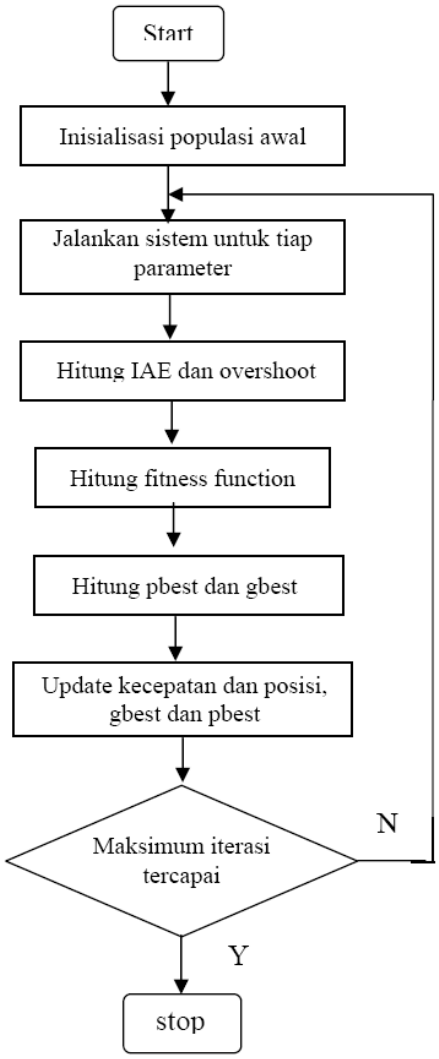
Parameter-parameter algoritma PSO yang digunakan sebagai berikut:

1. Jumlah populasi atau partikel, n = 20
2. Jumlah iterasi, iter = 20
3. Bobot inersia, w = 0,5
4. Konstanta akselerasi *cognitive*, c_1 = 1,5
5. Konstanta akselesasi sosial, c_2 = 1,5

Parameter-parameter algoritma PSO-LDW yang digunakan sebagai berikut:

1. Jumlah populasi atau partikel, n = 20
2. Jumlah iterasi, $iter_{max}$ = 20
3. w_2 = 0,65
4. w_1 = 0,35
5. Konstanta akselerasi cognitive, c_1 = 1,5
6. Konstanta akselerasi sosial, c_2 = 1,5

Hasil *tuning* dengan metode PSO-LDW dan perbandingannya dengan metode lainnya (Ziegler-Nichols atau ZN, Genetic Algorithm atau GA, Ant Colony Optimization atau ACO, Multi Objective ACO dan PSO) ditunjukkan pada gambar 3. Kontroler PID yang di-*tuning* dengan metode PSO-LDW menghasilkan sistem dengan karakteristik umum yang lebih baik daripada metode *tuning* lainnya. Khususnya pada *overshoot* yang dapat ditekan secara signifikan. Perbandingan performansi dinamik sistem dengan berbagai metode *tuning* ditunjukkan pada tabel 1.



Gambar 2 *Algoritma PSO*

Dari tabel 1, dapat diketahui performansi umum sistem dengan persamaan beikut (E.A. Gargari, dkk, 2007) :

$$f_{total}(\min) = f_1 + f_2 + f_3 + f_4 + f_5 \qquad (7)$$

di mana :

f_1 : *maximum overshoot* (Mp)

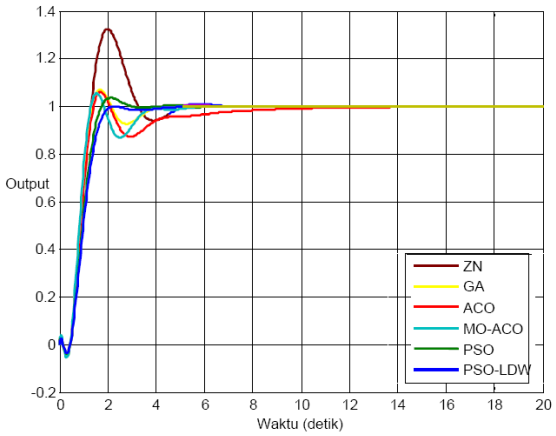
f_2 : *rise-time* (tr)

f_3 : *settling-time* (ts)
f_4 : IAE
f_5 : ISE

Dari persamaan (7), nilai f_{total} dari metode PSO-LDW menunjukkan nilai paling kecil, hal ini berarti performansinya paling baik dibandingkan metode *tuning* lainnya.

Tabel 1 Perbandingan metode tuning

ITEM		METODE TUNING					
		ZN	GA	ACO	MO-ACO	PSO	PSO-LDW
Parameter PID	Kp	2,808	2,391	2,4911	2,327	2,07	1,7139
	Ki	1,712	1,072	0,8158	1,021	1,0402	0,8365
	Kd	1,151	1,458	1,3540	1,668	1,1178	0,8553
Spesifikasi Performansi Dinamik	Final value	1	1	1	1	1	1
	Ess	0	0	0	0	0	0
	Mp(%) [f1]	32,45	7,09	6,08	5,45	3,59	0
	tr (s) [f2]	0,6376	0,660	0,6864	0,6083	0,8722	1,2599
	ts (s) [f3]	4,734	3,65	7,3	3,55	2,55	2,389
Indeks Performansi	IAE [f4]	1,361	1,005	1,279	1,033	1,035	1,2039
	ISE [f5]	0,8725	0,7564	0,7864	0,7416	0,8179	0,9183
	[f total]	40.055	13,161	16,132	11,383	8,865	5,771



Gambar 3 Kurva perbandingan respon waktu

Dari persamaan (7), nilai f_{total} dari metode PSO-LDW menunjukkan nilai paling kecil, hal ini berarti performansinya paling baik dibandingkan metode *tuning* lainnya.

6. KESIMPULAN

Dalam makalah ini telah membahas metode *tuning* kontroler PID dengan metode PSO-LDW. Berdasarkan analisis numerik, *tuning* kontroler PID menggunakan metode PSO-LDW dapat menghasilkan sistem dengan performansi yang lebih baik (khususnya *overshoot*) dibandingkan dengan metode PSO standar dan metode-metode sebelumnya.

DAFTAR PUSTAKA

[1] K. Ang, G. Chong, and Y. Li,"PID control system analysis, design and technology,"*IEEE Trans. Control System Technology*, vol. 13, pp.559-576, July 2005

[2] M. El Said El telbany,"Employing particle swarm optimizer and genetic algorithm for optimal tuning of PID controller: A comparative study," *ICGST-ACSE Journal*, vol. 7, Issue 2, Nov. 2007.

[3] G. Yu, and R. Hwang,"Optimal PID speed control of BLDC motor using LQR approach," *Proc. IEEE Int. Conf.System, Man and Cybernetics*, 2004, pp.473-478.

[4] Goldberg D., *Genetic Algorithms in Search, optimization, and Machine Learning*. Addison-Wesley, 1989.

[5] Michalewicz Z. and Dasgupta D., *Evolutionary Algorithms in Engineering Applications*. Springer-Verlag, 1997.

[6] Kennedy, J. and Eberhart, C., "Particle Swarm Optimization," *Proceedings of the 1995 IEEE International Conference on Neural Networks*, Australia, pp. 1942-1948, 1995.

[7] Kennedy, J. and Eberhart, C., *Swarm Intelligence*, Morgan Kaufman, 2001.

[8] C.L. Lin and H.Y. Jan and N.C. Shieh,"GA-based multiobjective PID control for a BLDC motor," *IEEE/ASME Trans. Mechatronics*, vol.8, No.1,pp. 56-65, 2003.

[9] D. B. Fogel, *Evolutionary Computation toward a New Philosophy of Machine Intelligence*: New York: IEEE, 1995.

[10] R. C. Eberhart and Y. Shi, "Comparison between genetic algorithms and particle swarm optimization," in *Proc. IEEE Int. Conf. Evol. Comput.*, Anchorage, AK, May 1998, pp. 611–616.

[11] Z.-L. Gaing, "A particle swarm optimization approach for optimum design of PID controller in AVR system," *IEEE Trans. Energy Conversion*, vol. 19, pp. 384-391, June 2004.

[12] Magnus Mossberg, Michel Gevers, Oliver Lequin,"A Comparison of Iterative Feedback Tuning and Classical PID Tuning Schemes", *Elsevier*, Novemeber 2002

[13] E.A Gargari, C. Lucas,"Design an Optimal PID Controller using Colonial Competitive Algorithm", IEEE Congress on Evolutionary Computation, Singapore, 2007

[14] Nalendran Pillay,"A Particle Swarm Optimization Approach for Tuning of SISO PID Controller", Master Thesis Durban University of Tehcnology, 2008

[15] RC Eberhart, Y. Shi,"Comparing Inertia Weights and Constriction Factors in PSO", IEEE, 2000