Application of Intelligent Search Algorithms in Proportional-Integral-Derivative Control of Direct-Current Motor System

Alfian Ma'arif\textsuperscript{1*}, Hanamirza Nabila\textsuperscript{2}, Iswanto\textsuperscript{3}, Oyas Wahyunggoro\textsuperscript{4}

\textsuperscript{1}Department of Electrical Engineering, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
\textsuperscript{2}Department of Electrical Engineering, Universitas Islam Indonesia, Yogyakarta, Indonesia
\textsuperscript{3}Department of Electrical Engineering, Universitas Muhammadiyah Yogyakarta, Yogyakarta, Indonesia
\textsuperscript{4}Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, Yogyakarta, Indonesia

*Email: alfianmaarif@ee.uad.ac.id

Abstract. Proportional-Integral-Derivative (PID) control is one of the famous controllers applied in industrial and robotics world. It is quite easy to understand and to be implemented. It is known that, with a proper combination of its gain parameters, a system with good performance can be obtained. This parameter tuning can be tricky since the number of the combination is almost infinite. Hence, methods to tune the parameter gains of proportional, integral and derivative is needed. Some optimization methods that can be used in tuning parameters are Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Differential Evolution (DE). The paper will analyse the performance of those methods by applying them in PID Controller to control a Direct Current (DC) motor system by comparing the iteration time, the number of parameters, and augmented system performance. The comparison is done on MATLAB simulation by using the same computer. It results as DE as the best among them with the least number of parameters and best DC motor system performances.

1. Introduction

PID Controller was first published officially by Minorsky in the early 19th century with his theory of three-term control. It is the most used controller in industrial and robotics world until now [1] [2]. Some advantages of PID Controller are easy to understand, easy to be implemented, and able to provide a good system's stability [3]. Many systems are able to be stabilized, from non-linear to linear system.

Nevertheless, it is not problem-free. Two major problems in PID Controller are its inability to control high order system and trivial task in tuning its parameter. Its ability to stabilize is limited to second order system or less. This can be solved by doing modification in its integrator part.
The tuning parameter of PID is important because it can guarantee the controller efficiency [4]. The most frequently used method in designing a PID controller's parameter is a trial and error method. It is practically done by adding random value in its parameter then tested on system or simulation. If the parameter value was 'right', it can provide good system's performance by chance. However, it is not the best result. Besides, it is ineffective and does not have a standard procedure.

There are many methods that are able to function as tune the parameter such as Ziegler Nichols [5], Coefficient Diagram Method (CDM) [6] [7], Fuzzy tuning [8], Neural network and Intelligent Search Algorithm [9]. Unfortunately, few methods above have limitations and weaknesses. Ziegler Nichols method is only suitable for first order system. CDM has inflexibility issue in its system model representation preference, which it has to be in state space. Meanwhile, Fuzzy is quite difficult to be designed, especially for a system with no previous experimental data.

Intelligent Search Algorithm, or Heuristic method, is another method to tune parameter by doing a repeated iteration to obtain the best parameter value. It copies natural phenomenon so that it has a structured procedure to solve complex problems which cannot be solved by an analytical method. Its weakness is it has a great number of iterations which requires longer iteration time. Besides, the obtained result somehow not always consistent if the searching process is retaken.

Due to the many innovations in technology nowadays, those weaknesses can be overcome. High-speed data process in modern computers helps to decrease the number of iterations and computing time. Hence, Intelligent Search Algorithm is the most potent method to optimize parameter tuning in PID Controller.

The algorithm itself has many variations due to which natural phenomenon in the world. Some of them are Genetic Algorithm (GA) [10] [11], Particle Swarm Optimization (PSO) [12], and Differential Evolution (DE) [13] [14]. They are the most commonly used algorithm in Intelligent Search Algorithm. To test the algorithm, it is applied in the PID Controller of DC Motor System. The performance of the system along with a number of iterations and iteration time will be observed and analysed. DC Motor is chosen due to its commonly used in industrial [13] and robotics world.

The structure of this paper is written as follows. The first section is started with modelling of DC motor, and a brief explanation about PID controller. The next section discusses the Genetic Algorithm, Particle Swarm Optimization, Differential Evolution and its objective functions. Result and discussion will be described next and it will be closed with conclusions.

2. Model of Direct Current (DC) Motor

The model diagram of DC motor system that is used in the research is shown in Figure 1. Based on Figure 1, $V_{DC}$ is the power source (Volt), $i$ is current (A), $R$ is resistance (Ω), $L$ is inductance (H), $f_k$ is the friction force, $J$ is inertia moment, $T$ is torque, and $\omega$ is angular speed (rad/s).
Model of DC motor system is derived from the electrical and mechanical point of view analysis. The equation from the electrical point of view is based on the Kirchhoff voltage law that the total voltage in the closed circuit is zero, which is written as:

\[ V_L + V_R + V_{Emf} = v(t) \]  

\[ L \frac{di(t)}{dt} + Ri(t) + K_e \omega(t) = v(t) \]  

where \( K_e \) is Back-Emf (Electro-mechanical force) constant (Vs/rad). Then, the equation (2) is transformed with Laplace Transformation so that it becomes

\[ I(s)(Ls + R) = V(s) - K_e \omega(s) \]

Meanwhile, when the system is seen from a mechanical point of view, the Newton second law of rotation can be applied

\[ T - f_k = J \dot{\omega}(t) \]

\[ K_t i(t) - K_b \omega(t) = J \frac{\omega(t)}{dt} \]

where \( K_t \) is torque constant, \( K_b \) is friction constant, \( \dot{\omega} \) is angular acceleration. The equation (4) is also transformed into Laplace as stated,

\[ J s \omega(s) + K_b \omega(s) = I(s) K_t \]

\[ I(s) = (J s + b) \frac{\omega}{K_t} \]

By substituting (3) to (5), the transfer function of the DC motor system can be obtained as

\[ \text{Figure 1. Model Diagram of DC Motor System} \]
\[ \omega(s) = \frac{K_t}{(Ls + R)(Js + K_b) + K_tK_e} \] (6)

3. Proportional-Integral-Derivative Controller

PID Control [15] equation in time domain is

\[ u = K_p e(t) + K_i \int_{t=0}^{t} e(t) \, dt + K_d \frac{e(t)}{dt}, \] (7)

where \( e \) is the error between the set point and feedback value, \( u \) is the control signal. The \( K_p \) is the proportional gain, \( K_i \) is the integral gain, and \( K_d \) is the derivative gain. The \( K_p, K_i \) and \( K_d \) later will be tuned by using the optimization method.

The block diagram of the proposed system shown in Figure 2.

![Diagram Block of Proposed Controller](image)

**Figure 2.** The Diagram Block of Proposed Controller

Based on Figure 2, \( r \) is the set point or speed reference, \( y_m \) is the feedback value (speed value from the sensor). The value of \( K_p, K_i \), and \( K_d \) are determined as the output from the tuning process which uses intelligent search algorithms (GA, PSO, DE).

4. Objective Function

In every optimization process, the optimization problem needs to be identified first in the objective function. It represents problems that will be solved that guides the searching process. It can be the minimum function or maximum function based on conditions.

In this design, the function itself also represents the fitness value. The control objective of the controller is to achieve minimum steady state error, minimum rise time, minimum settling time, and also minimum overshoot. The objective function of this design can be stated as follows,

\[ \text{Min}(f) = \left( w_1 \ast sse \right) + \left( w_2 \ast t_s \right) + \left( w_3 \ast t_r \right) + \left( w_4 \ast O_s \right), \] (8)

where, \( w_i \) is weight, \( sse \) is steady state error, \( t_r \) is rise time, \( t_s \) is settling time and \( O_s \) is overshoot.
When compared in the iteration of the optimization process, fitness value should be compared in maximum function. Therefore, it is logical that the solution with bigger fitness value is a better solution. In order to make this happen, the obtained fitness value is converted to maximum function by applying

$$Max(f) = \frac{1}{Min(f)}.$$  \hspace{1cm} (9)

5. Genetic Algorithm (GA)

GA was firstly declared by Holland in 1962. It copies the natural selection process in population which is important to genetic evolution. Some important terms used in GA are selection, crossover, mutation, and elitism. Selection is a process to pick individuals which will be used as parents in the reproduction process. Crossover is a process which chromosomes of individual parents are exchanged with each other to inherit traits. The mutation is a process where traits are changed. Elitism is a replacement of individual with less-quality to the new one in a population.

Procedures of the genetic algorithm can be explained briefly as follows,

1. Initialize population and decode an individual;
2. Evaluate the fitness value of the individual. It is normally formulated by its objective function;
3. Do selection to determine new parents in the reproduction process;
4. Do reproduction with selected parents to create a new individual. This step requires two operators: crossover and mutation;
5. Evaluate the fitness value of the new individual in the population. If it is better than another individual, replace the old individual with this new one (do elitism).

There are many kinds of selection operators. There are roulette wheel selection, tournament selection, and rank-based selection. The most popular among them all is the roulette-wheel selection. In this kind of selection, a random number $r$ is generated and the chosen parents are individuals which have closest relative fitness value to $r$.

Two most popular crossover operators are a 1-point crossover and $n$-point crossover. 1-point crossover is the simplest where a point in chromosome will separate parent's chromosomes into two segments and the new individual will be made by exchanging them. It can be improved by adding more points which are then called $n$-point crossover.

6. Particle Swarm Optimization (PSO)

PSO was first introduced by Kennedy and Eberhart in 1995 which made based on social and communication behaviour of bird’s flock. In PSO, a large number of particles move spread in multidimensional problem space. Every individual is assumed as a vector position and represents a potential solution of optimization problems.
Unlike other intelligent search algorithms which use evolution operator to manipulate individuals, PSO uses dynamic velocity which may change based on the individual’s experience during moving in the swarm. Therefore, every individual is pulled to average stochastic weight from a previous best local point near its flock.

Procedures of Particle Swarm Optimization (PSO) can be explained briefly as follows,

1. Initialize particles and generate velocity randomly;
2. Evaluate particles by comparing their fitness values. This will be the benchmark to determine the best local point ($P_{best}$) and best global point ($G_{best}$);
3. Compare $P_{best}$ of before and after iteration. If the new fitness value of a particle is better than previous $P_{best}$ then it will be the new $P_{best}$;
4. Get $G_{best}$ by finding the highest fitness value of $P_{best}$;
5. Update the velocity and position of the particle.

\[
V_{i}^{kg+1} = w(t) + V_{i}^{kg} + c_1 r_1 (P_{i}^{kg} - X_{i}^{kg}) + c_2 r_2 (Best_{i}^{kg} - X_{i}^{kg}) \quad (10)
\]

\[
X_{i}^{kg+1} = X_{i}^{kg} + V_{i}^{kg} \quad (11)
\]

where $g = 1, 2, ..., G$, $i = 1, 2, ..., Size$, $r_1$ and $r_2$ is random values between 0 – 1, $c_1$ is a local learning factor, $c_2$ is a global learning factor. Commonly, $c_2$ is bigger than $c_1$, or can be written as $c_2 > c_1 > 0$.

7. Differential Evolution (DE)

DE was first proposed by Storn and Price in 1995. Although it may seem that DE was created based on natural phenomenon, actually it was rather based on geometry argument. It has mutation and crossover operator in the searching process like in GA yet the new solution was created by selecting three vectors as 'parents' or strategic vectors. One of them is used as a base vector while two others are compared to find their vector difference to be added to the base vector.

This competition strategy, together with encoding and simple mutation differential, the complexity of the DE algorithm can be reduced. Main advantages of DE can be summarized in three points: less parameter, not easily trapped in local solution, and fast to converge.

Standard DE Algorithm needs at least 4 steps,

1. Initialize population generation,

\[
P_{ij} = rand_{p1j} (0, 1)(x_{ij}^{U} + x_{ij}^{L}) + x_{ij}^{L} \quad (12)
\]

where $i$ is the number of individuals (population), $j$ is the number of variables, $rand_{p1j} (0, 1)$ is the random value from 0 to 1, $x_{ij}^{U}$ is the upper limit, $x_{ij}^{L}$ is the lower limit;

2. Do mutation by selecting three random individuals in the population,
\[ h_{ij}(t + 1) = x_{p1j}(t) + F\left(x_{p2j}(t) - x_{p3j}(t)\right), \]  
(13)

while the function for no local solution is,

\[ h_{ij}(t + 1) = x_{bj}(t) + F\left(x_{p2j}(t) - x_{p3j}(t)\right), \]  
(14)

where, \( x_{bj} \) is the best individuals. \( F \) is the mutation factor that selected between \( 0 - 2 \). Generally, it can be set from 0.3 to 0.6. It will make fast convergence and the process may not trap in the local solution;

3. Do crossover to create a new vector,

\[ v_i(t + 1) = \begin{cases} 
  h_{ij}(t + 1), & \text{rand}_{ij} \leq CR \\
  p_{ij}(t), & \text{rand}_{ij} > CR 
\end{cases} \]  
(15)

where \( CR \) is the crossover factor in the range \([0,1]\). Generally, it can be set from 0.6 to 0.9;

4. Evaluate vector by comparing it with its parents. If it is better than its parents, then do the replacement,

\[ R_i(t + 1) = \begin{cases} 
  v_i(t + 1), & \text{f}(v_{ij}(t + 1), ..., v_{in}(t + 1)) < \text{f}(x_{ij}(t), ..., x_{in}(t)) \\
  x_{ij}, & \text{f}(v_{ij}(t + 1), ..., v_{in}(t + 1)) \geq \text{f}(x_{ij}(t), ..., x_{in}(t)) 
\end{cases} \]  
(16)

8. Numerical Simulation and Discussion

The method will be simulated in Simulink Matlab. The chosen simulation stop time is 5 seconds. The following are specifications of the computer used to do the computation: Windows 10 Home Basic, i5 8th generation core and a solid-state drive. The transfer function model of DC motor system is,

\[ \omega = \frac{0.01}{0.005s^2 + 0.06s + 0.001} V, \]  
(17)

where \( \omega \) is the angular velocity which is stated in rad/s (radian per second). In order to convert it to rotation per minute (rpm), the obtained result should be multiplied with \( 60/(2 \times \pi) \) or \( 30/\pi \).

In every test taken, data are compared by some criteria. Those are the number of parameter variables, number of generation, number of iteration, iteration time, the obtained result (\( K_p, K_i, K_d \)), the system’s steady state error (\( sse \)), system’s rise time (\( t_r \)), system’s settling time \( t_s \), and the system’s overshoot (\( O_s \)). The result of the experiment can be shown on Figure 3, Table 1, and Table 2.
Figure 3. Best Result in First, Second, and Third Experiment

Table 1. Compared Parameters

| Variables | Population | Iteration | Time (s) |
|-----------|------------|-----------|----------|
| PSO       | 11         | 10        | 10       | 48.854   |
| GA        | 9          | 10        | 10       | 10.753   |
| DE        | 7          | 10        | 10       | 33.998   |

Table 2. Parameters of PID Control and Performances System

| Method | $K_p$ | $K_i$  | $K_d$    | sse     | $t_r$  | $t_s$  | $O_s$  |
|--------|-------|--------|----------|---------|--------|--------|--------|
| PSO    | 20    | 12.6999| 0.22419  | -0.0005334 | 0.0252 | 0.1178 | 17.9839|
| GA     | 18.1836| 1.0156 | 1.5625   | 0.0571   | 0.1937 | 0.3516 | 0     |
| DE     | 20    | 1.702  | 0.906    | 0.0092   | 0.1108 | 0.2040 | 0     |
Overall, GA and DE often give good system's performance. PSO tends to give poor result such as oscillated or overshoot response. PSO's best result is obtained through many experiments. As seen in Figure 3, PSO gives overshoot response in both, first and second experiment. However, in the third experiment, PSO gives a response with no overshoot but slower response than other algorithms. GA and DE are more excellent than PSO in terms of providing good system performance.

Speaking of iteration time in Table 1, PSO needs longer time to find solutions rather than DE and GA with the same number of generation. GA takes the shortest iteration time while DE is less fast (slower) than GA but quite faster than PSO. In term of the number of variables which must be tuned, DE has the least variables among them all. Otherwise, PSO has the most variables.

More details of system performance can be seen in Table 2. Neglecting the occurrence of overshoot, system with best steady state error is achieved through PSO tuning process in first experiment and second experiment. However, best steady state error performance with no overshoot in response, achieved by DE in the first experiment with $sse = 0.0092$.

Aggressively of the system can be observed from the rise time performances. More aggressive system commonly has faster or less rise time performances. Overall, PSO provides the fastest rise time with 0.0252 seconds in the first experiment. However, DE gives fastest rise time among all algorithms in the second experiment and third experiment. Thus, indicates that DE tends to give best aggressively of the system.

In the terms of fast or slow response, all algorithms tend to give fast response which can be seen in the obtained rise time and settling time which is almost under 1 seconds. Again, PSO provides the best settling time response in first experiment but DE is better in second and third experiment. Hence, in case of giving best settling time performance, DE is the best. Meanwhile, from the obtained $K_p$, $K_i$, and $K_d$, consistency of tuning process result can be observed.

As in every optimization problem, the objective function roles to guide the searching process. Therefore, the objective function needs to be formulated correctly so that the searching process can lead to the best solutions. Based on experimental data taken so far, the previous objective function can be modified to find the best $K_p$, $K_i$, and $K_d$. It would be better if some conditions are added. For example,

| Experiment | PSO   | GA    | DE    |
|------------|-------|-------|-------|
| PSO        | 19.6734 | 15.625 | 20.00 |
| GA         | 0.59466 | 0.625  | 0.0235 |
| DE         | 0.000053151 | 0.0296 | 0.53017 |
|            | 0.0975 | 0.0869 | 0.0527 |
|            | 0.2848 | 0.1621 | 0.0970 |
|            | 7.7253 | 0.0000053151 | 0.0000053151 |

| Experiment | PSO   | GA    | DE    |
|------------|-------|-------|-------|
| PSO        | 17.5011 | 12.5  | 18.6362 |
| GA         | 7.3813 | 0.41016 | 1.6704 |
| DE         | 1.2912 | 1.1523 | 1.0618 |
|            | 0.1644 | 0.4554 | 0.0184 |
|            | 0.2069 | 0.2124 | 0.1271 |
|            | 0.4322 | 0.4043 | 0.2321 |
proportional values or $K_p$ should be greater than $K_i$ and $K_d$. Also, $K_i$ should be greater than $K_d$. Range values should be saturated to make a better result.

9. Conclusions

In the paper, parameters of PID Controller such as proportional, integral and derivative gains, have tuned by Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Differential Evolution (DE). Based on the simulation, Differential Evolution gives the best result with some advantages than the others. Differential Evolution provides the least number of parameter variables and fasts to converge. The best value of PID Controller for controlling the speed of DC motor is proportional gain $K_p = 20$, integral gain $K_i = 2.57$, and derivative gain $K_d = 0.0235$ with steady-state error $= 0.53017$, rise time $= 0.0527$, settling time $= 0.097$, overshoot $= 0$.

The future work of the research is as follow. The paper only compares three intelligent search method. Thus, it needs to compare with other methods such as bat algorithm. To make the result useful, ten methods are needed. The key to the optimization problem is in the objective function. The modification in the objective function will obtain the best and consistent solution.

10. References

[1] Gunawan S.A, Yuwono Y.C.H, Pratama G.N.P, Cahyadi A.I and Winduratna B. 2018 4th International Conference on Science and Technology (ICST) (Indonesia) 1–6.
[2] Sirisantisamrid K, Wongvanich N, Gulpanich S, and Tammarugwattana N, 2018 Proceedings of the International MultiConference of Engineers and Computer Scientists (Hong Kong) 14-16.
[3] El-Deen A.T, Mahmoud A.A.H, and El-Sawi A.R. 2015 Int. Rev. Autom. Control 8 81–82.
[4] Vishal V, Kumar V, Rana K.P.S and Mishra P. 2014 IEEE International Advance Computing Conference (IACC) 1342–1347.
[5] Ogata K 2010 Modern Control Engineering. (Boston: Prentice Hall).
[6] Ma’arif A, Cahyadi A. I., Wahyunggoro O, and Herianto. 2017 3rd International Conference on Science and Technology - Computer (ICST) 22–27.
[7] Ma’arif A, Cahyadi A.I, and Wahyunggoro O. 2018 Int. J. Adv. Sci. Eng. Inf. Technol., 8 930.
[8] Hasanjani R.A, Javadi S, and Nadooshan R.S. 2015 Trans. Inst. Meas. Control 37 164–176.
[9] Liu J. 2017 Intelligent Control Design and MATLAB Simulation. (Singapore: Springer) p. 33-56.
[10] Wati D.A.R and Hidayat R. 2013 International Conference on Robotics, Biomimetics, Intelligent Computational Systems (Indonesia) 30–34.
[11] Kumar P, Chatterjee S, Shah D, Saha U. K, and Chatterjee S. 2017 Cogent Engineering. 4 1-20.
[12] Wati D.A.R 2013 Proceeding of IEEE International Conference on Computational Intelligence and Cybernetics (CYBERNETICSCOM) (Indonesia) 40–44.
[13] Idir A, Kidouche M, Bensafia Y, Khettab K, and Tadjer S. A 2018 International Journal of Intelligent Engineering and Systems 20 241-249.
[14] Jigang H, Hui F, and Jie W 2019 Automatika 60 135–148.
[15] Franklin G.F, Powell J, and Emami A, 2015 Feedback Control of Dynamic Systems, Global Edition (New Jersey : Pearson Education Limited).