ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijareeie.com

Vol. 8, Issue 3, March 2019

Speed Control and Adaptive Tuning of DC Motor using P, PI, and Particle Swarm Optimization Techniques

Somnath Sharma[1] , Sumati Srivastava[2]

PG Student [Power System], Dept. of EEE, Maharishi University of Information Technology, Lucknow, U.P., India[1]

Asst. Professor, Dept. of EEE, Maharishi University of Information Technology, Lucknow, U.P., India[2]

ABSTRACT: The aim of this work is to design a speed controller of a DC motor by using P, PI, and bio-inspired optimization technique i.e. Particle Swarm Optimization (PSO).This paper implements a Particle Swarm Optimization (PSO) speed controller for controlling the speed of DC motor. Also traditional Proportional (P), Proportional-Integral (PI) controller has been developed and simulated using MATLAB/SIMULINK. The simulation results indicate that the PI controller has big overshoot, sensitivity to controller gains. The optimization with particle swarm verified the reduction of oscillations as well as improve the steady state error, rise time and overshot in speed response. . In this work bio-inspired optimization technique in controllers and their advantages over conventional methods is discussed. The simulation results confirmed the system is less sensitive to gain.

KEYWORDS: Bio-inspired optimization, Practical Swarm Optimization (PSO), Proportional (P), Proportional-Integral (PI) controller, DC motor.

I.INTRODUCTION

Highly efficient DC motor drives are commonly used for industrial applications due to their torque speed characteristics that make them suitable for a wide speed control range. A high starting torque with a constant torque region is obtained with a speed control after variation in the load. The torque and speed responses of a DC motor are suitable with various mechanical loads [1].

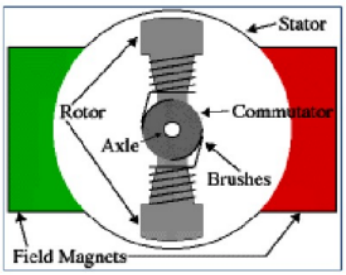

Figure 1 Block diagram of DC motor

Practically, the speed controller which implemented in a DC motor is simpler than the AC motor [2, 3]. The tuning of speed controller can be decided by a genetic algorithm (GA) [1]. As well as the gains of a PI controller is adjusted by optimization technique [3]. In several decades, the Proportional-Integral (PI) controllers have been used for dc speed control regarding the design simplicity and good performance. The system is considered the settling time, nonlinear modelling of dc motor which make the control robust [3].

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijareeie.com

Vol. 8, Issue 3, March 2019

This paper proposes the application of a particle swarm optimization technique for tuning parameters of a PI speed controller. The Particle Swarm Optimization (PSO) technique is recently applied in a few fields emerging because it's a powerful optimization tool. This paper is organized as follows: Section II is give illustrates of mathematical modelling of DC Motor with a classical PI controller. Section III gives particle swarm optimization. Section IV gives tuning of PI controller based on PSO. The simulation results are discussed in section V.

II. DC MOTOR DRIVE WITH PI CONTROLLER

PI controller block diagram is shown in Figure (2). In order to make zero steady state error for a step change of speed by adjusting the integral part of the controller.

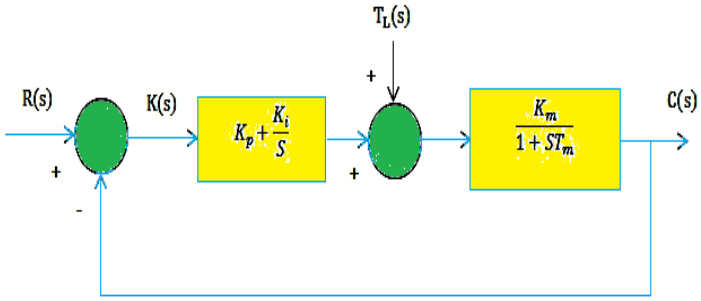

Figure 2. PI controller block diagram

Equation (1) gives transfer function of the control system.

$$\frac{C(S)}{R(S)} = \frac{K_m(SK_p+K_i)}{T_mS^2+(1+K_mK_p)S+K_i} \quad 1$$

Where
Ki is the integral gain of PI controller,
Kp, is the and the proportional gain of PI controller,
Tm, is the mechanical time constant of motor, and
Km, is the motor gain constant.
Whilst, Equation (2) gives the transfer function between output of the control system and the TL(S) is a load torque disturbance.

$$\frac{C(S)}{T_L(S)} = \frac{S(1+T_m)}{T_mS^2+(K_m+T_mK_i+K_p)S+K_i} \quad 2$$

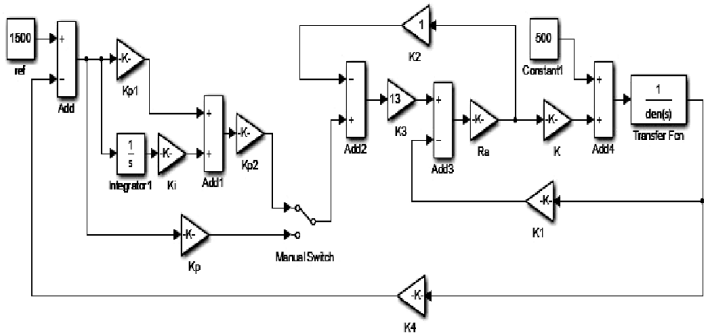

Figure 3. P, and PI controller

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijareeie.com

Vol. 8, Issue 3, March 2019

Where:
Ra = armature resistance (Ω).
La = self-inductance of armature (H).
J = moment of inertia (kg.m2).
B = friction coefficient (Nm*s/rad).

The DC motor parameters are given in Table I.

Specification of DC motor	Ra = 1.818Ω	La = 0.5H	J =0.0465 kg.m^2	B = 0.004 Nm*s/rad

Table I : The DC motor parameters

III. PARTICAL SWARM OPTIMIZATION (PSO)

Particle Swarm Optimization (PSO) is a new population-based evolutionary computation. Unlike genetic algorithms, the PSO updates populations without any genetic operators such as crossover and mutation. The PSO algorithm attempts to mimic the natural process of group communication of individual knowledge, which occurs when such swarms flock, migrate, forage, etc, in order to achieve some optimum property such as configuration or location. In PSO, the 'swarm' is initialized with a population of random solutions. Each particle in the swarm is a different possible set of the unknown parameters to be optimized. Representing a point in the solution space, each particle adjusts its flying toward a potential area according to its own flying experience and shares social information among particles. The goal is to efficiently search the solution space by swarming the particles toward the best fitting solution encountered in previous iterations with the intent of encountering better solutions through the course of the process and eventually converging on a single minimum error solution.

3.1. Standard PSO

Kennedy and Eberhart originated the original framework of PSO in 1995. In PSO, a swarm consists of N particles moving around in a D-dimensional search space. The random velocity assigned to each particle. Each particle modifies its flying based on its own and companion's experience at every iteration. The ith particle is denoted as Xi= (x_{i1}, x_{i2}, x_{i3}.....x_{iD}) whose best previous solution (pbest) is represented as Pi= (p_{i1}, p_{i2}, pi_3.....p_{iD}). Current velocity (position change rate) is described by Vi= (v_{i1}, v_{i2}, v_{i3}.....v_{iD}).Finally, the best solution achieved so far by the whole swarm (gbest) is represented as Pgi= (p_{g1}, $p_{g\,i2}$, $p_{g\,3}$..... p_{gD}). At each time step, each particle moves toward pbest and gbest locations. The fitness function evaluates the performance of particles to determine whether the best fitting solution is achieved. The particles are manipulated according to the following equations:

$$v_{id} = v_{id} + c_1 * rand * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \quad (3)$$

$$x_{id} = x_{id} + v_{id} \quad (4)$$

where c_1 and c_2 are two positive constants, called cognitive learning rate and social learning rate respectively; *rand*() is a random function in the range [4]. The velocity of the particles are limited in [V_{min}, V_{max}]. Since the original formula of PSO lacks velocity control mechanism, it has a poor ability to search at a fine grain [5]. A time decreasing inertia factor is designed by Eberhart and Shi to overcome this shortcoming in 1998 [6]:

$$v_{id} = w * v_{id} + c_1 * rand * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \quad (5)$$

$$x_{id} = x_{id} + v_{id} \quad (6)$$

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijareeie.com

Vol. 8, Issue 3, March 2019

where *w* is inertia factor which balances the global wide rang exploitation and the local nearby exploration abilities of the swarm.

3.2. Improved PSO

In the former researches [7], most of them widely investigated on the improvement of the velocity update equation and imposed the limit on the velocity of particles. Few were mentioned about the limit on the positions of particles in their studies. If there is no limit imposed on positions, it is possible for particles to fly out of defined search space, which sometimes leads to invalid solutions. To confine particles in defined search space, common methods implemented in the code of traditional PSO are to check the validity of the positions of particles and then take some measures to rectify invalid solutions at every iteration.

One rectification measure is to impose limit on positions, as it does on the velocity. If an element of the position is smaller then X_{min}, it is set equal to X_{min}; if greater than X_{max}, then equal to X_{max}.

Another is to reject the invalid particle, and then repeatedly evaluate the velocity update equation, formula (3) and formula (5), until the position updating equation, formula (3), obtaining a valid solution. Though those measures can restrict particles in defined search space, at the same time, they bring some excess computation. An improved PSO with momentum factor is introduced to solve this disadvantage [8]. The new technique can limit the particles in defined search space without checking the boundary at every iteration. In improved PSO, the particles are manipulated by the following equations:

$$v_{id} = w * v_{id} + c_1 * rand * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \quad (7)$$

$$x_{id} = (1 - mc) * x_{id} + mc * v_{id} \quad (8)$$

where *mc* is momentum factor ($0 < mc < 1$), and $V_{min} = X_{min}$; $V_{max} = X_{max}$.

IV. PARTICAL SWARM OPTIMIZATION (PSO) ALGORITHM

Particle Swarm Optimization might sound complicated, but it's really a very simple algorithm. Over a number of iterations, a group of variables have their values adjusted closer to the member whose value is closest to the target at any given moment. Imagine a flock of birds circling over an area where they can smell a hidden source of food. The one who is closest to the food chirps the loudest and the other birds swing around in his direction. If any of the other circling birds comes closer to the target than the first, it chirps louder and the others veer over toward him. This tightening pattern continues until one of the birds happens upon the food. It's an algorithm that's simple and easy to implement.

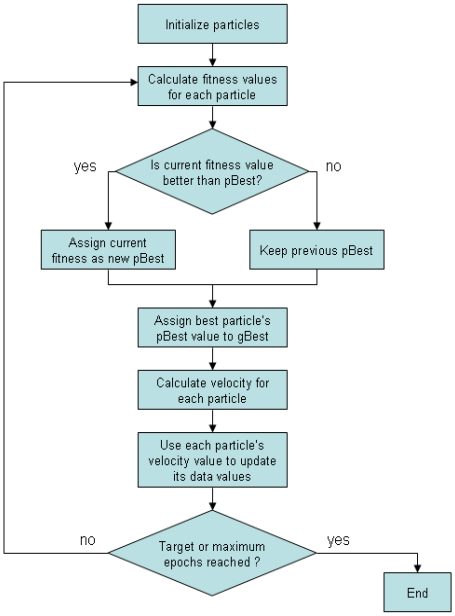

Figure 4. Flow diagram illustrating the particle swarm optimization algorithm.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijareeie.com

Vol. 8, Issue 3, March 2019

The algorithm keeps track of three global variables:

- **Target value or condition**
- **Global best (gBest) value indicating which particle's data is currently closest to the Target**
- **Stopping value indicating when the algorithm should stop if the Target isn't found**

Each particle consists of:

- **Data representing a possible solution**
- **A Velocity value indicating how much the Data can be changed**
- **A personal best (pBest) value indicating the closest the particle's Data has ever come to the Target**

The **particles' data** could be anything. If the data is a pattern or sequence, then individual pieces of the data would be manipulated until the pattern matches the target pattern. **The velocity value** is calculated according to how far an individual's data is from the target. The further it is, the larger the velocity value. If the data is a pattern or sequence, the velocity would describe how different the pattern is from the target, and thus, how much it needs to be changed to match the target. Each particle's **pBest value** only indicates the closest the data has ever come to the target since the algorithm started.The **gBest value** only changes when any particle's pBest value comes closer to the target than gBest. Through each iteration of the algorithm, gBest gradually moves closer and closer to the target until one of the particles reaches the target.

4.1 Pseudo Code of PSO algorithm

```
For each particle
{
   Initialize particle
}

Do until maximum iterations or minimum error criteria
{
   For each particle
   {
      Calculate Data fitness value
      If the fitness value is better than pBest
      {
         Set pBest = current fitness value
      }
      If pBest is better than gBest
      {
         Set gBest = pBest
      }
   }

   For each particle
   {
      Calculate particle Velocity
      Use gBest and Velocity to update particle Data
   }
```

Let S be the number of particles in the swarm, each having a position xi ∈ Rn in the search-space and a velocity vi ∈ Rn. Let pi be the best known position of particle i and let g be the best known position of the entire swarm. A basic PSO algorithm is then:[9]

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijareeie.com

Vol. 8, Issue 3, March 2019

for each particle $i = 1, ..., S$ **do**
  Initialize the particle's position with a uniformly distributed random vector: $\mathbf{x}_i \sim U(\mathbf{b}_{lo}, \mathbf{b}_{up})$
Initialize the particle's best known position to its initial position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
  if $f(\mathbf{p}_i) < f(\mathbf{g})$ **then**
    update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
  Initialize the particle's velocity: $\mathbf{v}_i \sim U(-|\mathbf{b}_{up}-\mathbf{b}_{lo}|, |\mathbf{b}_{up}-\mathbf{b}_{lo}|)$
while a termination criterion is not met **do**:
  for each particle $i = 1, ..., S$ **do**
    for each dimension $d = 1, ..., n$ **do**
      Pick random numbers: $r_p, r_g \sim U(0,1)$
      Update the particle's velocity: $\mathbf{v}_{i,d} \leftarrow \omega\, \mathbf{v}_{i,d} + \varphi_p\, r_p\, (\mathbf{p}_{i,d}-\mathbf{x}_{i,d}) + \varphi_g\, r_g\, (\mathbf{g}_d-\mathbf{x}_{i,d})$
    Update the particle's position: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
    if $f(\mathbf{x}_i) < f(\mathbf{p}_i)$ **then**
      Update the particle's best known position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
      if $f(\mathbf{p}_i) < f(\mathbf{g})$ **then**
        Update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$

The values $\mathbf{b}_{lo}$ and $\mathbf{b}_{up}$ are respectively the lower and upper boundaries of the search-space. The termination criterion can be the number of iterations performed, or a solution where the adequate objective function value is found [10]. The parameters ω, φ_p, and φ_g are selected by the practitioner and control the behaviour and efficacy of the PSO method

V. TUNING OF PI CONTROLLER BASED ON PSO

The values of the three parameters (Kp, Ki and Kd) must be adjusted. So that, the control input will provide possible accomplishment. These parameters have been included in a chromosome as illustrated in Figure (5). There are several controller design methods are implemented to get an acceptable results. The response with classical control methods needs retuning by the designer but these methods provide initial approximation.

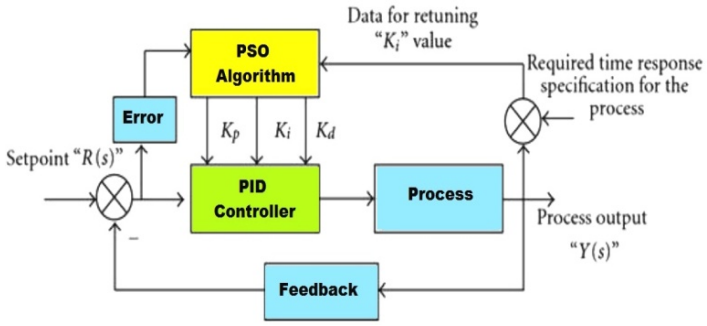

Figure 5: Chromosome structure or Block diagram for PID parameters tuning and retuning using PSO.

It is important to be accurately specified. In this paper, the fitness function (*Fobj*) is defined as follows:

$$F_{obj} = \{(100E_{SS}^{0.5} + M_P^2) + (10t_s + t_r)\}$$

where,
t_r is Rise time;
t_s is Settling time;
M_P is Overshoot;
E_{ss} is the steady state error.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijareeie.com

Vol. 8, Issue 3, March 2019

The PID controller parameters could be evaluated approximately using conventional tuning method such as Ziegler-Nichols experimental method [11].

VI. SIMULATION RESULTS

The simulation results of the input is unit step response and transfer function of DC motor using P, PI controllers, and PSO controller their performance parameters are described and compared. However, without controller, the DC motor in this case has a slow step response.

As shown in Figures (6)-(13) by using P, PI controllers and PSO controller, with two values of speed 1500 r.p.m and 2000 r.p.m under different load torque values (1.2 Nm and 1.9 Nm).
It can be illustrated from the figures that the improvement of the response under different dynamic operations. There are different cased have been considered to verified the proposed method (PSO). These cases are shown for speed responses under various dynamic operations (the load torque (TL) are 1.2Nm and 1.9)

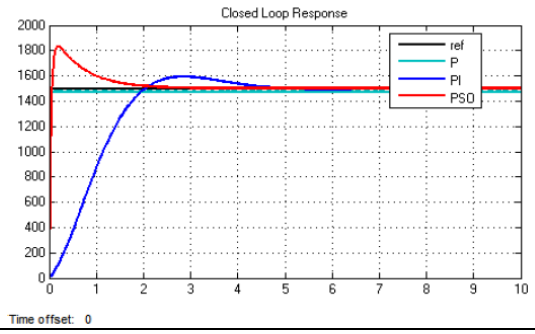

Figure 6 speed reference at 1500rpm, 5000KHz, TL= 1.2Nm at 0.12sec

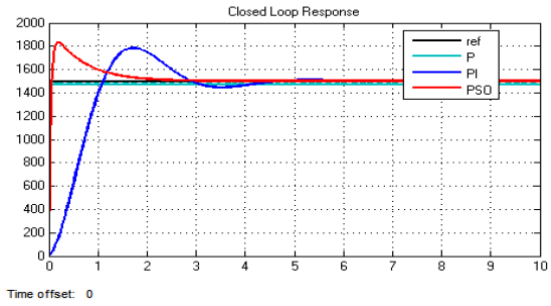

Figure 7 speed reference at 1500rpm, 10000KHz, TL= 1.2Nm at 0.12sec

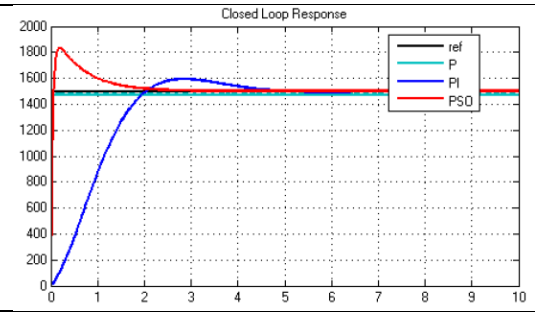

Figure 8 speed reference at 1500rpm, 5000KHz, TL= 1.9Nm at 0.9sec

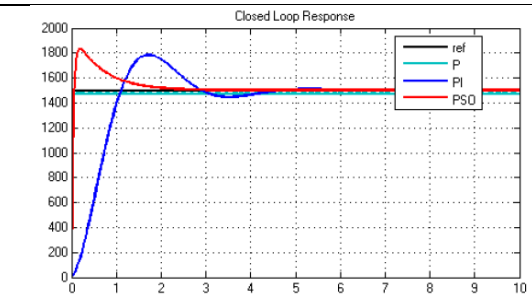

Figure 9 speed reference at 1500rpm, 10000 KHz, TL= 1.2Nm at 0.12sec

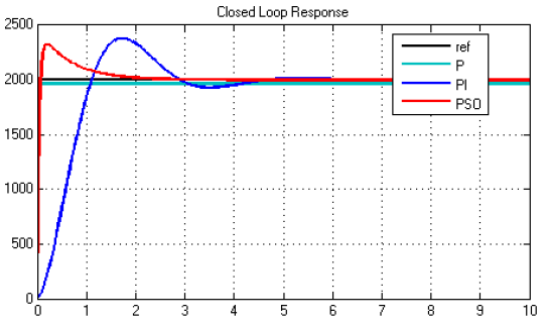

Figure 10 speed reference at 2000rpm, 5000 KHz,

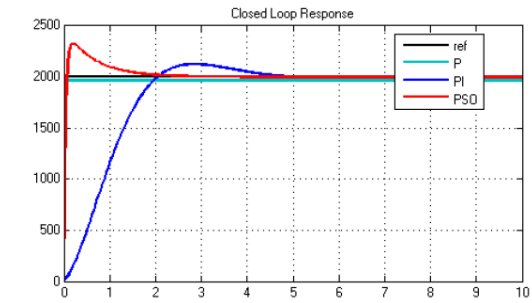

Figure 11 speed reference at 2000rpm, 10000 KHz, TL=

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijareeie.com

Vol. 8, Issue 3, March 2019

TL= 1.2Nm at 0.12sec	1.2Nm at 0.12sec
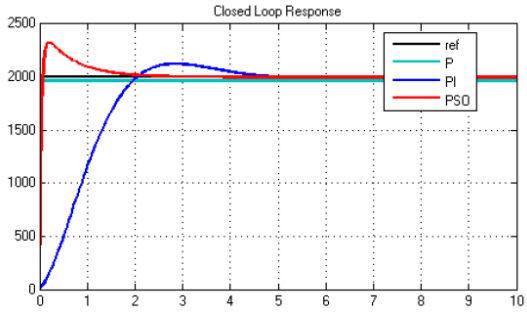	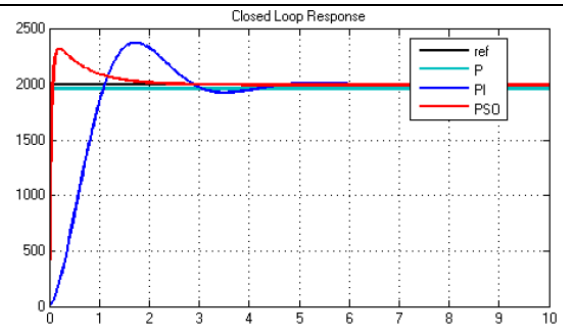
Figure 12 speed reference at 200rpm, 5000 KHz, TL= 1.9Nm at 0.32sec	**Figure 13 speed reference at 200rpm, 10000 KHz, TL= 1.2Nm at 0.12sec**

When running of PSO algorithm for different combination of c*1*, *c2* and *w* that give the optimal speed response as shown in Table 2.

Parameters	Values	Values	Values
Number of Particles	10	10	10
Max. No. of Iterations	20	20	20
C_2	1.2	1.9	1.2
C_1	0.2	0.32	0.12
Ω	0.9	0.9	1.5

Table 2 PSO parameters values

From simulation results, it was observed that under different values of speed the PI controller taken a long rise time while the PSO controller performed well in the case of sufficiently large reference input changes regarding a short settling time. It can be revealed also that the delay time is decreased in PSO controller with different dynamic operations.

VII. CONCLUSION

In this paper, the P, and PI controller has been designed and optimized the parameters of the speed controller by a Particle Swarm Optimization (PSO) technique. The proposed PSO has a good accuracy and divergence speed comparing with based method of P, and PI speed controllers according to obtained evaluation results. In addition, design of PID controller using PSO is caused that the rate of rise time, delay time, and settling time in step response curve is reduced in comparison with P, and PI. It should be noted that PSO performance in design and optimization process can be more improved by increasing the number of iterations.

REFERENCES

[1] Santosh Kumar Suman, Vinod Kumar Giri, "Speed control of DC motor using optimization techniques based PID Controller", IEEE International Conference on Engineering and Technology (ICETECH), Pages: 581 – 587, 2016
[2] Aziz Ahmed Yogesh Mohan Aasha Chauhan Pradeep Sharma Comparative Study of Speed Control of D.C.Motor Using PI, IP, and Fuzzy Controller", International Journal of Advanced Research in Computer and Communication EngineeringVol. 2, Issue 7, July 2013.
[3] Adel A. A. El-Gammal1, Adel A. El-Samahy," A Modified Design of PID Controller For DC Motor Drives Using Particle Swarm Optimization PSO", International Conference on Power Engineering, Energy and Electrical Drives, POWERENG '09.18-20 March 2009
[4] I. Petras, L. Dorcak, and I. Kostial, "Control quality enhancement by fractional order controllers," *Acta Montanistica Slovaca*, vol. 3, no. 2, pp. 143-148, April 1998.
[5] P. J. Angelinc, "Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences," *Proc. of the Seventh Annual Conference on Evolutionary Programming*, pp. 601-610, 1998.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijareeie.com

Vol. 8, Issue 3, March 2019

[6] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," *Proc. of the IEEE Congress on Evolutionary Computation*, pp.69-73, 1998.
[7] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," *Proc. of the IEEE International Conference on Neural Networks*, pp. 1942-1948, 1995.
[8] Y. Liu, Z. Qin, and X.-S. He, "Supervisor student model in particle swarm optimization," *Proc. of the IEEE Congress on Evolutionary Computation*, pp. 542-547, 2004.
[9] Clerc, M. (2012). "Standard Particle Swarm Optimisation" (PDF). HAL Open Access Archive.
[10] Bratton, Daniel; Kennedy, James (2007). Defining a Standard for Particle Swarm Optimization. Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS 2007). pp. 120–127. doi:10.1109/SIS.2007.368035. ISBN 978-1-4244-0708-8.
[11] J. Kennedy, R. Eberhart, Y. Shi, "Swarm Intelligence", Morgan Kaufmann Publishers, 1st Edition, San Francisco, pp. 80-95, 2001.