Application of Modified Shuffled Frog Leaping Algorithm for Robot Optimal Controller Design

Mohammd Pourmahmood Aghababa[1], M.E.Akbari[2], A.M. Shotorani[3], R.M.Shotorbani[4]

1. *Department of Electrical Engineering, Ahar Branch, Islamic Azad University, Ahar, Iran, m-Pourmahmood@iau-Ahar.ac.ir*
2. *Department of Electrical Engineering, Ahar Branch, Islamic Azad, University, Ahar, Iran, m-Akbari@iau-Ahar.ac.ir*
3. *Department of Electrical Engineering, Azerbaijan University of Tarbiat Moallem, Iran, a.m.shotorbani@gmail.com*
4. *Department of Mechanical Engineering, Tabriz University, Iran, r.m.shotorbani@yahoo.com*

Abstract— In this paper, a modified shuffled frog leaping (*MSFL*) algorithm is proposed to speed up the convergence of the standard shuffled frog leaping (*SFL*) method. The *MSFL* approach is based on an adaptive accelerated position changing of frogs. This modification causes a fast convergence rate and consequently achieving a rapid adaptive algorithm. The proposed method is used to design the optimal controller parameters for a five bar linkage manipulator robot. Simulation results have verified the effectiveness and robustness of the proposed method in practical issues. The features and the advantages of *MSFL* algorithm, such as escaping from local optima traps, global optimization, good robustness, simple mechanism and fast convergence, would make *MSFL* method as a promising optimization approach.

Index Terms— Shuffled frog leaping algorithm, Optimization approach, Convergence rate, PID controller, Five bar linkage manipulator robot.

---------- ◆ ----------

1 Introduction

Evolutionary algorithms (*EAs*) are stochastic optimization methods that mimic the metaphor of natural biological evolution and/or the social behavior of species. Shuffled Frog Leaping (*SFL*) is one of the new *EAs* that has been proposed by Eusuff and Lansey for determining optimal discrete pipe sizes for new pipe networks and for network expansions [1]. It is based on evolution of memes carried by the interactive individuals and a global exchange of information among themselves. The SFL works based on memetic evolution (transformation of frogs) and information exchange in the population. Frogs which are the hosts of memes (consisting memotype like gene in chromosome in GA) search the particle with the highest amount of food in a swamp by improving their memes. This characteristic can be used in an intelligent manner in control systems.

PID (Proportional-Integral-Derivative) control is one of the earliest control strategies. It has been widely used in the industrial control field. Its widespread acceptability can be recognized by: the familiarity with which it is perceived amongst researchers and practitioners within the control community, simple structure and effectiveness of algorithm, relative ease and high speed of adjustment with minimal down-time and wide range of applications where its reliability and robustness produces excellent control performances. However, successful applications of PID controllers require the satisfactory tuning of three parameters (which are proportional gain (K_P), integral time constant (K_I) and derivative time constant (K_D)) according to the dynamics of the process. Unfortunately, it has been quite difficult to tune properly the gains of PID controllers because many industrial plants are often burdened with problems such as high order, time delays and nonlinearities [4].

Traditionally, these parameters are determined by a trial and error approach. Manual tuning of PID controller is very tedious, time consuming and laborious to implement, especially where the performance of the controller mainly depends on the experiences of design engineers. In recent years, many tuning methods have been proposed to reduce the time consumption on determining the three controller parameters. The most well known tuning method is the Ziegler-Nichols tuning formula [3]; it determines suitable parameters by observing a gain and a frequency on which the plant becomes oscillatory.

Considering the limitations of the Ziegler-Nichols method and some empirical techniques in raising the performance of PID controller, recently artificial intelligence techniques such as fuzzy logic [5, 6], fuzzy neural network [7] and some stochastic search and optimization algorithms such as simulated annealing [8], genetic algorithm [9, 10, 11], particle swarm optimization approach [4], immune algorithm [12] and ant colony optimization [13] have been applied to improve the performances of PID controllers. In these studies, it has been shown that these approaches provide good solutions in tuning the parameters of PID controllers. However, there are several causes for developing improved techniques to design PID controllers. One of them is the important impact it may give because of the general use of the controllers. The other one is the enhancing operation of PID controllers that can be resulted from improved design techniques. Finally, a better tuned optimal PID controller is more interested in real world applications.

This paper proposes the *MSFL* technique as a new optimization algorithm. The proposed method is applied for determining the optimal values for parameters of PID controllers. Here, we formulate the problem of designing PID controller as an optimization problem and our goal is to design a controller with high performance by adjusting four performance indexes, the maximum overshoot, the settling time, the rise time and the integral absolute error of step response. After designing PID controllers for some simple benchmark transfer functions, an optimal PID controller is designed for a five bar linkage manipulator robot using *MSFL* algorithm. The advantages of this methodology are that it is a simple method with less computation burden, high-quality solution and stable convergence specifications.

The rest of this paper is organized as follows. In section 2, first an overview of *SFL* algorithm is given. Then, the modified version of *SFL* is introduced. Section 3 deals with the formulation of the optimal PID controller designing problem through defining a new cost function. In section 4, simulation results of optimal PID controller designing procedure are given for some benchmarks. Then, dynamic equations of the five-bar-linkage manipulator robot are illustrated following by optimal controller design using proposed *MSFL*. Finally, the paper ends with some conclusions in section 5.

2 MSFL Algorithm

In this section, the original *SFL* is briefly reviewed. Afterwards, we propose *MSFL* as an enhanced *SFL* algorithm.

2.1 SFL algorithm

SFL algorithm, introduced by Eusuff and Lansey for water distribution system optimization, is a metaheuristic for solving optimization problems [1]. *SFL* is a population based cooperative search metaphor inspired by natural memetics. The algorithm uses memetic evolution in the form of influencing of ideas from one individual to another in a local search. Conceptually, the local search is similar to particle swarm optimization. A shuffling strategy allows the exchange of information among local searchers, leading them toward a global optimum [1].

In *SFL*, the population consists of a set of frogs (solutions) partitioned into subsets, referred to as memeplexes. Different memeplexes are considered as different cultures of frogs, each performing a local search. Within each memeplex, the individual frogs hold ideas, that can be influenced by the ideas of other frogs, and evolve through a process of memetic evolution. After a defined number of memetic evolution steps, ideas are passed among memeplexes in a shuffling process [2]. The local search and the shuffling processes continue until some predefined convergence criteria are satisfied [1].

In general, a *SFL* works as follows. First, an initial population of *P* frogs is created randomly. Afterwards, the frogs are sorted in a descending order according to their fitness. Then, the entire population is divided into m memeplexes, each containing n frogs. In this process, the first frog goes to the first memeplex, the second frog goes to the second memeplex, frog *m* goes to the *m*th memeplex, and frog *m*+1 goes back to the first memeplex and so on. Within each memeplex, the frogs with the best and the worst fitnesses are identified as X_b and X_w, respectively. Also, the frog with the global best fitness is identified as X_g. Then, a process is applied to improve only the frog with the worst fitness and (not all the frogs) in each cycle.

Accordingly, the position of the frog with the worst fitness is adjusted as follows [1]:

Chang frog position:

$$D_i = rand \times (X_b - X_g) \tag{1}$$

New position:

$$X_{i+1} = X_i + D_i \quad \text{where} \quad -D_{max} \leq D_i \leq D_{max}, \tag{2}$$

where rand is a random number between 0 and 1, and D_{max} is the maximum allowed change in a frog's position. If this process produces a better solution, it is replaced for the worst frog. Otherwise, the calculations in equations (1) and (2) are repeated but with respect to the global best frog (i.e. X_b is replaced by X_g). If no improvement is possible, then a new solution is randomly generated to replace the worst frog. Hence, the calculations continue for a specific number of iterations [1]. Accordingly, the main parameters of *SFL* are: number of frogs *P*; number of memeplexes; number of generation for each memeplex before shuffling; number of shuffling iterations; and maximum step size.

2.2 Modified shuffled frog leaping algorithm

The main drawback of SFL algorithm is slow convergence, closely related to the lack of adaptive acceleration terms in the position updating formula. In equation (1), rand determines the movement step sizes of frogs through the X_b and X_w positions. In the standard SFL, these step sizes are random numbers between 0 and 1 for all frogs.

In each cycle, the value of the objective function is a criterion that presents the relative improvement of a frog movement with respect to the previous one. Thus the difference between the values of the objective function in consequent iterations can represent the frog acceleration. Therefore, position changing formulae turns to the following form.

$$D_i = rand \times C \times (f(X_b) - f(X_w)) \times (X_b - X_w) \tag{3}$$

New position

$$X_{i+1} = X_i + D_i \tag{4}$$

where $C \in (0, C_{max}]$ is a constant, C_{max} is a case dependant upper limit, f(X_b) and f(X_w) are the best and the worst fitness functions that are found by the frogs in each memeplexs. Similar to the original *SFL*, if the process produces a better solution, the worst frog is replaced by the better one. Otherwise, the calculations in equations (3) and (4) are repeated with respect to the global best frog instead (i.e. X_g and f(X_g) replace X_b and f(X_b), respectively). If no improvement is possible, then a new solution is randomly generated to replace the worst frog.

The proposed modification term, (f(X_b) - f(X_w)), called adaptive coefficient, causes an adaptive movement. In each iteration, the modification term defines the movement size, adaptively. Therefore, the adaptive coefficient decreases/ increases the movement size relative to being closer/farther from the optimum point, respectively. By means of this method, position changing can be updated adaptively instead of being fixed or changed linearly. Therefore, using the adaptive coefficient, the convergence rate of the algorithm will be increased rather than being performed by proportional large or short steps. So, the above modification accelerates the convergence of the algorithm. This new version is called modified *SFL* (*MSFL*). The main characteristics of *MSFL* algorithm are: adaptive movements, fast convergence, better diversification ability and escaping from local optima. Finally, the proposed *MSFL* is still a general optimization algorithm that can be applied to any real world continuous optimization problems.

3 MSFL Algorithm for Designing PID Controller

The PID controller is used to improve the dynamic response and to reduce the steady-state error. The transfer function of a PID controller is described as:

$$G(s) = Kp + K_I / s + K_D s \tag{5}$$

where K_P, K_I and K_D are the proportional gain, integral and derivative time constants, respectively. For designing an optimal PID controller, a suitable objective function that represents system requirements, must be defined in the first step. A set of good control parameters K_P, K_I and K_D can produce a good step response that will resultant in minimization of performance criteria. The optimal PID controller parameters that minimize the performance indexes are designd using the proposed *MSFL* algorithm. This section deals with defining the objective function. Then an efficient procedure is proposed to design an optimal PID controller. Finally, the effectiveness and efficiency of the proposed *MSFL* algorithm in designing optimal PID controller is tested using some benchmark plants. All the computations are implemented with Matlab®/Simulink®.

3.1 Objective function definition

In the design of a PID controller, the performance criterion or objective function is first defined based on some desired specifications and constraints under input testing signal. Some typical output specifications in the time domain are overshoot, rise time, settling time, and steady-state error. In general, three kinds of performance criteria, the integrated absolute error (*IAE*), the integral of squared-error (*ISE*), and the integrated of time-weighted-squared-error (*ITSE*) are usually considered in the control design under step testing input, because they can be evaluated analytically in the frequency domain. It is worthy to notice that using different performance indices probably makes different solutions for PID controllers. The three integral performance criteria in the frequency domain have their own advantages and disadvantages. For example, a disadvantage of the *IAE* and *ISE* criteria is that their minimization can result in a response with relatively small overshoot but a long settling time. Although the *ITSE* performance criterion can overcome the disadvantage of the *ISE* criterion, the derivation processes of the analytical formula are complex and time-consuming [4]. The IAE, ISE, and ITSE performance criteria formulas are as follows:

$$IAE = \int_0^\infty |r(t) - y(t)| \; dt = \int_0^\infty |e(t)| \; dt \tag{6}$$

$$ISE = \int_0^\infty e^2(t) \; dt \tag{7}$$

$$ISTE = \int_0^\infty t e^2(t) \; dt \tag{8}$$

In this paper, another time domain performance criterion defined by

$$\begin{aligned} min_K \; W(K) = & (1/(1+e^{-\alpha})) \times (T_s + T_r) + \\ & (e^{-\alpha}/(1+e^{-\alpha})).(M_p + E_{ss}) \end{aligned} \tag{9}$$

is used for evaluating the PID controller, where *K* is [K_P, K_I, K_D], and $\alpha \in$ [-5, 5] is the weighting factor. The optimum selection of α depends on designer's requirements and the characteristics of the plant under control. We can set α to be smaller than 0 to reduce the overshoot and steady-state error. On the other hand, we can set α to be larger than 0 to reduce the rise time and settling time. Note that, if α is set to 0, then all performance criteria (i.e. overshoot, rise time, settling time, and steady-state error) will have the same worth.

3.2 MSFL based PID Controller

For designing an optimal PID controller, determination of vector *K* with regards to the minimization of performance index is the main issue. Here, the minimization process is performed using the proposed *MSFL* algorithm. For this purpose, step response of the plant is used to compute four performance criteria overshoot (M_p), steady-state error (E_{ss}), rise time (T_r) and setting time (T_s) in the time domain. At first, the lower and upper bounds of the controller parameters should be specified. Then a population of frogs is initialized, randomly in the specified range. Each frog represents a solution (i.e. controller parameters *K*) that its performance index should be evaluated. This work is performed by computing M_p, E_{ss}, T_r, and T_s using the step response of the plant, iteratively. Then, by using the four computed parameters, the performance index is evaluated for each frog according to these performance criterions. Now the main procedure of *MSFL* algorithm starts as follows. The frogs are sorted in a descending order according to their performance index. Then, the entire population is divided into m memeplexes, as mentioned earlier in section 2. Within each memeplex, X_b, X_w and X_g are determined. Then, the frog with the worst fitness is improved using the modified process mentioned in section 2.2. Afterwards, the superseding frogs are created, by the procedure mentioned in the section 2.2. This process is repeated until a satisfactory of a stopping criterion. In this stage, the frog corresponding to X_g is designated as the optimal vector *K*.

The flowchart of designing PID controller based on *MSFL* is shown in Fig. 1.

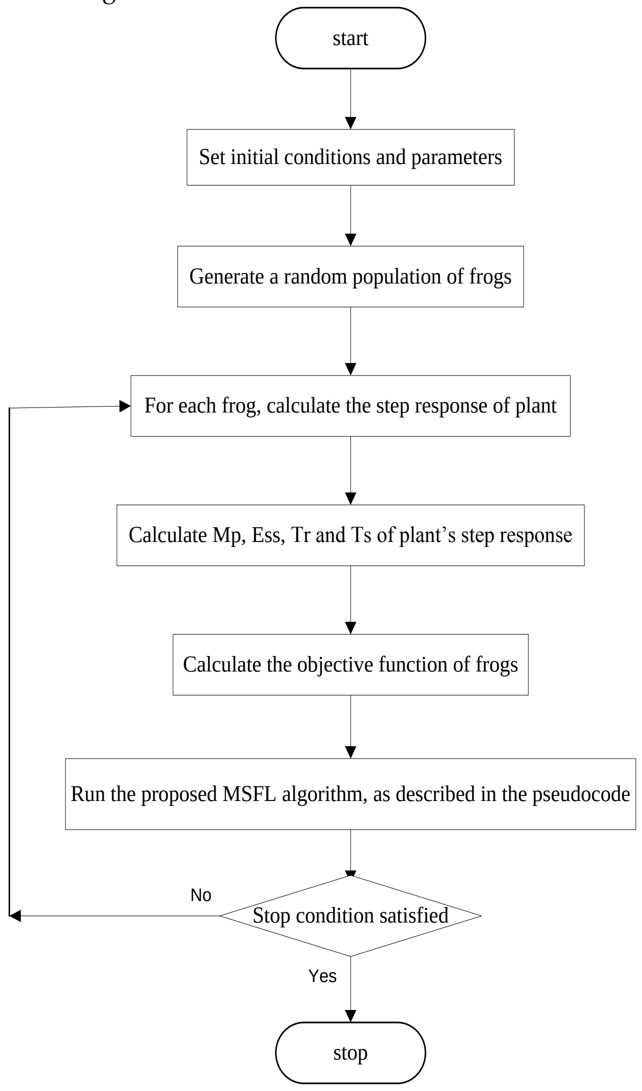
start
Set initial conditions and parameters
Generate a random population of frogs
For each frog, calculate the step response of plant
Calculate Mp, Ess, Tr and Ts of plant's step response
Calculate the objective function of frogs
Run the proposed MSFL algorithm, as described in the pseudocode
No
Stop condition satisfied
Yes
stop

Fig. 1. The flowchart of *MSFL* based PID Controller design procedure

3.3 Optimal PID Controller for Typical Transfer Functions

In order to verify the modified *SFL*'s favourable performance, comparison experiments have been carried out for the following four typical control plants. The transfer functions of four control systems are given as follows.

$$G_1(s)=\frac{1}{s^3+6s^2+7s+10}); \; 0\le K_P\le 10,\ 0\le K_I\le 10,\ 0\le K_D\le 10$$

$$G_2(s)=\frac{27}{(s+1)(s+3)^3}; \; 0\le K_P\le 5,\ 0\le K_I\le 5,\ 0\le K_D\le 5$$

$$G_3(s)=\frac{3e^{-0.1s}}{(s^2+1.2s+1)(s+3)}; \; 0\le K_P\le 5,\ 0\le K_I\le 5,\ 0\le K_D\le 5$$

$$G_4(s)=\frac{e^{-0.1s}}{(s+1)(0.5s+1)(0.25s+1)(0.125s+1)};$$
$$0\le K_P\le 5,\ 0\le K_I\le 3,\ 0\le K_D\le 3$$

Different settings were evaluated to determine suitable values for parameters of *MSFL* and *SFL* algorithms. A population of 100 frogs, 10 memeplexes, and 5 iterations per memeplex were found suitable to obtain good solutions for both *SFL* and *MSFL* algorithms. The maximum number of iterations for all experiments is considered to be 100. Also, α is set to 0 to have the same merit for all performance criteria in the objective function.

In order to obtain the optimal PID controller parameters, in each experiment, *MSFL* and *SFL* are run 10 times with 200 iterations. The best solutions of different case studies, which have the optimal or near optimal PID controller parameters, are summarized in Table 2. Using these PID controller parameters, the unit step response of each case study was obtained as shown in the figurs 2-5. From the simulation results, it can be found that *MSFL* is better than *SFL,* considering the best performance index value. Moreover, *MSFL* produces the smooth curve for the output in conjunction with little fluctuation and small overshoot.

TABLE 1
SUMMARY OF SIMULATION RESULTS OF BENCHMARK PLANTS

Case	algorithm	P	I	D	M_p	T_s	T_r	E_{ss}	cost	iteration
1	*MSFL*	9.6	10	8.6	4.7	1.905	1.266	0	3.9355	37
	SFL	8.56	9.6	5.76	8.6	2.92	1.10	0	6.3100	123
2	*MSFL*	2.8	0.99	1.41	2.25	1.605	0.97	0	2.4125	45
	SFL	3.2	1.1	1.6	5.5	2.96	0.875	0	4.6675	121
3	*MSFL*	2	0.8	1.5	1.5	2.04	1.255	0	2.3975	57
	SFL	2.3	1.4	1.4	2.7	3.53	0.96	0	7.1900	154
4	*MSFL*	2.5	1	1.1	3.7	1.575	0.912	0	3.0935	66
	SFL	2.2	1.3	1.2	9.4	5.05	0.96	0	7.7050	161

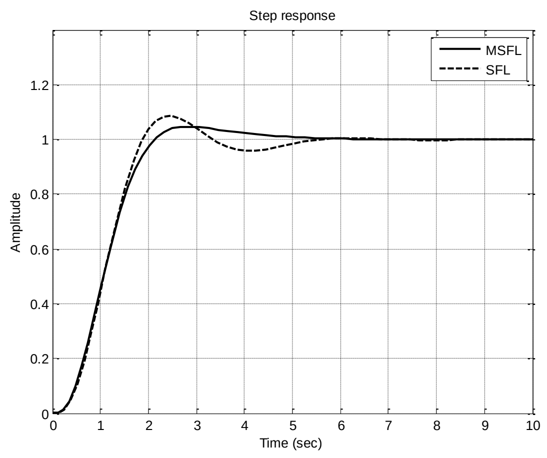

Fig. 2. Comparison of step response of the $G_1(s)$ plant using *MSFL* and *SFL* methods

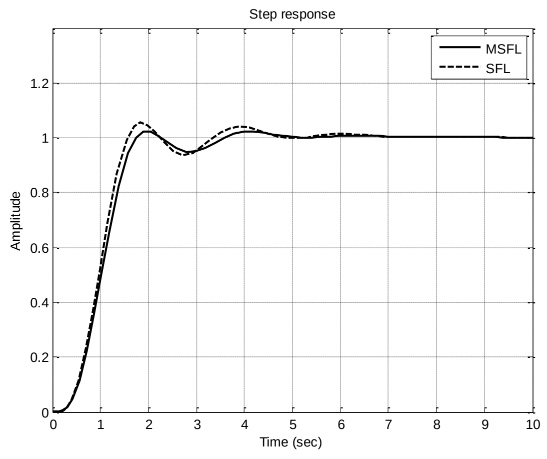

Fig. 3. Comparison of step response of the $G_2(s)$ plant using *MSFL* and *SFL* methods

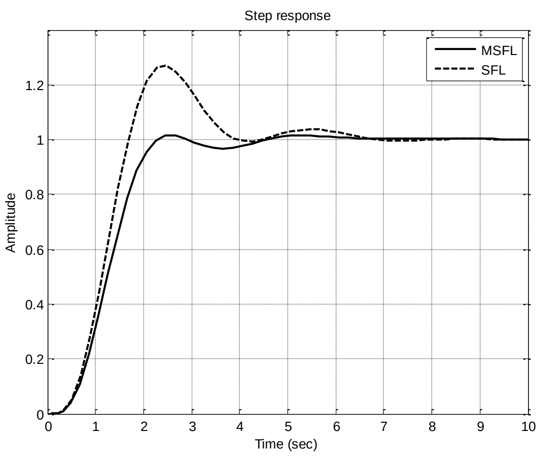

Fig. 4. Comparison of step response of the *G₃(s)* plant using *MSFL* and *SFL* methods

4 Application of MSFL for Robot Optimal Controller Design

In this section, an optimal PID controller is designed for a five-bar-linkage manipulator robot. The results of *MSFL* method are compared by the results of the standard *SFL* algorithm.

4.1 Optimal PID Controller for five-Bar-Linkage Manipulator Robot

To show the efficiency and desirable performance of the proposed algorithm in designing optimal PID controllers, a well known Mechatronics application, i.e., a robot is considered. The examined robot configuration is a five-bar-linkage. Dynamic equations of the robot are described in the following subsection. Afterwards, *MSFL* algorithm for an optimum PID controller is utilized.

4.1.1 Dynamic Equations of five-bar-Linkage Manipulator Robot

In recent years, there has been a growing interest in the design and control of lightweight robots. Several researchers have studied the modelling and control of a single link flexible beam [15]. Fig. 6 shows the 5 bar linkage manipulator built in our robotics research lab. Also, Fig. 7 depicts the five-bar linkage manipulator schematic where the links form a parallelogram. Let q_i, T_i and I_h^i be the joint variable, torque and hub inertia of the ith motor, respectively. Also, let I_i, l_i, d_{Ci} and m_i be the inertia matrix, length, distance to the centre of gravity and mass of the ith link, correspondingly.

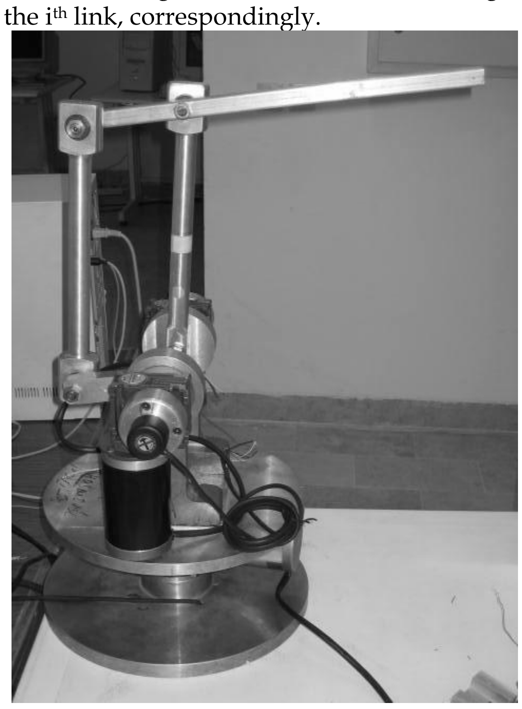

Fig. 6. Planar presentation of robot

The dynamic equations of the manipulator are [17]:

$$T_1 = (M_{11} + I_h^1)\ddot{q}_1 + M_{12}\ddot{q}_2 + \frac{\partial M_{12}}{\partial q_2}\dot{q}_2^{\,2} + g(m_1 d_{c1} + m_3 d_{c3} + m_4 l_1)\cos q_1 \tag{10}$$

$$T_2 = (M_{22} + I_h^2)\ddot{q}_2 + M_{21}\ddot{q}_1 + \frac{\partial M_{21}}{\partial q_1}\dot{q}_2^{\,2} + g(m_1 d_{c2} + m_3 l_2 + m_4 d_{c4})\cos q_2 \tag{11}$$

where g is the gravitational constant and

$$M_{11} = I_{11}^1 + I_{11}^3 + m_1 d_{c1}^2 + m_3 d_{c3}^2 + m_4 I_1^2 \tag{12}$$

$$M_{22} = I_{11}^2 + I_{11}^4 + m_2 d_{c2}^2 + m_3 l_2^2 + m_4 d_{c4}^2 \tag{13}$$

$$M_{12} = M_{21} = (m_3 d_{c3} l_2 - m_4 d_{c4} l_1)\cos(q_1 - q_2) \tag{14}$$

It's noticed from (12)-(14) that for

$$m_3 l_{c3} l_2 = m_4 l_{c4} l_1 \tag{15}$$

we have M_{12} and M_{21} equal to zero, that is, the matrix of inertia is diagonal and constant. Hence the dynamic equations of this manipulator will be

$$T_1 = (M_{11} + I_h^1)\ddot{q}_1 + g(m_1 l_{c1} + m_3 l_{c3} + m_4 l_1)\cos q_1, \tag{16}$$

$$T2 = (M_{22} + I_h^2)\ddot{q}_2 + g(m_1 l_{c2} + m_3 l_2 + m_4 l_{c4})\cos q_2, \tag{17}$$

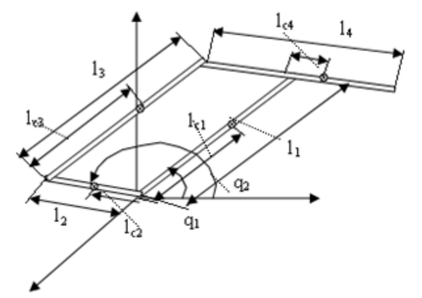

Fig. 7. Planar presentation of the robot

Notice that T_1 depends only on q_1 but not on q_2. On the other hand T_2 depends only on q_2 but not on q_1. This discussion helps to explain the popularity of the parallelogram configuration in industrial robots. If the condition (15) is satisfied, then we can adjust the two rotations independently, without worrying about interactions between them.

4.1.2 Simulation Results

Having 2 motors, the manipulator specification consisting of mass, length and centre of gravity of links are given in Table 2. The main purpose is designing an optimal PID controller for each of motors to control their rotations, with good performance. Using equations (16) and (17), five-bar-linkage manipulator robot is easily simulated using Matlab and Simulink. The block diagram of the five-bar-linkage manipulator robot with PID controller for motor 1 is shown in Fig. 8. The block diagram for motor 2 is similar to this figure. A population of 80 frogs, 8 memeplexes, and 4 iterations per memeplex were found suitable for obtaining good solutions for both *SFL* and MSFL algorithms. The maximum iteration of all experiments is considered equal to 200. Also, α is set to 0 for all performance criteria to have the same merit in the objective function.

The following process is done to determine the optimal values of the PID controller parameters (i.e., vector *K*). First, the lower and upper bounds of the three controller parameters are selected as 0 and 30, respectively. Then, all frogs of the population are initialized, randomly. Each frog *K* (the controller parameters) is sent to Matlab® Simulink® block and the values of four performance criteria in the time domain, i.e., M_p, E_{ss}, T_r and T_s are calculated iteratively. Afterwards, the objective function is evaluated for each frog according to these performance criteria. Then, the procedure of *MSFL* algorithm is performed, as illustrated in the flowchart of Fig. 3. At the end of any iteration, the program checks the stop criterion. When one termination condition is satisfied, the program stops and the latest global best solution X_g is the best solution of K.

Fig. 9 illustrates the step response without PID controllers for two motors. Figures 10 and 11 show the step response of rotation for motors 1 and 2, respectively. The simulation results of the best solution are summarized in Table 3. These results demonstrate that cost function is converged rapidly.

TABLE 2
FIVE-BAR-LINKAGE MANIPULATOR DATA

Link	Mass (Kg)	Length (m)	*C* of G (m)
1	0.288	0.33	0.166
2	0.0324	0.12	0.06
3	0.3702	0.33	0.166
4	0.2981	0.45	0.075

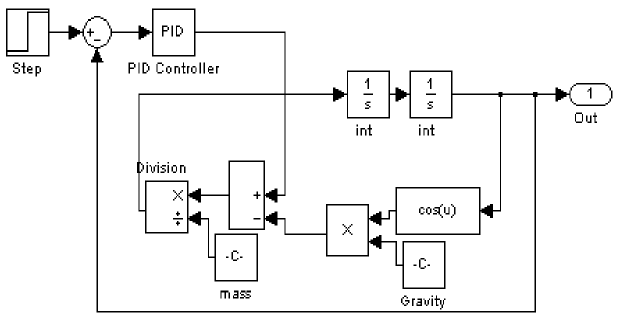

Fig. 8. Block diagram of the motor with PID controller.

In conclusion, MSFL algorithm has rapid convergence characteristic and is highly effective in solving the optimal tuning problem of PID controller parameters.

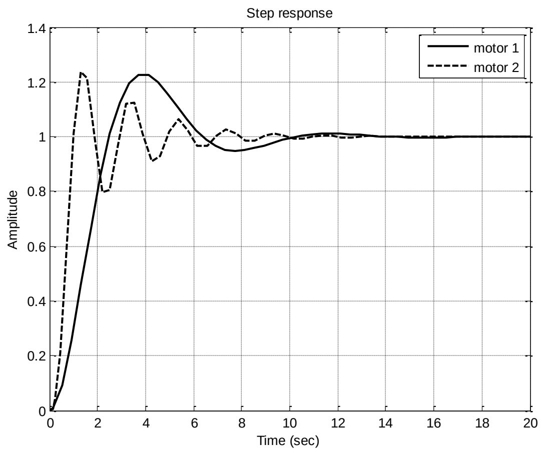

Fig. 9. Step response of the robot motors without PID controller

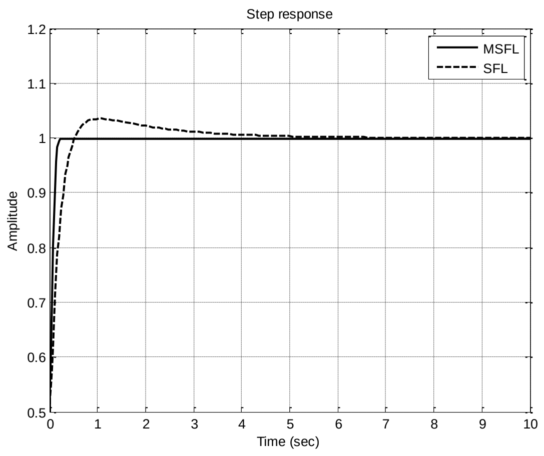

Fig. 10. Comparison of step response of the motor #1 rotation using *MSFL* and *SFL* methods.

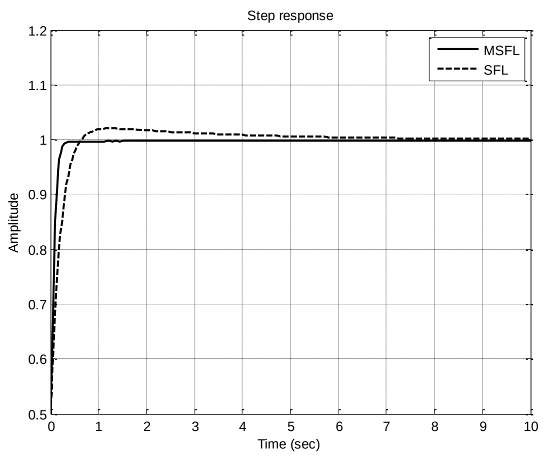

Fig. 11. Comparison of step response of the motor #2 angel using *MSFL* and *SFL* methods.

TABLE 3
SUMMARY OF SIMULATION RESULTS OF FIVE-BAR-ROBOT MOTORS

motor	algorithm	*P*	*I*	*D*	M_p	T_s	T_r	E_{ss}	cost	iteration
1	*MSFL*	30	1.5	2.1	0	0.136	0.11	0	0.1230	36
	SFL	17.9	10.1	3.6	3.5	0.375	0.29	0	2.0825	142
2	*MSFL*	28.6	1.1	2.4	0	0.172	0.13	0	0.1510	32
	SFL	21.7	6.7	4.4	2	0.415	0.31	0	1.3625	154

5 CONCLOUSION

In this paper a modified *SFL* algorithm (*MSFL*) was proposed to improve the performance of the standard *SFL* algorithm. Different benchmarks were used to illustrate the mentioned advantages. Dealing with this problem, a new time domain performance criterion was proposed. In all case studies, *MSFL* performed better than *SFL* approach which exposed *MSFL* as a promising optimization method. The optimal controller design of the five-bar-linkage manipulator robot has been considered, as a practical application. The proposed method was implemented for tuning the controller for the robot. High promising results demonstrate that the proposed algorithm is robust, efficient and can obtain higher quality solution with better computational efficiency.

REFERENCES

[1] M.M. Eusuf, KE. Lansey, "Optimization of water distribution network design using the shuffled frog leaping algorithm." *J Water Resour Plan Manage*, Vol 129, no.3, pp. 210–225, 2003.
[2] S.Y. Liong, M.d. Atiquzzaman., "Optimal design of water distribution network using shuffled complex evolution." *J Inst Eng*, Singapore, vol. 44, no. 1, pp. 93–107, 2004.
[3] J.G. Ziegler and N.B. Nichols, "Optimum settlings for automatic controllers," *Trans. On ASME.*, vol. 64, pp. 759-768, 1942.
[4] Z.L. Gaing, "A Particle Swarm Optimization Approach for Optimum Design of PID controller in AVR system," *IEEE Transactions on Energy Conversion,* vol. 9, no. 2, pp. 384-391, 2003.
[5] Z.Y. Zhao, M. Tomizuka, and S. Isaka, "Fuzzy gain scheduling of PID controllers," *IEEE Trans. System, Man, and Cybernetics,* vol. 23, no. 5, pp. 1392-1398, 1993.
[6] A. Visioli, "Fuzzy logic based set-point weight tuning of PID controllers," *IEEE Trans. System, Man, and Cybernetics – Part A: System and Humans,* vol. 29, no. 6, pp. 587-592, 1999.
[7] S.Y. Chu, C.C. Teng, "Tuning of PID controllers based on gain and phase margin specifications using fuzzy neural network," *Fuzzy Sets and Systems,* vol. 101, no. 1, pp. 21-30, 1999.
[8] G. Zhou and J.D. Birdwell, "Fuzzy logic-based PID autotuner design using simulated annealing," *Proceedings of the IEEE/IFAC Joint Symposium on Computer-Aided Control System Design,* pp. 67 – 72, 1994.
[9] D.P. Kwok and F. Sheng, "Genetic algorithm and simulated annealing for optimal robot arm PID control," *Proc IEEE Conf. Evolutionary*

Computation, pp. 707–713, 1994.

[10] R.A. Krohling and J.P. Rey, "Design of optimal disturbance rejection PID controllers using genetic algorithm," *IEEE Trans. Evol. Comput.*, vol. 5, pp. 78–82, 2001.

[11] P. Wang and D.P. Kwok, "Optimal design of PID process controllers based on genetic algorithms," *Control Engineer Practice,* vol. 2, no. 4, pp.641-648, 1994.

[12] D.H. Kim, "Tuning of a PID controller using a artificial immune network model and local fuzzy set," *Proceedings of the Joint 9th IFSA World Congress and 20th NAFIPS International Conference,* vol. 5, pp. 2698 – 2703, 2001.

[13] Y.T. Hsiao, C.L. Chuang, and C.C. Chien, "Ant colony optimization for designing of PID controllers," *Proceedings of the 2004 IEEE Conference on Control Applications/ International Symposium on Intelligent Control/International Symposium on Computer Aided Control Systems Design,* Taipei, Taiwan, 2004.

[14] J.P. Huissoon and D. Wang, "On the design of a five-bar-linkage manipulator" *Robotica,* vol. 9, pp 441-446, 1991.

[15] D. Wang and M. Vidyasagar, "Modeling of a five-bar-Linkage Manipulator with One Flexible Link," *in Proc. IEEE Int. Symp, subject,* Turkey, pp. 21–26, 1988.

[16] Asada, Haruhiko, Youcef-Toumi, and Kamal, "Analysis and Design of a Direct-Drive Arm with a Five-Bar-Link Parallel Drive Mechanism," *Journal of Dynamic Systems, Measurement, and Control,* 1984.

[17] D.Wang, J.P.Huissoon and K.Luscott, "A teaching robot for demonstrating robot control strategies," *manufacturing research corporation of Ontario,* 1993.