Intelligent Control and Automation, **2013, 4, 42-54**
http://dx.doi.org/10.4236/ica.2013.41007 Published Online February 2013 (http://www.scirp.org/journal/ica)

Current Search: Performance Evaluation and Application to DC Motor Speed Control System Design

Deacha Puangdownreong
Department of Electrical Engineering, South-East Asia University, Bangkok, Thailand
Email: deachap@sau.ac.th

Received September 27, 2012; revised October 27, 2012; accepted November 4, 2012

ABSTRACT

This paper proposes the current search (CS) metaheuristics conceptualized from the electric current flowing through electric networks for optimization problems with continuous design variables. The CS algorithm possesses two powerful strategies, exploration and exploitation, for searching the global optimum. Based on the stochastic process, the derivatives of the objective function is unnecessary for the proposed CS. To evaluate its performance, the CS is tested against several unconstrained optimization problems. The results obtained are compared to those obtained by the popular search techniques, *i.e.*, the genetic algorithm (GA), the particle swarm optimization (PSO), and the adaptive tabu search (ATS). As results, the CS outperforms other algorithms and provides superior results. The CS is also applied to a constrained design of the optimum PID controller for the dc motor speed control system. From experimental results, the CS has been successfully applied to the speed control of the dc motor.

Keywords: Current Search; Metaheuristics; PID Controller; DC Motor Speed Control System

1. Introduction

Over the last five decades, many metaheuristic algorithms have been developed to solve combinatorial and numeric optimization problems. They have been also successfully applied to solve various real-world engineering optimization problems. Most metaheuristic algorithms are based on the stochastic process. This means that the derivatives of the objective function is unnecessary. Unlike the conventional or classical algorithms, they are usually based on the deterministic process commonly using the derivatives (or gradient information) to find the optimum solution. By literatures, metaheuristics can be classified in many ways depending on their nature, for example, evolutionary; nature-inspired, bio-inspired, and physic-inspired; population-based and single-solution (trajectory-) based [1,2]. The most powerful metaheuristics should have at least two major properties, *i.e.*, exploration (or diversification) and exploitation (or intensification). Exploration or diversification means to generate diverse solutions to explore the search space on the global scale. Exploitation or intensification means to focus on the search in a local region by exploiting the information to reach the best local solution within this region [2,3]. Population-based metaheuristics algorithms, such as genetic algorithm (GA) [4,5], ant colony optimization (ACO) [6], particle swarm optimization (PSO) [7], harmony search (HS) [8], firefly search (FS) [9], hunting search (HuS) [10], cuckoo search (CuS) [11,12], and bat-inspired search (BS) [13], have strong explorative property, while single-solution based metaheuristics algorithms, such as simulated annealing (SA) [14,15], tabu search (TS) [16,17], and adaptive tabu search (ATS) [18,19], have strong exploitation characteristics.

In 2012, the current search (CS) metaheuristics was proposed to solve optimization problems [20]. The CS algorithms are conceptualized from the electric current flowing through electric networks. It has been successfully applied to design the PID controllers for unstable systems [21], and the PIDA controllers for hard-to-be-controlled systems [22]. This paper consists of five sections. The CS algorithms are explained in senses of exploration and exploitation properties as appeared in Section 2. The CS metaheuristics is conducted to evaluate its performance via testing against several well-known unconstrained optimization problems. Results obtained are compared among the genetic algorithm (GA), the particle swarm optimization (PSO), and the adaptive tabu search (ATS) as presented in Section 3. The CS is applied to a constrained design of the optimum PID controller for the dc motor speed control. The experimental setup, results, and discussions are illustrated in Section 4, while conclusions follow in Section 5.

2. Current Search Algorithms

Algorithms of the CS metaheuristics are conceptualized from the electric current flowing through electric net-

works. The behavior of electric current is like a tide that always flow from higher to lower places. The less the resistance of blanch in electric networks, the more the current flows. All blanches connected in networks represent the feasible solutions in search space. The local entrapment is occurred when the current hits the open network connection. The optimum solution found is located at the end of the optimum current path. Algorithms of the CS are described as follows.

Step 1. Initialize the search space Ω, iteration counter $k = j = l = 1$, maximum allowance of solution cycling $j_{\max}$, number of initial solutions (feasible current path in a network) N, number of neighborhood members n, maximum objective function value ε, search radius R, and set $\Psi = \Gamma = \Xi = \varnothing$.

Step 2. Uniformly random initial solution $\boldsymbol{X}_i$, $i = 1,\cdots, N$ within Ω. Evaluate the objective function $f(\boldsymbol{X}_i)$ of $\forall \boldsymbol{X}$ Rank $\boldsymbol{X}_i$, $i = 1,\cdots, N$ giving $f(X_1) <\cdots< f(X_N)$, then store ranked $\boldsymbol{X}_i$ into set Ψ.

Step 3. Let $\boldsymbol{x}_0 = \boldsymbol{X}_k$ as selected initial solution.

Step 4. Uniformly random neighborhood $\boldsymbol{x}_{k,l}$, $l = 1, \cdots, n$ around $\boldsymbol{x}_0$ within radius R. Evaluate the objective function $f(\boldsymbol{x}_{k,l})$ of $\forall \boldsymbol{x}$. A solution giving the minimum objective function is set as $\boldsymbol{x}^*$.

Step 5. If $f(\boldsymbol{x}^*) < f(\boldsymbol{x}_0)$, keep $\boldsymbol{x}_0$ in set Γ_k and set $\boldsymbol{x}_0 = \boldsymbol{x}^*$, set $j = 1$ and return to Step 4. Otherwise keep $\boldsymbol{x}^*$ in set Γ_k and update $j = j + 1$.

Step 6. If $f(\boldsymbol{x}_0) < \varepsilon$, adjust radius $R = \rho R$, $\rho \in [0, 1]$ to speed up the search process.

Step 7. If $j < j_{\max}$, return to Step 4. Otherwise keep $\boldsymbol{x}_0$ in set Ξ and update $k = k + 1$.

Step 8. Stop the search process when termination criteria (TC) are satisfied. The optimum solution found is $\boldsymbol{x}_0$ Otherwise return to Step 3.

The CS algorithms can be summarized by the pseudo code as shown in **Figure 1** and can be represented by the flow diagram in **Figure 2**. In Step 1, the search space Ω performs the feasible boundary where the electric current can flow. The maximum allowance of solution cycling $j_{\max}$ implies the local entrapment occurred in the selected path. The number of initial solutions N is set as feasible paths of the electric currents. The number of neighborhood members n provides the sub-directions of the electric currents in the selected path, and the search radius R is given as the sub-search space where the electric current can flow in the selected path. Ψ, Γ, and Ξ are provided as the memory lists to store ranked initial solutions, solutions found in each search path, and the best solution in each search path, respectively.

The uniformly random approach is conducted in Step 2 to perform the feasible path of the electric currents within Ω. These paths will be ranked by the objective function to arrange the signification of paths from most to least. This operation is regarded as the exploration property to generate diverse solutions to explore the search

Pseudo-code of ACS

```
procedure
    Initialize the search space = Ω,
    the memory lists (ML) Ψ = Γ = Ξ = ϕ,
    the iteration counter i = j = k = l = 1,
    the maximum search iteration in each path I_max,
    the maximum allowance of solution
   cycling j_max, the number of initial solutions
   ( feasible paths of currents in network) N,
   number of neighborhood members n,
   the maximum objective function value ε,
   and the search radius R.
   while (i < N) do // generate search paths
        Generate initial solutions X_i randomly
        within Ω; Evaluate f(X_i) via the objective
        function; Rank X_i giving f(X_i) from min
        to max values (X_1 gives the min, while
        X_N gives the max objective function values);
        Store ranked X_i into Ψ;
   end_while
   while (k ≤ N) or the TC are not satisfied do
        Set x_0 = X_k;
        while (i ≤ I_max) do
             Generate neighborhood x_{k,l} (l =1,2,…,n)
             randomly around x_0 within R;
             for l ←1 to n do
                Evaluate f(x_{k,l}) via the objective
                function ; Set x* as a solution giving
                the minimum objective value ;
             end_for
             if f(x*) < f(x_0) do
                Keep x_0 into Γ_k;
                Set x_0 = x*;
                Set j = 1;
             else do
                Keep x* into Γ_k;
                Update j = j+1;
             end_if
             if f(x_0) < ε do
                Adapt R = ρR, ρ ∈ [0, 1];
             end_if
             if j == j_max do // activate other paths
                Keep x_0 into Ξ;
                Update k = k+1;
                Go out -loop while;
             end_if
             Update i = i+1;
        end_while
        Keep x_0 into Ξ;
        Update k = k+1;
   end_while
end_procedure
```

Figure 1. Pseudo code of the CS algorithms.

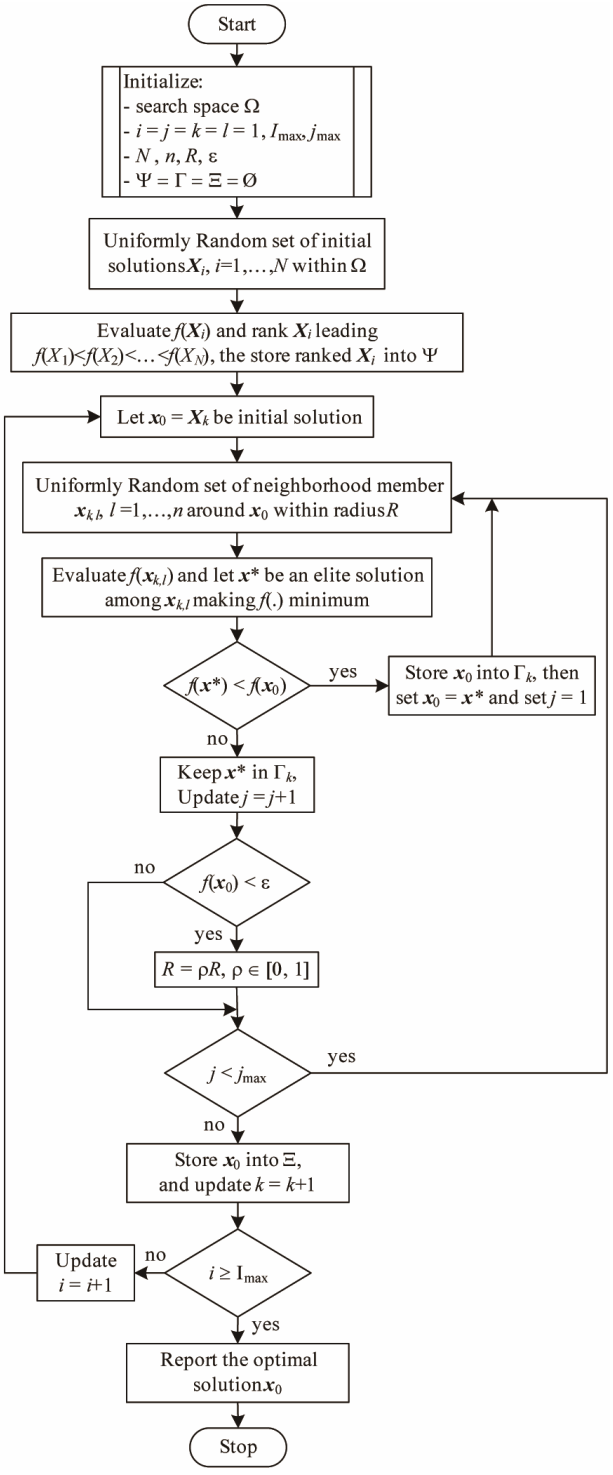


Figure 2. The flow diagram of the CS algorithms.

space on the global scale. The more the numbers of initial solutions, the more the diverse solutions are generated over the search space. Once the most significant path of the current is selected in Step 3 - 5, the search process will consecutively find the optimum solution within the sub-search space.

In Step 6, the search radius R is gradually decreased when the search comes close to local or global solutions. In this situation, the shorter the search radius, the greater the probability of the optimum solution can be found. This operation is regarded as the exploitation property focusing on the search by exploiting the information to reach the best local solution within a local region.

In Step 7 - 8, each feasible solution will be evaluated via the objective function until the optimum solution is found. The local entrapment in the selected path will be identified via the maximum allowance of solution cycling. If occurred, the second, the third, and so on, of the significant paths ranked in Step 2 will consecutively employed, until optimum solution will be found or the TC will be met.

Generally, the TC can be defined in many ways depending on the particular problems. One TC can be determined as the maximum number of search iterations which can be defined from the ratio of search space to search radius. There are other TC approaches, such as maximum number of objective function evaluation, maximum iteration without improvement in solution quality, and maximum CPU time, for example. In engineering optimization problems, TC usually used is the sufficient solution quality concept represented by the cost J. The value of the cost J depends on applications and can be set from design specifications, component tolerances, and so on.

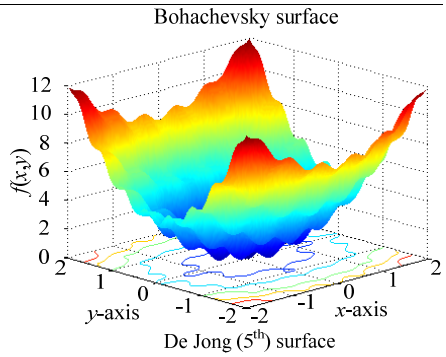
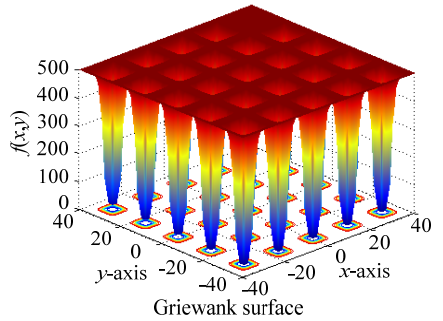
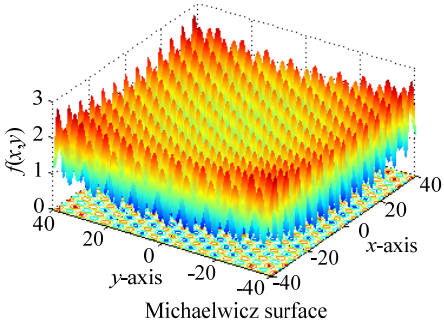
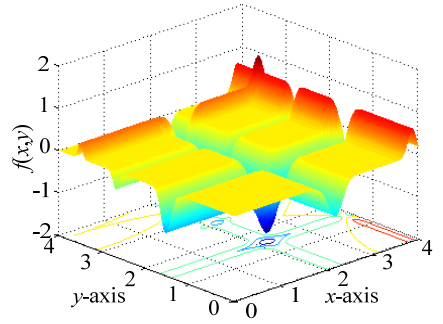
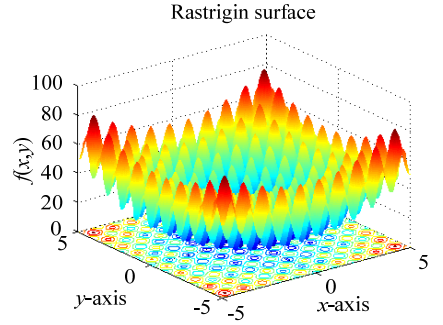
According to CS algorithms, exploration and exploitation properties can complement each other and make the CS most powerful. Algorithms of the CS are elaborately described for the readers to follow and general enough for applications to various optimization problems.

3. Performance Evaluation

This section presents the performance comparison studies among genetic algorithm (GA), particle swarm optimization (PSO), adaptive tabu search (ATS), and current search (CS). Algorithms of the GA, the PSO, and the ATS are omitted. Readers may refer to good sources [4,5] for the GA, [7] for the PSO, and [18,19] for the ATS, respectively. GA, PSO, ATS, and CS are tested against eight well-known unconstrained optimization problems [23] including Bohachevsky function (BF), the fifth function of De Jong (DJF), Griewank function (GF), Michaelwicz function (MF), Rastrigin function (RF), Salomon function (SalF), Schwefel function (SchF), and Shekel's fox-holes function (SF). **Table 1** summarizes these test functions in which J_{min} is the minimum values of objective functions required to terminate the search.

Algorithms of PSO, ATS, and CS were coded by MATLAB running on Intel Core2 Duo 2.0 GHz 3 Gbytes DDR-RAM computer, while GA is used from the MAT-

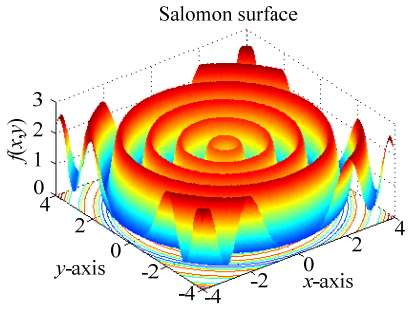
Table 1. Summary of the unconstrained optimization problems.

Test functions	Equations, optimum solutions, and search spaces	3D Surfaces
BF	$f(x,y)=x^2+2y^2-0.3\cos(3\pi x)-0.4\cos(4\pi y)+0.7$ global minimum located at $x=0, y=0$ with $f(x,y)=0$, search space: $x\in[-2,2], y\in[-2,2]$, and $J_{\min}\le 1\times 10^{-6}$	
DJF	$f(x,y)=\left[1/500+\sum_{j=1}^{25}\left\{1/\left(j+\left(x-a_{1j}\right)^6+\left(y-a_{2j}\right)^6\right)\right\}\right]^{-1}$, where $a_{ij}=\begin{pmatrix}-32 & -16 & 0 & 16 & 32 & -32 & \cdots & 0 & 16 & 32\\ -32 & -32 & -32 & -32 & -32 & -16 & \cdots & 32 & 32 & 32\end{pmatrix}$ global minimum located at $x=-32, y=-32$, with $f(x,y)=0.9980$, search space: $x\in[-40,40], y\in[-40,40]$, and $J_{\min}\le 0.9990$	
GF	$f(x,y)=1+\{(1/4000)(x^2+y^2)\}-\{\cos(x)\cos(y/\sqrt{2})\}$ global minimum located at $x=0, y=0$ with $f(x,y)=0$, search space: $x\in[-40,40], y\in[-40,40]$, and $J_{\min}\le 1\times 10^{-6}$	
MF	$f(x,y)=-\sin(x)\sin^{20}(x^2/\pi)-\sin(y)\sin^{20}(2y^2/\pi)$ global minimum located at $x=2.20319, y=1.57049$ with $f(x,y)=-1.8013$, search space: $x\in[0,4], y\in[0,4]$, and $J_{\min}\le -1.80129$	
RF	$f(x,y)=x^2+y^2-10\cos(2\pi x)-10\cos(2\pi y)+20$ global minimum located at $x=0, y=0$ with $f(x,y)=0$, search space: $x\in[-5,5], y\in[-5,5]$, and $J_{\min}\le 1\times 10^{-6}$	

Continued

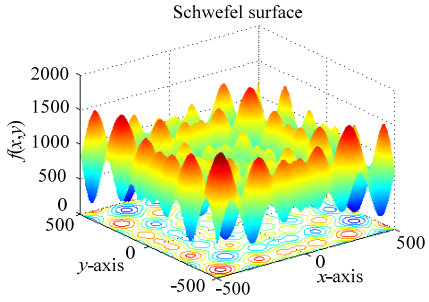
SalF

$f(x,y)=1-\cos\left(2\pi\sqrt{x^2+y^2}\right)+0.1\sqrt{x^2+y^2}$ global minimum located at $x=0$, $y=0$ with $f(x, y)=0$, search space: $x\in[-4,4], y\in[-4,4]$, and $J_{\min}\le 1\times 10^{-6}$



SchF

$f(x,y)=837.9658-\left\{x\sin\left(\sqrt{x}\right)+y\sin\left(\sqrt{y}\right)\right\}$ global minimum located at $x=420.9687$, $y=420.9687$ with $f(x, y)=0$, search space: $x\in[-500,500], y\in[-500,500]$, and $J_{\min}\le 1\times 10^{-6}$



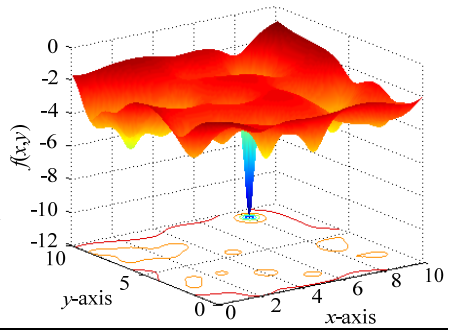
SF

$f(x,y)=-\sum_{j=1}^{30}\left[1\Big/\left\{c_j+\left(x-a_{1j}\right)^2+\left(y-a_{2j}\right)^2\right\}\right]$,

where $a_{ij}=\begin{pmatrix}9.6810 & 9.4000 & 8.0250 & \cdots & 9.4960 & 4.1380\\ 0.6670 & 2.0410 & 9.1520 & \cdots & 4.8300 & 2.5620\end{pmatrix}$,

$c_j=(0.8060 \quad 0.5170 \quad 0.1000 \quad 0.9080 \quad \cdots \quad 0.6080 \quad 0.3260)$

global minimum located at $x=8.0241$, $y=9.1465$ with $f(x, y)=-12.1190$, search space: $x\in[0,10], y\in[0,10]$, and $J_{\min}\le -12.1189$



LAB-GA Toolbox [5]. Each of these algorithms performs search on each test function for 50 trials. Each search trial begins the search with different initial solutions, while search parameters are kept the same for all trials. This approach is commonly referred to as multiple-points-single-strategy (MPSS) in metaheuristic contexts. Search parameter settings for the GA follow MATLAB-GA Toolbox [5], for the PSO follow [7], for the ATS follow [18,19]. **Tables 2** and **3** summarize the search parameters of the ATS and the CS respectively.

The averages results over 50 trials are summarized in **Table 4**. There are two groups of data, *i.e.*, average number of objective function evaluations and average search time. Referring to the search time data in **Table 4**, the CS spends search time of 58.27% less than the GA does, 47.79% less than the ATS does, and 47.72% less than the PSO does as average. Referring to the number of objective function evaluations in **Table 4**, the CS evaluates the objective function of 58.26% less than the GA does, 47.49% less than the ATS does, and 47.72% less than the PSO does as, respectively.

Table 5 summarizes the solutions obtained from those algorithms. It can be noticed that the CS outperforms other algorithms and provides solutions with the best quality within the shortest search time. This outstanding performance of the CS is achieved due to its explorative and exploitative characteristics. **Figure 3** represents the CS movements over the search spaces of test functions. It was found that the large areas of all search spaces are explored by the explorative characteristic of the CS. In **Figure 3**, the CS can provide a high-quality solution rapidly because the elite search path can be provided. The search rapidly focuses the search to the solution with its exploitative characteristic.

The CS metaheuristics has been applied to a constrained parametric search problem, that is, design of the optimum PID controller for the dc motor speed control which is widely used in industries. In the next section, experimental setup and implementation results are illustrated.

Table 2. ATS parameters.

Test functions	*N*	*R*	*BT* (*n_re*_max)	$Count_{max}$	*AR*	
					Stage I	*Stage II*
BF	50	0.1	5	10,000	$J < 1 \times 10^{-2} \rightarrow R = 0.01$	$J < 1 \times 10^{-4} \rightarrow R = 0.001$
DJF	100	5.0	5	10,000	$J < 1 \times 10^{2} \rightarrow R = 1.00$	$J < 10 \rightarrow R = 0.50$
GF	100	0.5	5	10,000	$J < 1 \times 10^{-2} \rightarrow R = 0.01$	$J < 1 \times 10^{-4} \rightarrow R = 0.001$
MF	50	2.0	5	10,000	$J < 1 \rightarrow R = 0.01$	$J < 0 \rightarrow R = 0.001$
RF	100	0.5	5	10,000	$J < 1 \times 10^{-3} \rightarrow R = 0.001$	$J < 1 \times 10^{-4} \rightarrow R = 0.0001$
SalF	100	0.5	5	10,000	$J < 1 \times 10^{-3} \rightarrow R = 0.001$	$J < 1 \times 10^{-4} \rightarrow R = 0.0001$
SchF	100	10.0	5	10,000	$J < 1 \times 10^{-2} \rightarrow R = 0.001$	$J < 1 \times 10^{-4} \rightarrow R = 0.0001$
SF	100	1.5	5	10,000	$J < -2 \rightarrow R = 1.00$	$J < -5 \rightarrow R = 0.50$

Notes: *N* = number of neighborhood members, *R* = search radius, *BT* (back-tracking), *n_re*_max = maximum allowance of solution cycling before invoking *BT*, $Count_{max}$ = maximum search iterations, *AR* (adaptive search radius) [18,19].

Table 3. CS parameters.

Test functions	*n*	*R*	*N*	I_{max}	*R-adjustment*	
					Stage I	*Stage II*
BF	50	0.1	50	1000	$J < 1 \times 10^{-2} \rightarrow R = 0.01$	$J < 1 \times 10^{-4} \rightarrow R = 0.001$
DJF	50	5.0	100	1000	$J < 1 \times 10^{2} \rightarrow R = 1.00$	$J < 10 \rightarrow R = 0.50$
GF	100	0.5	100	1000	$J < 1 \times 10^{-2} \rightarrow R = 0.01$	$J < 1 \times 10^{-4} \rightarrow R = 0.001$
MF	50	2.0	50	1000	$J < 1 \rightarrow R = 0.01$	$J < 0 \rightarrow R = 0.001$
RF	100	0.5	100	1000	$J < 1 \times 10^{-3} \rightarrow R = 0.001$	$J < 1 \times 10^{-4} \rightarrow R = 0.0001$
SalF	100	0.5	200	1000	$J < 1 \times 10^{-3} \rightarrow R = 0.001$	$J < 1 \times 10^{-4} \rightarrow R = 0.0001$
SchF	100	10.0	100	1000	$J < 1 \times 10^{-2} \rightarrow R = 0.001$	$J < 1 \times 10^{-4} \rightarrow R = 0.0001$
SF	100	1.5	50	1000	$J < -2 \rightarrow R = 1.00$	$J < -5 \rightarrow R = 0.50$

Notes: *n* = number of neighborhood members, *R* = search radius, *N* = search (current) paths, I_{max} = maximum search iterations.

Table 4. Average results over 50 trials.

Test functions	Average number of function evaluations				Average search time (seconds)			
	GA	PSO	ATS	CS	GA	PSO	ATS	CS
BF	480,100	465,100	424,123	200,300	6.52	6.32	5.76	2.72
DJF	358,400	352,520	375,485	114,222	20.50	18.75	19.67	6.07
GF	902,150	780,464	752,260	501,500	11.67	10.10	9.74	6.49
MF	200,375	145,680	126,450	50,100	3.01	2.18	1.89	0.75
RF	495,560	474,725	468,526	240,150	9.80	9.39	9.27	4.75
SalF	913,250	610,500	585,375	454,100	17.46	11.67	11.19	8.68
SchF	405,345	248,654	286,140	150,250	5.69	3.49	4.12	2.11
SF	685,450	545,120	601,150	320,300	12.07	9.60	10.59	5.64

Table 5. Results obtained from GA, PSO, ATS, and CS.

Test functions	Results	GA	PSO	ATS	CS
BF	Min	2.1250×10^{-8}	1.3651×10^{-9}	1.4563×10^{-9}	1.9790×10^{-10}
	Max	3.4613×10^{-7}	6.4197×10^{-8}	4.0219×10^{-8}	1.5602×10^{-9}
	Average	1.0312×10^{-7}	5.7490×10^{-9}	6.0498×10^{-9}	3.0564×10^{-10}
	Std.	1.0140×10^{-7}	5.6873×10^{-9}	6.1012×10^{-9}	1.1102×10^{-10}
DJF	Min	0.9980	0.9980	0.9980	0.9980
	Max	0.9989	0.9985	0.9982	0.9981
	Average	0.9984	0.9983	0.9981	0.9980
	Std.	0.0003	0.0002	0.0001	0.0001
GF	Min	4.0248×10^{-8}	2.4597×10^{-9}	2.8471×10^{-9}	1.0142×10^{-10}
	Max	2.0527×10^{-7}	4.6894×10^{-8}	1.8210×10^{-8}	2.4992×10^{-9}
	Average	6.0141×10^{-7}	7.0315×10^{-9}	8.4010×10^{-9}	4.6617×10^{-10}
	Std.	5.9414×10^{-7}	6.5143×10^{-9}	4.9834×10^{-9}	1.4652×10^{-10}
MF	Min	–1.8013	–1.8013	–1.8013	–1.8013
	Max	–1.8012	–1.8012	–1.8012	–1.8012
	Average	–1.8012	–1.8012	–1.8013	–1.8013
	Std.	0.0002	0.0002	0.0001	0.0001
RF	Min	3.9830×10^{-8}	4.0539×10^{-9}	4.0143×10^{-9}	1.2322×10^{-10}
	Max	4.8527×10^{-7}	3.3127×10^{-8}	1.0381×10^{-8}	1.0103×10^{-9}
	Average	1.4291×10^{-7}	6.9973×10^{-9}	6.3967×10^{-9}	3.8730×10^{-10}
	Std.	1.6608×10^{-7}	2.9978×10^{-9}	5.9910×10^{-9}	1.9195×10^{-10}
SalF	Min	4.0462×10^{-8}	5.6563×10^{-9}	6.4251×10^{-9}	1.0102×10^{-9}
	Max	6.0985×10^{-7}	4.1027×10^{-8}	4.2879×10^{-8}	2.4510×10^{-8}
	Average	8.0302×10^{-7}	7.9410×10^{-9}	8.0693×10^{-9}	4.0301×10^{-9}
	Std.	7.9878×10^{-7}	6.9901×10^{-9}	7.9902×10^{-9}	4.1207×10^{-9}
SchF	Min	1.1374×10^{-7}	5.1503×10^{-8}	4.0470×10^{-8}	1.8547×10^{-8}
	Max	8.1302×10^{-7}	3.4532×10^{-7}	2.0604×10^{-7}	2.0021×10^{-7}
	Average	4.4531×10^{-7}	8.0831×10^{-8}	6.8876×10^{-8}	2.5654×10^{-8}
	Std.	5.8523×10^{-7}	7.6574×10^{-8}	7.0130×10^{-8}	1.9983×10^{-8}
SF	Min	–12.1190	–12.1190	–12.1190	–12.1190
	Max	–12.1189	–12.1189	–12.1189	–12.1189
	Average	–12.1189	–12.1189	–12.1189	–12.1190
	Std.	0.0002	0.0002	0.0002	0.0001

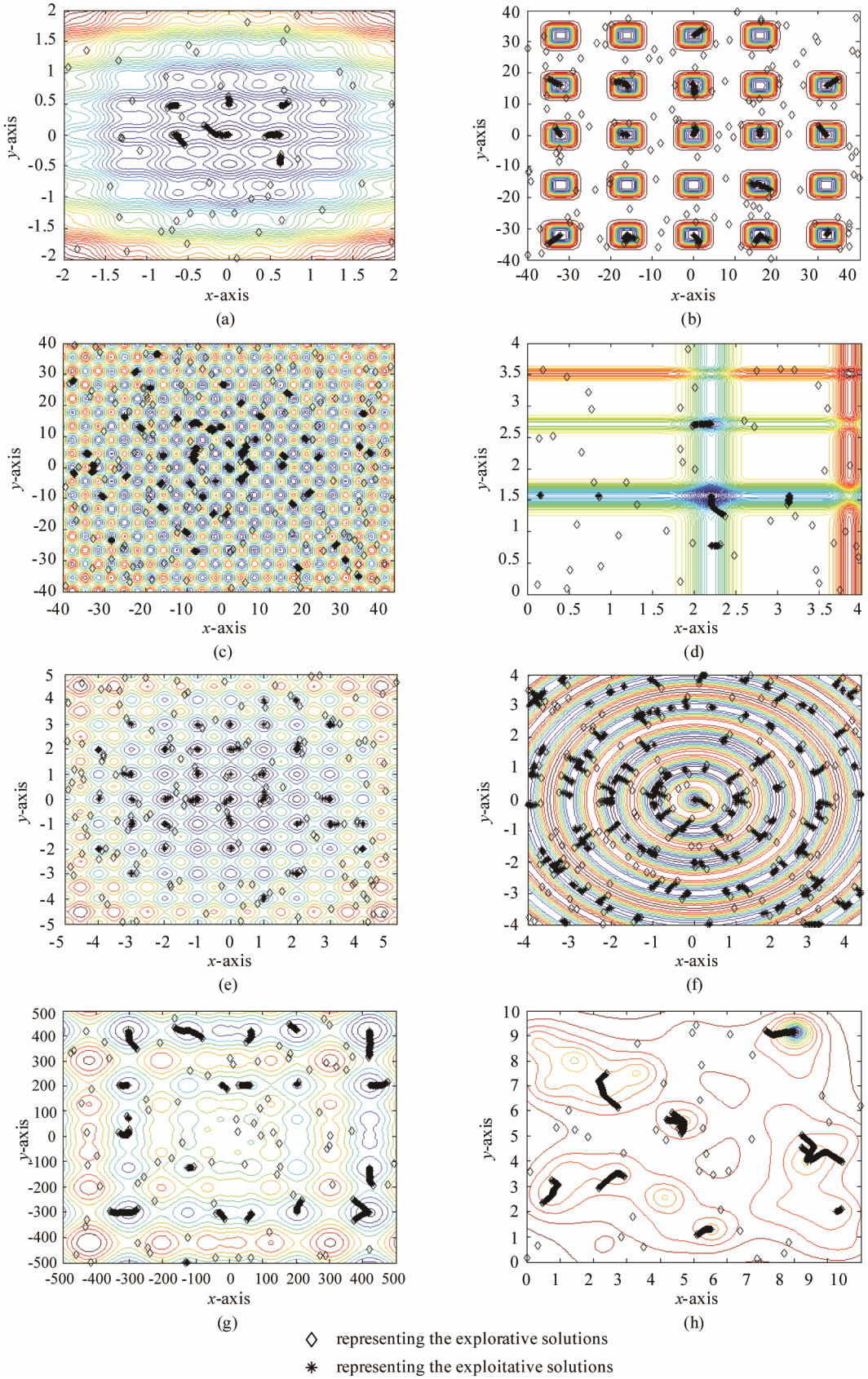


Figure 3. CS movements: (a) BF; (b) DJF; (c) GF; (d) MF; (e) RF; (f) SalF; (g) SchF; and (h) SF.

4. DC Motor Speed Control Design

In this section, the CS is applied to design an optimum PID controller for the dc motor speed control system. This section can be divided into four parts, *i.e.*, mathematical model, model identification, design of PID controller by the CS, and experimental results, respectively. Details are described as follows.

4.1. Mathematical Model

The schematic diagram of armature-controlled dc motor can be represented in **Figure 4** [24], where R_a is an armature-winding resistance, L_a is an armature-winding inductance, R_f is a field-winding resistance, L_f is a field-winding inductance, J is a moment of inertia, B is a viscous-friction coefficient, $e_a(t)$ is an applied armature voltage, $i_a(t)$ is an armature current, $e_f(t)$ is a field voltage, $i_f(t)$ is a field current, $e_b(t)$ is a back emf (electromotive force) voltage, $T(t)$ is a motor torque, $\theta(t)$ is an angular displacement, and $\omega(t)$ is an angular velocity (speed), respectively.

Mathematical model of armature-controlled dc motor in term of the transfer function can be formulated by the differential equations and Laplace transform. The transfer function of armature-controlled dc motor speed control is stated in Equation (1) [24], where K_t is a torque constant and K_b is a back emf constant.

$$\frac{\Omega(s)}{E_a(s)} = \frac{K_t}{JL_a s^2 + (BL_a + JR_a)s + BR_a + K_t K_b} \quad (1)$$

4.2. Model Identification

The controlled plant consists of dc motor (LEYBOLD-DIDACTIC GMBH, Type 731-91, 0.3 kW, 220 V, 2.2 A, 2000 rpm), an actuator (SCR full-wave controlled rectifier), a speed sensor (tachogenerator LEYBOLD, Type 731-09), and a lowpass filter circuit. Model identification process starts with testing the controlled plant by step input excitation. A step-transient response test was conducted for the identification of the motor model at the 1860 rpm operating point as shown in **Figure 5**. It was found that the controlled plant gives the rise time of 2.50 sec, steady-state level of 4 V (1860 rpm), and without overshoot. The identification using MATLAB and System Identification Toolbox [25] provides the normalized 3rd-order plant model expressed by Equation (2). **Figure 6** depicts the simulation step response obtained by the model. The simulation result agrees very well with the experimental one. Therefore, the identified model in Equation (2) is very good representations of the controlled plant.

$$G_p(s)\big|_{\text{plant}} = \frac{1}{1\times10^{-5}s^3 + 0.04534s^2 + 0.5524s + 1} \quad (2)$$

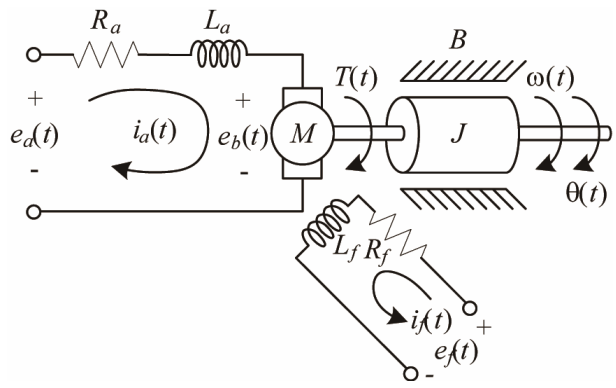


Figure 4. Schematic diagram of dc motor.

4.3. Design of PID Controller by CS

The CS-based PID controller design for the dc motor speed control system can be performed by the block diagram representation as shown in **Figure 7**. The theoretical function of the PID controller is expressed in Equation (3), where K_p is the proportional gain, K_i, is the integral gain, and K_d is the derivative gain, respectively. The

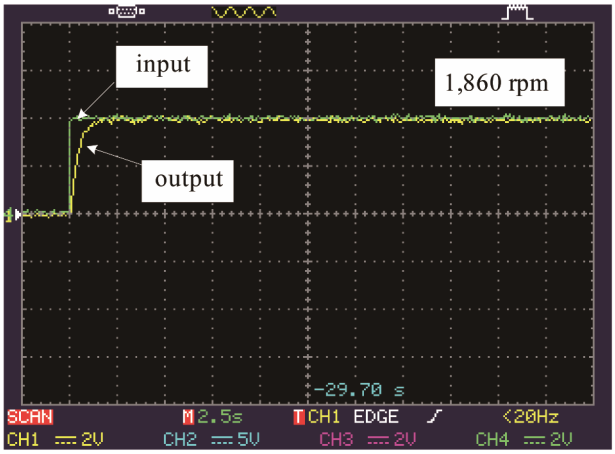


Figure 5. DC motor step response.

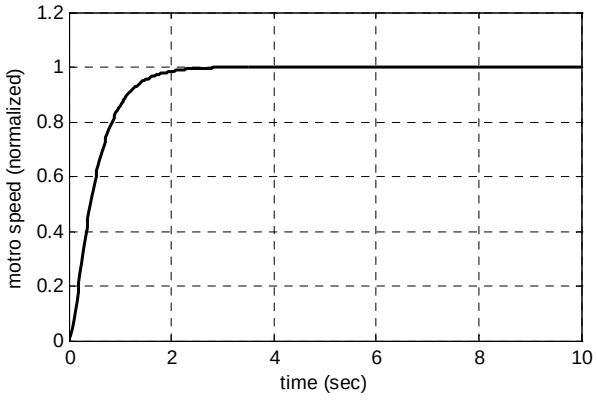


Figure 6. Step response of the plant model.

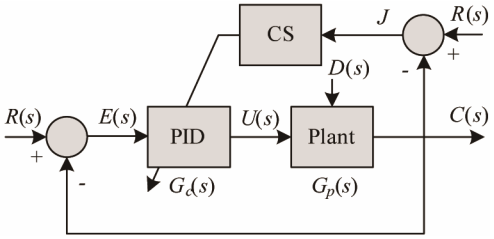


Figure 7. CS-based PID design.

objective function, J, the sum of absolute errors between $R(s)$ and $C(s)$ as stated in Equation (4), will be fed back to the CS tuning block. J is minimized to find the optimum PID controller's parameters, *i.e.*, K_p, K_i, and K_d, giving satisfactory responses.

$$G_c(s)\big|_{\text{PID}} = K_p + \frac{K_i}{s} + K_d s \tag{3}$$

$$J = \sum_{t=0}^{T} \left| r(t) - c(t) \right| \tag{4}$$

$$\begin{aligned} \min \quad & J = \sum_{t=0}^{T} \left| r(t) - c(t) \right| \\ \text{subject to} \quad & t_r \le 1.00 \text{ sec.}, \\ & M_p \le 10.00\%, \\ & t_s \le 2.50 \text{ sec.}, \\ & e_{ss} \le 1 \times 10^{-5}, \\ & 0.1 \le K_p \le 10, \\ & 0.1 \le K_i \le 10, \\ & 0.0001 \le K_d \le 0.001 \end{aligned} \tag{5}$$

Referring to **Figure 7**, the objective function J is minimized according to inequality constraints in Equation (5), where t_r is rise time, M_p is maximum percent overshoot, t_s is settling time, and e_{ss} is steady state error. The CS search parameters are priory set as follows: the maximum search iteration of each path $I_{\max}$ = 200, the number of initial solutions N = 10, the number of neighborhood members n = 20, and the search radius R = 20% of search spaces. Algorithms of the CS were coded by MATLAB running on Intel Core2 Duo 2.0 GHz 3 Gbytes DDR-RAM computer. The tests were conducted 20 trial runs to obtain the optimum PID controller for the dc motor speed control system.

After the search process stopped, the CS successfully provides the optimum PID controller for the motor control system as stated in Equation (6), where K_p = 1.6501, K_i = 4.0986, and K_d = 0.0049, within 216.3504 sec of average search time consumed. The step input and the step disturbance responses of controlled system are depicted in **Figure 8**. As results, the system without PID controller gives t_r = 2.50 sec. and cannot be recovered once step disturbance of 0.3 (30% of unit step) is applied

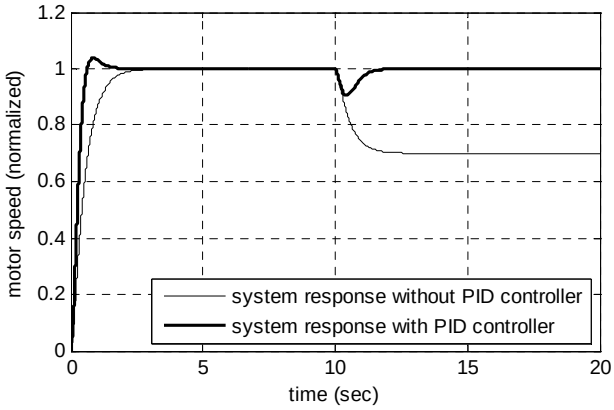


Figure 8. Response of the controlled system.

at the 10th second. This causes the e_{ss} of 0.3 at steady-state response. For the system with PID controller obtained by the CS, the step input response gives t_r = 0.8506 sec, M_p = 4.04%, t_s = 2.0104 sec, and e_{ss} = 0.0, while the step disturbance response, once the step disturbance of 0.3 is applied, provides the recovering time t_{re} = 1.8542 sec, M_p = 8.24%, and e_{ss} = 0.0. It was found that the optimum PID controller can be successfully achieved by the CS according to inequality constraints stated in Equation (5).

$$G_c(s)\big|_{\text{PID}} = 1.6501 + \frac{4.0986}{s} + 0.0049s \tag{6}$$

4.4. Experimental Results

To obtain the experimental results, a closed loop dc motor speed control system with PID controller is a necessary test bed. The diagram in **Figure 9** represents the experimental setup. The optimum PID controller obtained by the CS is realized by electronic circuits (op-amp LM335 and RC network) to ensure real-time operation. Driving the motor requires a full-wave controlled rectifier controlled by IC-TCA785. Signal conditioning circuits including lowpass filter and zero-span circuits are also conducted. The experimental setup installed in the laboratory is shown in **Figure 10**, while the experimental results are depicted in **Figure 11**.

Referring to **Figure 11**, it was found that the controlled dc motor speed control system with PID controller designed by the CS can provide the very satisfactory results in both command following and load regulating (disturbance rejection) modes. In command following mode, the system response gives t_r = 0.98 sec, M_p = 5.01%, t_s = 2.42 sec, and e_{ss} = 0.0, while in disturbance rejection mode once the step disturbance of 0.3 is applied the system provides the recovering time t_{re} = 1.55 sec, M_p = 7.85%, and e_{ss} = 0.0. The system responses obtained from the test bed agree very well with those obtained from the model simulation. Very good agreement be-

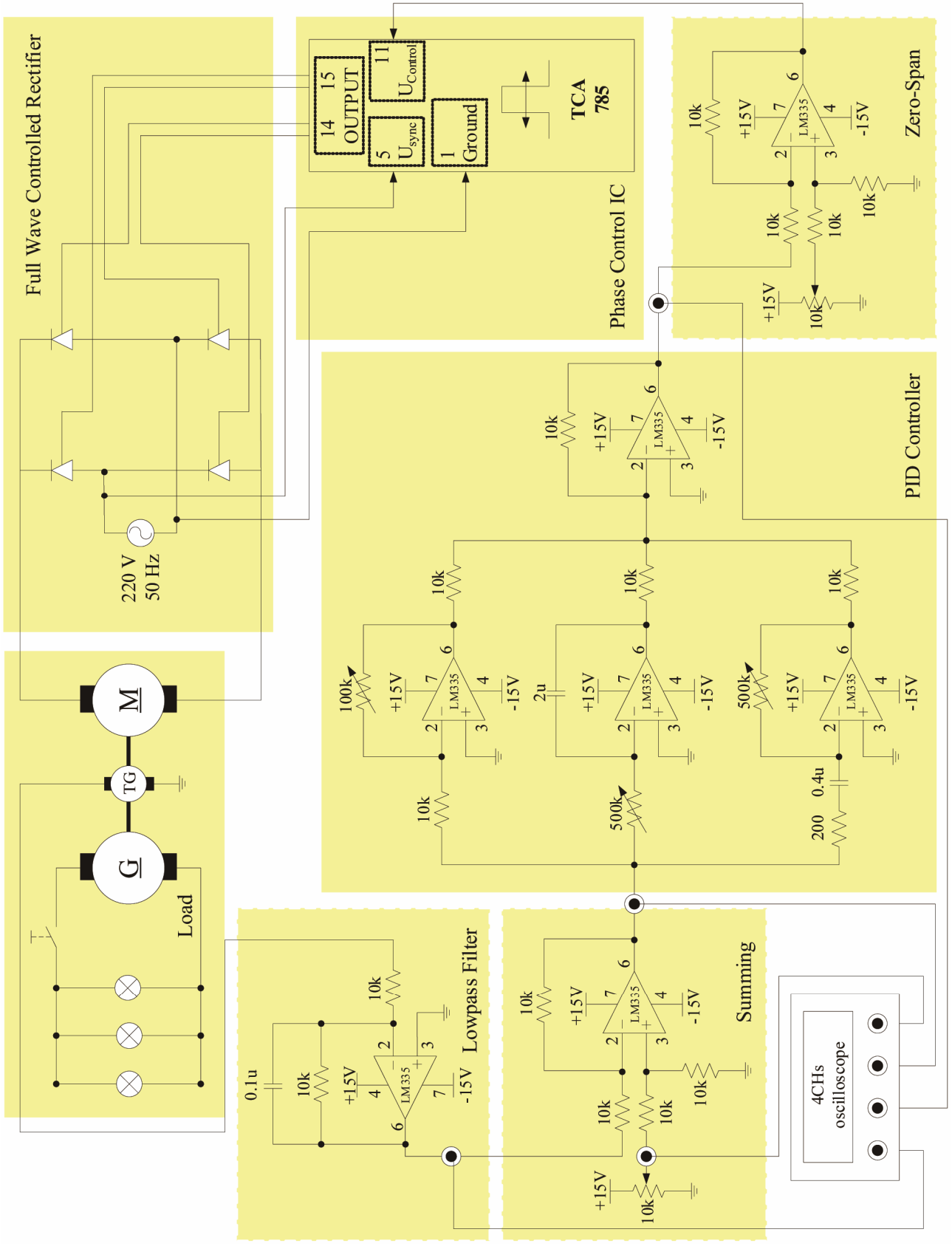


Figure 9. Circuit diagram representing the experimental setup.

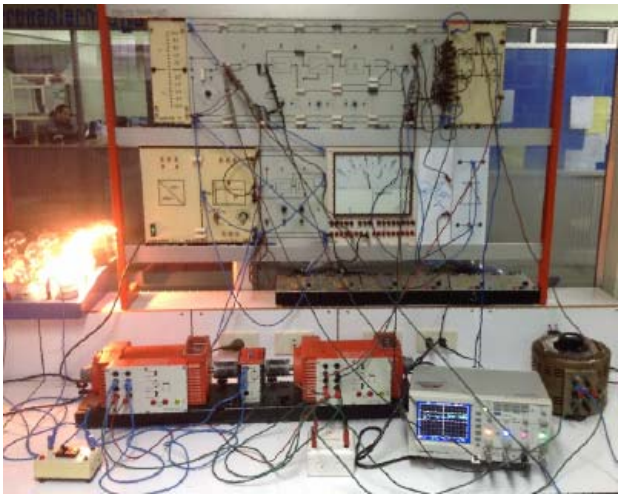


Figure 10. Experimental setup.

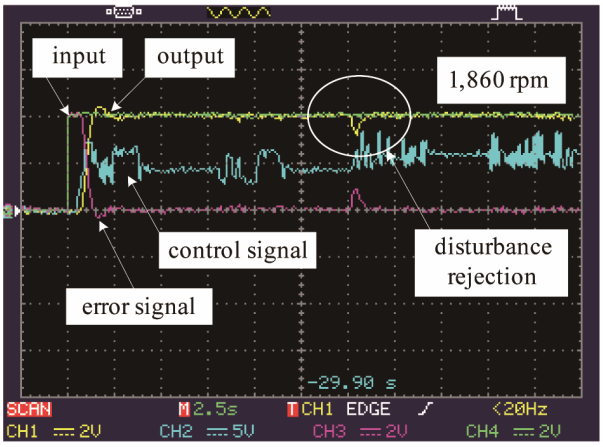


Figure 11. Experimental results.

tween the simulation results and the experiment ones can be observed from **Figures 8** and **11**.

5. Conclusion

The current search (CS) metaheuristics has been proposed in this paper. The CS algorithms have been conceptualized from the electric current flowing through electric networks for optimization problems. Algorithms of the CS have been elaborately described. Performance evaluation of the CS has been assessed by testing against eight well-known optimization problems. With two powerful strategies, *i.e.*, exploration and exploitation, the proposed CS has provided superior search performances to GA, PSO, and ATS. The CS has been applied to design the optimum PID controller for the dc motor speed control system. An experimental bed of a closed loop dc motor speed control system was constructed at the laboratory. As results, the optimum PID controller can be obtained by the CS. Very good agreement between the simulation and experimental results has been also proposed. This can be concluded that the optimum PID controller for the dc motor speed control system can be successfully obtained by the proposed CS metaheuristics. For the future trends of this work, the convergence proofs of the CS metaheuristics will be proposed. Moreover, the CS is still needed to be applied to various realworld engineering optimization among discrete, combinatorial, as well as multiobjective Pareto optimization problems especially in control engineering domain including model identification, system stabilization, and control synthesis.

REFERENCES

[1] F. Glover and G. A. Kochenberger, "Handbook of Metaheuristics," Kluwer Academic Publishers, Dordrecht, 2003.

[2] E. G. Talbi, "Metaheuristics Forn Design to Implementation," John Wiley & Sons, Hoboken, 2009. doi:10.1002/9780470496916

[3] X. S. Yang, "Nature-Inspired Metaheuristic Algorithms," Luniver Press, 2010.

[4] D. E. Goldberg, "Genetic Algorithms in Search Optimization and Machine Learning," Addison Wesley Publishers, Boston, 1989.

[5] MathWorks, "Genetic Algorithm and Direct Search Toolbox: For Use with MATLAB," User's Guide, Version 1, MathWorks, Natick, Mass, 2005.

[6] M. Dorigo and T. Stützle, "Ant Colony Optimization," MIT Press, Cambridge, 2004. doi:10.1007/b99492

[7] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *IEEE Proceedings of the International Conference on Neural Networks*, Vol. 4, 1995, pp. 1942-1948. doi:10.1109/ICNN.1995.488968

[8] Z. W. Geem, "Recent Advance in Harmony Search Algorithm," Studies in Computational Intelligence, Springer, Berlin, 2010. doi:10.1007/978-3-642-04317-8

[9] X. S. Yang, "Firefly Algorithms for Multimodal Optimization, Stochastic Algorithms," Foundations and Applications SAGA 2009, *Lecture Notes in Computer Sciences*, Vol. 5792, 2009, pp. 169-178. doi:10.1007/978-3-642-04944-6_14

[10] R. Oftadeh, M. J. Mahjoob and M. Shariatpanahi, "A Novel Mata-Heuristic Optimization Algorithm Inspired by Group Hunting of Animals: Hunting Search," *Computers and Mathematics with Applications*, Vol. 60, No. 7, 2010, pp. 2087-2098. doi:10.1016/j.camwa.2010.07.049

[11] X. S. Yang and S. Deb, "Cuckoo Search via Lévy Flights," In: *Proceedings of the World Congress on Nature and Biologically Inspired Computing*, IEEE Publications, 2009, pp. 210-214.

[12] X. S. Yang and S. Deb, "Engineering Optimization by Cuckoo Search," *International Journal of Mathematical Modeling and Numerical Optimization*, Vol. 1, No. 4, 2010, pp. 330-343. doi:10.1504/IJMMNO.2010.035430

[13] X. S. Yang, "A New Metaheuristic Bat-Inspired Algorithm," In: J. R. Gonzalez, *et al.*, Eds., *Nature Inspired*

Cooperative Strategies for Optimization (*NISCO* 2010), Springer, Berlin, 2010, pp. 65-74.

[14] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, Vol. 220, No. 4598, 1983, pp. 671-680. doi:10.1126/science.220.4598.671

[15] P. J. M. van Laarhoven and E. H. L. Aarts, "Simulated Annealing: Theory and Applications," Kluwer Academic Publishers, Dordrecht, 1987. doi:10.1007/978-94-015-7744-1

[16] F. Glover, "Tabu Search—Part I," *ORSA Journal on Computing*, Vol. 1, No. 3, 1989, pp. 190-206.

[17] F. Glover, "Tabu Search—Part II," *ORSA Journal on Computing*, Vol. 2, No. 1, 1990, pp. 4-32. doi:10.1287/ijoc.2.1.4

[18] S. Sujitjorn, T. Kulworawanichpong, D. Puangdownreong and K.-N. Areerak, "Adaptive Tabu Search and Applications in Engineering Design," Frontiers in Artificial Intelligent and Applications, IOS Press, Amsterdam, 2006.

[19] D. Puangdownreong, T. Kulworawanichpong and S. Sujitjorn, "Finite Convergence and Performance Evaluation of Adaptive Tabu Search," In: *Lecture Notes in Artificial Intelligence*, Springer-Verlag, Heidelberg, 2004, pp. 710-717.

[20] A. Sukulin and D. Puangdownreong, "A Novel Meta-Heuristic Optimization Algorithm: Current Search," *Proceedings of the* 11*th WSEAS International Conference on Artificial Intelligence*, *Knowledge Engineering and Data Bases* (*AIKED* '12), Cambridge, 2012, pp. 125-130.

[21] D. Puangdownreong and A. Sukulin, "Obtaining an Optimum PID Controllers for Unstable Systems Using Current Search," *International Journal of Systems Engineering*, *Applications & Development*, Vol. 2, No. 6, 2012, pp. 188-195.

[22] D. Puangdownreong, "Application of Current Search to Optimum PIDA Controller Design," *Intelligent Control and Automation*, Vol. 3, No. 4, 2012, pp. 303-312.

[23] M. M. Ali, C. Khompatraporn and Z. B. Zabinsky, "A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test Problems," *Journal of Global Optimization*, Vol. 31, No. 4, 2005, pp. 635-672. doi:10.1007/s10898-004-9972-2

[24] K. Ogata, "Modern Control Engineering," Prentice Hall, New Jersey, 2010.

[25] MathWorks, "System Identification Toolbox," User's Guide, Version 7.2, 2008.

