This article has been accepted for publication in a future proceedings of this conference, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/ICAC65379.2025.11196536, 2025 30th International Conference on Automation and Computing (ICAC)

A Novel Particle Swarm Optimizer with Randomly Occurring Uncertainty

Linwei Chen[1], Zidong Wang[1], Weibo Liu[1], and Xiaohui Liu[1]
[1] Department of Computer Science, Brunel University London, Uxbridge, Middlesex, UB8 3PH, United Kingdom.
E-mails:{Linwei.Chen, Zidong.Wang, Weibo.Liu2, Xiaohui.Liu}@brunel.ac.uk

***Abstract*—Particle Swarm Optimization (PSO) has been widely applied due to its simplicity and effectiveness in solving optimization problems. However, PSO often suffers from premature convergence and stagnation in local optima, especially in complex search spaces. This paper proposes a novel PSO algorithm by incorporating stochastic perturbations into the velocity updating mechanism. Specifically, the acceleration coefficients are perturbed by the introduced randomly occurring uncertainty to improve the search ability of the entire swarm. Experimental results on representative CEC benchmark functions demonstrate that the proposed algorithm outperforms several existing PSO variants.**

***Index Terms*—Particle Swarm Optimization, acceleration coefficients, randomly occurring uncertainty.**

I. Introduction

Particle Swarm Optimization (PSO) is a population-based evolutionary algorithm inspired by social behaviors of organisms such as bird flocking and fish schooling [1]. PSO has been effectively applied to various real-world problems, ranging from robotics to data analytics, including areas such as path planning for robots, robot learning, feature selection, and image segmentation [2]–[5].

Despite its success, PSO suffers from premature convergence, particularly for complex and high-dimensional optimization problems. To address the premature convergence problem, researchers have proposed numerous PSO variants [6]. In [6], the recent developments on PSO algorithms can be broadly categorized into four types: 1) adjusting control parameters; 2) designing novel velocity and position updating strategies; 3) developing new topological structures; and 4) hybridizing PSO with other evolutionary algorithms.

In PSO, the control parameters (including the inertia weight and the acceleration coefficients) play a significant role in discovering potential solutions in the search space [7]. The inertia weight reflects the influence of the particle's previous velocity on its current velocity, representing the particle's trust in its current motion. The inertia weight enables particles to maintain momentum and encourages particles to explore the whole search space. The acceleration coefficients guide the particle's trajectory toward better solutions. The cognitive acceleration coefficient represents the weight given to the particle's own best-known position, driving it towards its personal experience. While the social acceleration coefficient reflects the influence of the swarm's global best position, guiding the particle based on the experience of the entire group. The control parameters shape the dynamic behavior of particles and directly affect the algorithm's ability to efficiently explore and exploit the search space [7].

Updating the control parameters in PSO, particularly the acceleration coefficients, has received considerable attention over the past few decades [8], [9]. Beyond deterministic schedules, recent studies have introduced randomness (e.g., noise and uncertainty) into the acceleration coefficients with the purpose of enhancing the search ability of the optimizer. Noises typically refer to uncontrollable random errors in measured quantities, and uncertainty is a broader concept that encompasses model-driven variability, which can be deliberately introduced to influence algorithmic behavior. For example, the Randomized PSO (RPSO) has been proposed in [10], which perturbs acceleration coefficients using the Gaussian noise. The introduction of the Gaussian noise contributes to the enhancement of the escaping ability of the entire swarm from local optima. However, the use of unbounded Gaussian noise may cause particles to move beyond the search space, potentially resulting in inconsistent adjustment of the velocity, thereby causing the premature convergence issue. In this situation, it becomes a seemingly natural idea to introduce the uncertainty into the optimizer due to its bounded property.

Motivated by the above discussions, this paper proposes a PSO with Randomly Occurring Uncertainty (PSORU) algorithm. The Cosine function is employed to model the uncertainty, which provides an effective perturbation on the acceleration coefficients of the optimizer. The advantages of using the randomly occurring uncertainty modeled by the Cosine function can be summarized: 1) the perturbation of the acceleration coefficients using the uncertainty tunes the optimizer's dynamic characteristics, which would enhance the search performance of the optimizer with a higher possibility of escaping from the local optima comparing with the conventional PSO algorithm; and 2) the Cosine function modeled uncertainty is made to occur with a relatively small provability, thereby contributing to a proper balance between convergence and exploration. The main contributions of this paper can be summarized as follows:

1) A novel PSORU algorithm is proposed by introducing the randomly occurring uncertainty modeled by the Cosine function into the velocity updating scheme to perturb the acceleration coefficients, thereby improving the optimizer's capability of escaping from local optima.
2) The effectiveness of the PSORU algorithm is comprehensively evaluated on the selected CEC2017 bench-

This article has been accepted for publication in a future proceedings of this conference, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/ICAC65379.2025.11196536, 2025 30th International Conference on Automation and Computing (ICAC)

mark functions. Results demonstrate the superiority of the PSORU algorithm over several well-known PSO algorithms.

The rest of this paper is organized as follows. In Section II, the proposed PSORU algorithm is explained in detail. Experimental results, parameter settings, and discussions are presented in Section III. Finally, conclusions and future directions are drawn in Section IV.

II. Methodology

A. Motivation

In PSO, control parameters critically balance the search status of the swarm between exploration and exploitation [6]. The inertia weight determines the trade-off between global and local search tendencies, while the acceleration coefficients guide particles toward their personal best positions and the global best position. Effective tuning of the control parameters is essential to maintain the swarm diversity and alleviate premature convergence, particularly in complex optimization problems.

Many existing PSO variants rely on static values or deterministic scheduling of control parameters. Recent developments in PSO have shown that introducing randomness into control parameters could effectively enhance the population diversity and improve the search performance of the optimizer [6]. Motivated by the success of RPSO [10], a seemingly natural idea is to introduce the uncertainty into the velocity updating process of PSO with the hope of improving the search performance of the optimizer. The Gaussian white noise (GWN) used in RPSO is unbounded and may cause instability in particle movement. In contrast, bounded and smooth uncertainty mechanisms can provide more stable perturbations and enable more consistent exploration of the problem space. Furthermore, when such uncertainty is activated stochastically, it offers better control over the optimization dynamics. To be specific, the uncertainty is employed to perturb the time-varying acceleration coefficients, which are made to occur randomly with a small probability, thereby contributing to a proper balance between the convergence and the diversity. Uncertainty can be modeled by using a wide range of mathematical functions. In this paper, the Cosine function is applied to model the uncertainty due to its easy implementation.

B. The PSORU Algorithm

In the PSORU algorithm, the randomly occurring uncertainty introduces randomness into the optimization process, encouraging particles to explore diverse regions of the search space. The velocity and position of the i-th particle at the t-th iteration are updated as follows:

$$\begin{aligned} v_i(t+1) &= \omega(t)v_i(t) + (c_1(t) + \alpha_1\Delta c_1(t))r_1(p_b(t) - x_i(t)) \\ &\quad + (c_2(t) + \alpha_2\Delta c_2(t))r_2(g_b(t) - x_i(t)) \\ x_i(t+1) &= x_i(t) + v_i(t+1) \end{aligned} \tag{1}$$

where $c_1(t)$ and $c_2(t)$ are time-varying acceleration coefficients [9]; $\omega(t)$ is the linearly decreasing inertia weight [7]; r_1 and r_2 are random variables uniformly distributed in $[0, 1]$; $p_b(t)$ denotes the i-th particle personal best position; and $g_b(t)$ represents the global best position of the swarm; α_1 and α_2 are binary activation factors that operate independently; and $\Delta c_1(t)$ and $\Delta c_2(t)$ are two independent uncertainties.

Here, α_1 and α_2 control the presence of perturbations $\Delta c_1(t)$ and $\Delta c_2(t)$, respectively. $\Delta c_1(t)$ and $\Delta c_2(t)$ occur randomly based on the given activation probability, which offers a dynamic control over the balance between exploration and exploitation. In this paper, a popular mathematical function, the Cosine function, is employed to model the uncertainties $\Delta c_1(t)$ and $\Delta c_2(t)$. The inclusion of employing the Cosine function is motivated by the usage of the Sigmoid function in the AWPSO [11]. The Cosine function, denoted by $\cos(x)$, is a fundamental trigonometric function that is smooth and bounded. The smooth and bounded characteristics contribute to a natural regulation of the acceleration coefficients.

Table II-B summarizes the activation status of perturbations $\Delta c_1(t)$ and $\Delta c_2(t)$ under different combinations of the binary indicators α_1 and α_2. Each $\alpha_i \in \{0, 1\}$ determines whether the corresponding perturbation Δc_i is active (i.e., $\alpha_i = 1$) or inactive (i.e., $\alpha_i = 0$).

TABLE I
Activation Status of Perturbations Based on (α_1, α_2)

(α_1, α_2)	$\Delta c_1(t)$ **Status**	$\Delta c_2(t)$ **Status**
(0, 0)	Inactive	Inactive
(0, 1)	Inactive	Active
(1, 0)	Active	Inactive
(1, 1)	Active	Active

The combination of activation factors α_1 and α_2 results in four possible activation states: neither component active, only social active, only cognitive active, or both active. These states influence the contribution of corresponding perturbation terms in Eq. (1).

The activation factors α_1 and α_2, governed by the respective activation probabilities p_1 and p_2, are defined by:

$$\begin{cases} P(\alpha_1 = 1) = p_1, & P(\alpha_1 = 0) = 1 - p_1, \\ P(\alpha_2 = 1) = p_2, & P(\alpha_2 = 0) = 1 - p_2. \end{cases} \tag{2}$$

Note that the value of the activation probabilities p_1 and p_2 is reasonably small, which leads to a proper tradeoff between the population diversity and the convergence performance.

C. The Procedure of the PSORU Algorithm

The proposed PSORU algorithm integrates randomly occurring uncertainty into the velocity update phase. The procedure of the PSORU algorithm is presented in Algorithm 1.

III. Experimental results and discussions

A. Configuration

The experiments were conducted on a PC with an Intel(R) Core(TM) i5-12400F CPU running at 2.50 GHz, and a 64-bit

This article has been accepted for publication in a future proceedings of this conference, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/ICAC65379.2025.11196536, 2025 30th International Conference on Automation and Computing (ICAC)

Algorithm 1: Procedure of the PSORU

Input: Swarm size N; maximum iterations T; linearly decreasing inertia weight $\omega(t)$; cognitive coefficient $c_1(t)$; social coefficient $c_2(t)$; activation probabilities p_1, p_2.
Output: Global best solution $g_b(t)$.
Randomly initialize particle positions $x_i(t)$ and velocities $v_i(t)$, for $i = 1, 2, \ldots, N$;
Evaluate the initial fitness of each particle and set personal bests $p_b(0)$ and global best $g_b(0)$;
for $t = 1$ **to** T **do**
  foreach *particle i in the swarm* **do**
    Evaluate current fitness of particle i;
    if *current fitness is better than* $p_b(t)$ **then**
      Update $p_b(t)$;
    end
  end
  Update global best $g_b(t)$ based on all $p_b(t)$;
  foreach *particle i in the swarm* **do**
    Sample r_1, r_2; Sample α_1, α_2;
    Generate uncertainties $\Delta c_1(t)$ and $\Delta c_2(t)$ using cosine function;
    Update velocity $v_i(t+1)$ using Eq. (1);
    Update position $x_i(t+1)$ based on the updated velocity;
  end
  if *stopping criteria are met* **then**
    break;
  end
end
return $g_b(t)$.

Windows 10 operating system. MATLAB 2023b was used as the programming environment.

TABLE II
EXPERIMENT CONFIGURATIONS

Parameter	Value
Swarm Size (`swarm`)	100
Maximum Iterations (`maxiter`)	1000
Dimensionality (`dim`)	30
Number of Runs (`runnum`)	50
Activation Probability (`p`)	0.3

B. Benchmark functions

To evaluate the effectiveness of the proposed PSORU algorithm, experiments are carried out on ten representative benchmark functions from the CEC2017 benchmark suite [12]. Specifically, the Shifted and Rotated Bent Cigar Function (i.e., F1 in CEC2017) is selected from the unimodal functions. The Shifted and Rotated Expanded Scaffer's F6 Function and the Shifted and Rotated Schwefel's Function (i.e., F5 and F9 in CEC2017) are selected from the simple multimodal functions. Hybrid Function 1, Hybrid Function 5, and Hybrid Function 6 (i.e., F10, F14, and F19 in CEC2017) are selected from the hybrid functions. Composition Function 1, Composition Function 3, Composition Function 6, and Composition Function 9 (i.e., F20, F22, F25, and F28 in CEC2017) are selected from the composition functions.

C. Experimental results

To further assess the performance of the proposed algorithm, comparison experiments are conducted among the proposed PSORU algorithm, the basic PSO algorithm [1], the PSO-TVAC algorithm [9], and the PSO-LDIW algorithm [7].

TABLE III
COMPARISONS OF VARIOUS PSO ALGORITHMS IN 30-DIMENSIONAL SEARCH SPACE

		PSO	PSO-LDIW	PSO-TVAC	PSORU
$f_1(x)$	Minimum	2.73×10^9	5.94×10^2	1.32×10^2	1.03×10^2
	Mean	7.75×10^9	4.38×10^9	1.89×10^8	4.81×10^8
	Std. Dev.	4.40×10^9	3.80×10^9	6.34×10^8	8.40×10^8
$f_2(x)$	Minimum	6.19×10^2	6.00×10^2	6.00×10^2	6.00×10^2
	Mean	6.31×10^2	6.04×10^2	6.01×10^2	6.02×10^2
	Std. Dev.	7.69	3.50	1.38	1.80
$f_3(x)$	Minimum	7.20×10^3	2.80×10^3	3.42×10^3	2.93×10^3
	Mean	8.05×10^3	4.36×10^3	4.62×10^3	4.34×10^3
	Std. Dev.	4.37×10^2	5.90×10^2	6.67×10^2	5.81×10^2
$f_4(x)$	Minimum	1.52×10^3	1.17×10^3	1.15×10^3	1.17×10^3
	Mean	1.80×10^3	1.35×10^3	1.24×10^3	1.27×10^3
	Std. Dev.	2.15×10^2	2.22×10^2	6.72×10^1	5.91×10^1
$f_5(x)$	Minimum	1.14×10^6	4.95×10^3	1.92×10^3	2.12×10^3
	Mean	7.94×10^6	5.14×10^4	3.06×10^4	2.65×10^4
	Std. Dev.	4.42×10^6	4.87×10^4	2.24×10^4	2.29×10^4
$f_6(x)$	Minimum	2.32×10^3	2.09×10^3	2.06×10^3	2.07×10^3
	Mean	2.65×10^3	2.37×10^3	2.36×10^3	2.29×10^3
	Std. Dev.	1.48×10^2	1.59×10^2	1.57×10^2	1.28×10^2
$f_7(x)$	Minimum	2.48×10^3	2.35×10^3	2.34×10^3	2.34×10^3
	Mean	2.53×10^3	2.40×10^3	2.37×10^3	2.37×10^3
	Std. Dev.	2.28×10^1	2.86×10^1	2.12×10^1	2.05×10^1
$f_8(x)$	Minimum	2.84×10^3	2.76×10^3	2.72×10^3	2.72×10^3
	Mean	2.92×10^3	2.84×10^3	2.80×10^3	2.78×10^3
	Std. Dev.	5.19×10^1	5.57×10^1	4.40×10^1	3.47×10^1
$f_9(x)$	Minimum	3.96×10^3	2.80×10^3	4.27×10^3	2.82×10^3
	Mean	6.37×10^3	5.41×10^3	4.91×10^3	4.73×10^3
	Std. Dev.	7.64×10^2	8.43×10^2	4.09×10^2	5.87×10^2
$f_{10}(x)$	Minimum	3.72×10^3	3.43×10^3	3.40×10^3	3.38×10^3
	Mean	4.23×10^3	3.80×10^3	3.80×10^3	3.76×10^3
	Std. Dev.	2.47×10^2	2.52×10^2	1.84×10^2	1.69×10^2

The statistical results of the PSO performance evaluation are presented in Table III. Note that lower fitness values correspond to better performance, since all test functions are formulated as minimization problems. Results show that the proposed PSORU algorithm achieves the smallest minimum fitness values on 5 out of 10 functions (f_1, f_2, f_7, f_8, and f_10), demonstrating its strong exploitation capability. As shown in Table III, the PSORU algorithm also achieves the lowest mean fitness values on 6 benchmark functions (f_3, f_5, f_6, f_7, f_8, f_9, and f_{10}) and the smallest standard deviation on five functions (f_4, f_6, f_7, f_8, and f_{10}) comparing with the selected PSO algorithms. We can draw the conclusion from the statistical analysis of the results that the proposed PSORU algorithm exhibits strong robustness when applied to different types of complex optimization problems.

This article has been accepted for publication in a future proceedings of this conference, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/ICAC65379.2025.11196536, 2025 30th International Conference on Automation and Computing (ICAC)

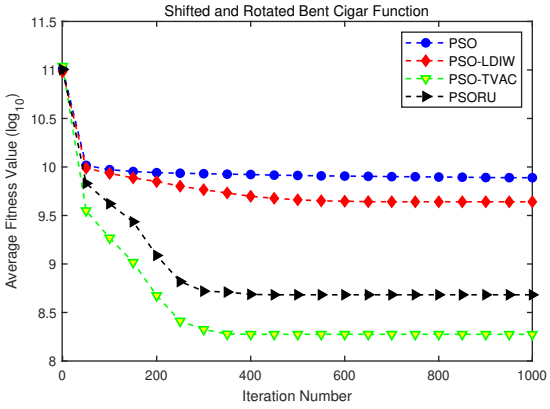

Fig. 1. Convergence plot of the Shifted and Rotated Bent Cigar Function

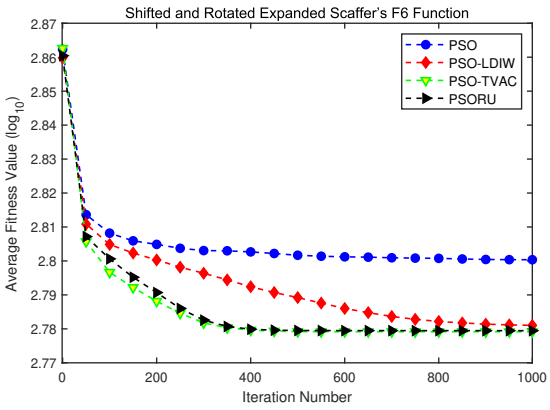

Fig. 2. Convergence plot of the Shifted and Rotated Expanded Scaffer's F6 Function

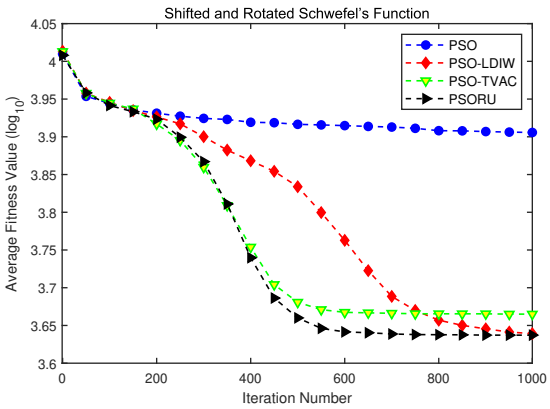

Fig. 3. Convergence plot of the Shifted and Rotated Schwefel's Function

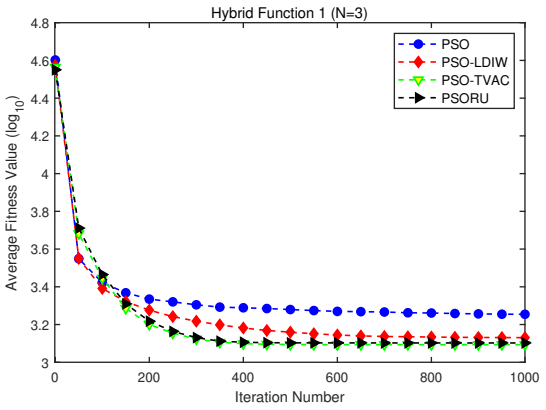

Fig. 4. Convergence plot of the Hybrid Function 1

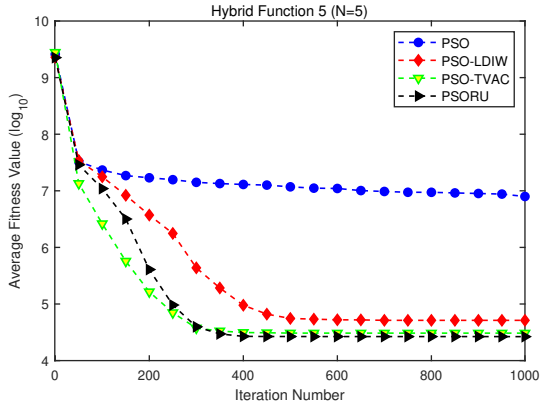

Fig. 5. Convergence plot of the Hybrid Function 5

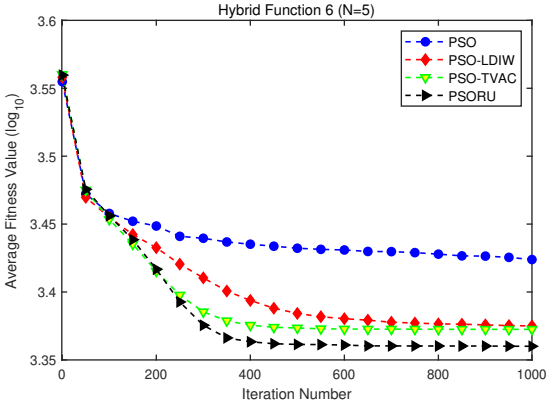

Fig. 6. Convergence plot of the Hybrid Function 6

This article has been accepted for publication in a future proceedings of this conference, but has not been fully edited. Content may change prior to final publication.
Citation information: DOI: 10.1109/ICAC65379.2025.11196536, 2025 30th International Conference on Automation and Computing (ICAC)

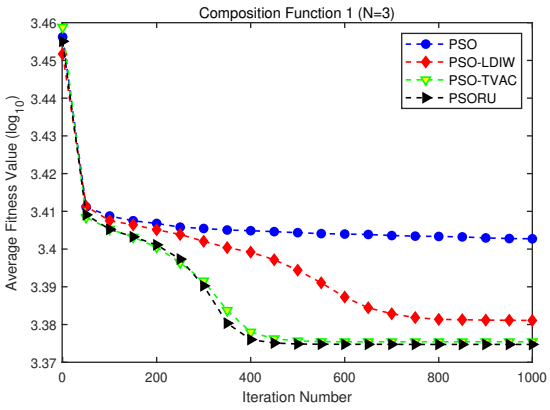

Fig. 7. Convergence plot of the Composition Function 1

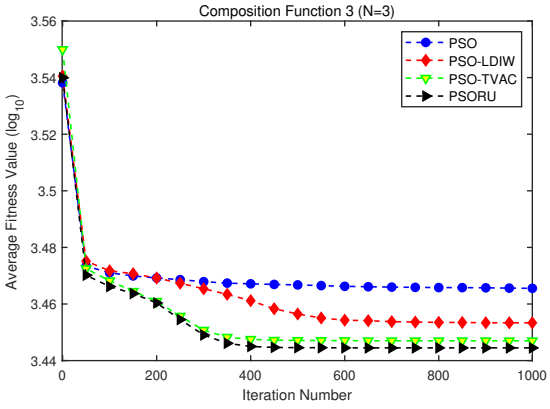

Fig. 8. Convergence plot of the Composition Function 3

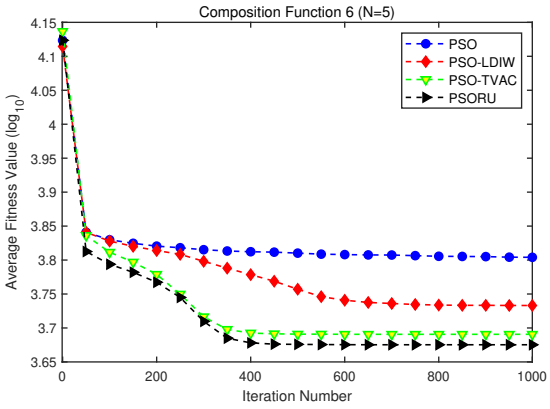

Fig. 9. Convergence plot of the Composition Function 1 ($N = 6$)

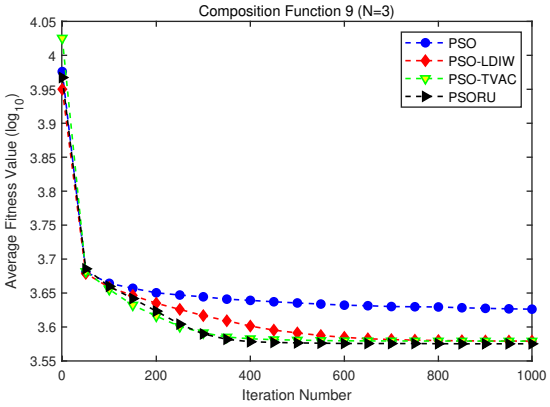

Fig. 10. Convergence plot of the Composition Function 9

The convergence plots of the chosen PSO algorithms are displayed in Figs. 1-10. It can be seen that the PSORU algorithm demonstrates significantly better convergence speed and lowest final fitness value compared to the PSO-LDIW and the PSO-TVAC algorithms. Specifically, the PSORU algorithm performs better on multimodal and composition functions, achieving best or comparable results on 8 out of 10 test functions as shown in Figs. 1-10. The PSORU algorithm does not achieve the best performance in Fig. 1, which corresponds to the Shifted and Rotated Bent Cigar Function. This result may be influenced by the relatively small population size and limited iteration count in the experiment. Under such settings, PSO-TVAC, with its time-varying acceleration coefficients, may better exploit the sharp ridge structure and converge more efficiently on this unimodal function. Nevertheless, PSORU still shows competitive performance in most of the selected functions, thereby showing strong global search capability and robustness. Besides, the PSORU algorithm converges fast on many benchmark functions, showing low sensitivity to different problem types. In summary, experimental results indicate that the PSORU algorithm is an effective and robust algorithm that is capable of handling various complex optimization problems.

IV. Conclusion and Future Work

This paper has presented a novel PSORU algorithm that incorporates randomly occurring uncertainty into the velocity update mechanism. The uncertainty has been modeled by the Cosine function. The effectiveness of the PSORU algorithm has been validated on a series of benchmark functions selected from the CEC2017 benchmark suite. Experimental results have shown that the PSORU algorithm demonstrates superior performance than the basic PSO, the PSO-TVAC, and the PSO-LDIW algorithms in terms of the convergence speed and the robustness for handling complex optimization problems. Some possible future directions include population sensitivity analysis, as the current setting employs a fixed swarm size without considering its influence on performance. Addition-

ally, the modeling of uncertainty plays a critical role in shaping the optimizer's dynamics. It is worth developing an adaptive mechanism to model uncertainty or dynamically adjust activation probabilities during the search process. Furthermore, future work could explore the scalability and robustness of the PSORU algorithm across problems of varying dimensionality and complexity.

REFERENCES

[1] J. Kennedy and R. Eberhart, Particle swarm optimization, In: *Proceedings of the International Conference on Neural Networks*, Perth, Australia, Nov. 1995, pp. 1942-1948.

[2] H. C. Huang, FPGA-based parallel metaheuristic PSO algorithm and its application to global path planning for autonomous robot navigation, *Journal of Intelligent & Robotic Systems*, vol. 76, pp. 475-488, 2014.

[3] J. Pugh, A. Martinoli and Y. Zhang, Particle swarm optimization for unsupervised robotic learning, In: *Proceedings of the 2005 IEEE Swarm Intelligence Symposium*, Pasadena, CA, USA, Jun. 2005, pp. 92-99.

[4] Y. Xue, B. Xue and M. Zhang, Self-adaptive particle swarm optimization for large-scale feature selection in classification, *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 5, pp. 1-27, 2019.

[5] S. Ait-Aoudia, E. Guerrout and R. Mahiou, Medical image segmentation using particle swarm optimization, In: *Proceedings of the 2014 International Conference on Information Visualisation*, Paris, France, July. 2014, pp. 287-291.

[6] J. Fang, W. Liu, L. Chen, S. Lauria, A. Miron, and X. Liu, A survey of algorithms, applications and trends for particle swarm optimization, *International Journal of Network Dynamics and Intelligence*, vol. 2, no. 1, pp. 24–50, Mar. 2023.

[7] Y. Shi and R. C. Eberhart, Parameter selection in particle swarm optimization, In: *Proceedings of the International Conference on Evolutionary Programming*, San Diego, USA, Mar. 1998, pp. 591-600.

[8] M. Clerc and J. Kennedy, The particle swarm - explosion, stability, and convergence in a multidimensional complex space, *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58-73, Feb. 2002.

[9] A. Ratnaweera, S. K. Halgamuge and H. C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240-255, 2004

[10] W. Liu, Z. Wang, N. Zeng, Y. Yuan, F. E. Alsaadi, and X. Liu, A novel randomised particle swarm optimizer, *International Journal of Machine Learning and Cybernetics*, vol. 12, pp. 529-540, Feb. 2021.

[11] W. Liu, Z. Wang, Y. Yuan, N. Zeng, K. Hone and X. Liu, A novel sigmoid-function-based adaptive weighted particle swarm optimizer, *IEEE Transactions on Cybernetics*, vol. 51, no. 2, pp. 1085-1093, Feb. 2021.

[12] N. H. Awad, M. Z. Ali, P. N. Suganthan, J. J. Liang, and B. Y. Qu, Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective real-parameter numerical optimization, in Technical Report, Singapore: Nanyang Technological University, 2016.