Hindawi
Shock and Vibration
Volume 2017, Article ID 8204867, 16 pages
https://doi.org/10.1155/2017/8204867

Research Article

An Improved Multiobjective Particle Swarm Optimization Algorithm Using Minimum Distance of Point to Line

Zhengwu Fan,[1] Tie Wang,[1] Zhi Cheng,[2] Guoxing Li,[1] and Fengshou Gu[1,3]

[1] *Department of Vehicle Engineering, Taiyuan University of Technology, Shanxi, China*
[2] *Department of Mechanical Engineering, Taiyuan University of Science Technology, Shanxi, China*
[3] *Centre for Efficiency and Performance Engineering, University of Huddersfield, Queensgate, Huddersfield HD1 3DH, UK*

Correspondence should be addressed to Tie Wang; wangtie57@163.com

Received 12 February 2017; Revised 9 May 2017; Accepted 15 May 2017; Published 25 September 2017

Academic Editor: Toshiaki Natsuki

In a multiobjective particle swarm optimization algorithm, selection of the global best particle for each particle of the population from a set of Pareto optimal solutions has a significant impact on the convergence and diversity of solutions, especially when optimizing problems with a large number of objectives. In this paper, a new method is introduced for selecting the global best particle, which is minimum distance of point to line multiobjective particle swarm optimization (MDPL-MOPSO). Using the basic concept of minimum distance of point to line and objective, the global best particle among archive members can be selected. Different test functions were used to test and compare MDPL-MOPSO with CD-MOPSO. The result shows that the convergence and diversity of MDPL-MOPSO are relatively better than CD-MOPSO. Finally, the proposed multiobjective particle swarm optimization algorithm is used for the Pareto optimal design of a five-degree-of-freedom vehicle vibration model, which resulted in numerous effective trade-offs among conflicting objectives, including seat acceleration, front tire velocity, rear tire velocity, relative displacement between sprung mass and front tire, and relative displacement between sprung mass and rear tire. The superiority of this work is demonstrated by comparing the obtained results with the literature.

1. Introduction

The dynamic behavior of a vehicle is critical for determining driving performance and ride comfort. Simultaneously, it also influences the dynamic loads applied to both the road and main axles, which can cause damage to road surfaces and early failure of chassis components. However, determining the system parameters usually conflicts with the main objectives of ride comfort, driving performance, and low dynamic loads. Therefore, optimization of these parameters has been actively studied. With the development of computational capacity and computing technologies, Nariman-Zadeh et al. [1] and Mahmoodabadi et al. [2] performed two-objective and five-objective optimization using advanced evolutionary algorithms and particle swarm optimization, which resulted in numerous effective trade-offs between conflicting objectives, including seat acceleration, front tire velocity, rear tire velocity, relative displacement between sprung mass and front tire, and relative displacement between sprung mass and rear tire, for a five-degree-of-freedom vehicle vibration model. Weiping et al. [3] presented the optimization of an eight-degree-of-freedom model using fuzzy theory and a multipopulation genetic algorithm to determine a set of parameters that achieves the best driver's seat performance. To reduce the dynamic load between the tires and road and improve ride comfort, Pengmin et al. proposed a modification of the suspension system parameters via a four-degree-of-freedom model [4]. Zhonglang and Pengmin [5] proposed modifying ride comfort and dynamic load through a seven-degree-of-freedom model and suspension system adjustment based on a specific 4 × 2 tractor case. Drehmer et al. [6] applied particle swarm optimization and sequential quadratic programming algorithms to obtain the optimal suspension parameters for different road profiles and vehicle velocity conditions.

Optimization is becoming an important research tool in scientific research and engineering practice. The multiobjective optimization problem (MOP) is normally considered the minimizing of an objective and can be in the form of the following equation:

$$\min \quad F(x) = \left[f_1(x), f_2(x) \cdots f_m(x)\right]^T \quad x \in \Omega. \quad (1)$$

Here, Ω is variable space and x is a variable vector that needs to be determined. $F(x)$ consists of a set of objective functions with objective space R^m. Each objective is generally contradicted during multiobjective optimization, that is, one of them being the best objective and the other objective being worst accordingly. In this case, it is less probable to find a solution that optimizes all the objectives. On this basis, the necessity of finding a solution trade-off is defined as making each objective into a balanced level, or so-called Pareto optimal solutions. Multiobjective optimizations require filtering of a set of optimal solutions instead of the sole best solution. One of the Pareto optimal solutions is not obviously better based on a lack of sufficiently detailed information, which is necessary for determining as many Pareto solutions as possible.

Multiobjective evolutionary algorithms (MOEAS) have been proposed to resolve multiobjective problems, for instance, the nondominated sorting genetic algorithm II (NSGA-II) [7] by Deb et al. and the Pareto archived evolution strategy (PAES) by Knowles and Corne [8], which approaches a nondominated front of solution set, as well as the Improved Strength Pareto Evolutionary Algorithm (SPEA2) [9] by Zitzler et al. These researchers were inspired by natural evolution and population evolutionary algorithms.

To improve previous solutions, a new solution called particle swarm optimization (PSO) [10] was proposed by Kennedy and Eberhart and based on population evolutionary algorithms and motivated by social behavior, such as bird flock food searching. The approach can be considered a distributed behavioral algorithm that performs (in a more general form) a multidimensional search. In the simulation, the behavior of each individual is affected by either the best local (i.e., within a certain neighborhood) or the best global individual. The approach uses the concept of population and a measure of performance similar to the fitness value used with evolutionary algorithms. Additionally, individual adjustments are analogous to the use of a crossover operator. However, this approach introduces the use of flying potential solutions through hyperspace (used to accelerate convergence), which does not seem to have an analogous mechanism in traditional evolutionary algorithms. Another important difference is that PSO allows individuals to benefit from their past experiences, whereas in an evolutionary algorithm, the current population is normally the only "memory" used by the individuals. PSO has been successfully used for both continuous nonlinear and discrete binary single-objective optimization. Particle swarm optimization seems particularly suitable for multiobjective optimization primarily due to the high convergence rate for single-objective optimization.

In recent years, PSO has been investigated to solve MOP problems.

Ray and Liew [11] introduced a swarm metaphor for multiobjective design optimization. The algorithm employs a multilevel sieve to generate a set of leaders, a probabilistic crowding radius-based strategy (based on a roulette-wheel scheme that ensures a larger crowding radius has a higher probability of being selected as a leader) for leader selection and a simple generational operator for information transfer.

Parsopoulos and Vrahatis [12] presented a study of the particle swarm optimization method in multiobjective optimization problems. This algorithm adopts an aggregating function where three approaches were implemented: a conventional linear aggregating function, a dynamic aggregating function, and the bang-bang weighted aggregation approach. According to this approach, all the objectives are summed to a weighted combination $F = \sum \omega_i f_i(x)$, where ω_i, $i = 1, \ldots, k$, are nonnegative weights. It is usually assumed that $\sum \omega_i = 1$. A multiobjective problem is scaled to a single-objective problem before optimization.

Hu and Eberhart [13] proposed a dynamic neighborhood PSO for multiobjective design optimization. In this algorithm, only one objective is optimized at a time. The relatively simple objective function is fixed, and the difficult function is optimized. This algorithm adopts one-dimensional optimization to handle multiple objectives.

Currently, the Pareto Dominance method has become increasingly popular, but none of the proposals to extend PSO to solve multiobjective optimization problems used a secondary population. This may limit the algorithm performance. However, in more recent papers, these ideas have already been incorporated by other authors. The most representative proposals are the following.

Coello and Lechuga [14] first extended PSO to handle multiobjective optimization problems by using a Pareto ranking scheme, and the use of global attraction mechanisms combined with a historical archive of previously found nondominated vectors would motivate convergence toward globally nondominated solutions. Additionally, repository updates and global best guide selection are performed considering a geographically based system (hypercube) defined in terms of the objective function values of each individual, which produces well-distributed Pareto fronts. The literature [15] includes an improved version of the algorithm [2], which has an added constraint-handling mechanism and a mutation operator that significantly improves the exploratory capabilities of the original algorithm to maintain diversity.

Hu et al. [16] proposed an improved version of the literature [13]. An *extended memory* is introduced to store global Pareto optimal solutions and reduce computation time.

Fieldsend and Sing [17] present a MOPSO, which uses an unconstraint archive. In this algorithm, a different data structure (called dominated tree) for storing the elite particles facilitates the choice of a best local guide for each particle of the population.

Mostaghim and Teich [18] proposed a MOPSO, which updated the external archive by *Pareto Dominance*. An archive is domination-free if no two points in the archive dominate each other. Obviously, while executing the function update, dominated points must be deleted to maintain a domination-free archive. Additionally, this method proposed a sigma method in which the best local guides for each particle are adopted to improve the convergence and diversity of a PSO approach used for multiobjective optimization.

Li [19] proposed an approach in which the main mechanisms of the NSGA-II [a] are adopted in a PSO algorithm. One mechanism is a Nondominated Sorting Particle Swarm Optimizer to push the population toward a Pareto front. To maintain population diversity, a crowding distance assignment mechanism is proposed that resorts the nondominant solutions in the nonDomPSOList according to crowding distance values and then randomly selects a global best guide for the *i*th particle from a specific part (e.g., top 5%) of the sorted nonDomPSOList.

The Raquel and Naval Jr. [20] proposed algorithm extends PSO to solve multiobjective optimization problems by incorporating the mechanism of crowding distance computation in the global best selection and the deletion method of the external archive of nondominated solutions whenever the archive is full. The crowding distance mechanism combined with a mutation operator maintains the diversity of nondominated solutions in the external archive. In this algorithm, a bounded external archive stores nondominated solutions found in previous iterations. The global best guide of the particles is selected from among the nondominated solutions with the highest crowding distance values. Similar to [12], this method randomly selects a global best guide for the *i*th particle from a specific part (e.g., top 10%) of the sorted archive.

In this paper, a minimum distance of point to line multiobjective particle swamp optimization (MDPL-MOPSO) is developed. The main differences between our approach and other proposals that exist in the literature are as follows:

(1) This algorithm adopts a bounded archive mechanism to store the global Pareto optimal solutions. The global best guide of each particle in the population was selected from the archive.

(2) A minimum distance of point to line method was utilized to find the global best guide of each particle in the population. The selection of the global best guide of the particle swarm is crucial in a multiobjective-PSO algorithm. It affects the convergence capability of the algorithm and maintains an adequate spread of nondominated solutions. No other proposal uses the mechanism in the way it is adopted in this paper.

(3) The mutation operator of MOPSO was adapted because of the exploratory capability it could provide the algorithm by initially performing mutation on the entire population and then rapidly decreasing its coverage over time [14]. This is helpful in terms of preventing premature convergence due to existing local Pareto fronts in some optimization problems.

(4) This algorithm is evaluated based on representative functions along with a performance comparison with the MOPSO algorithm [14], CD-MOPSO algorithm [15], and NSGA-II algorithm [7].

2. MDPL-MOPSO Algorithm

2.1. MDPL-MOPSO Implementation Procedure

(1) Particle swarm initialization;

- (a) Set swarm set number M and particle dimension D;
- (b) Set the range for variable $X(i)$ values for $X(i)_{\min}$ and $X(i)_{\max}$;
- (c) Particle speed control, variable $v(i)_{\max} = X(i)_{\max} - X(i)_{\max}$;
- (d) Swarm position and speed initialization, particle creation in random, $X(i)$, $v(i)$, $i = 1, 2, \ldots, M$.

(2) Parameter setting evolution;

- (a) Max Iterations $t_{\max}$;
- (b) Maximum and Minimum for inertia weight, value as $\omega_{\min} = 0.4$, $\omega_{\max} = 0.9$;
- (c) Learning factors c_1 and c_2, $c_1 = c_2 = 2$.

(3) Evaluate objective function value

For $i = 1$ to M (M is the population size)

For $k = 1$ to L (L is objective function numbers)

- (a) Evaluate $f_k(x_i)$
- (b) Particle normalization

$$f'_k(x_i) = \frac{f_k(x_i) - \min f_k(x_i)}{\max f_k(x_i) - \min f_k(x_i)}. \tag{2}$$

(4) Particle position initialization

$p_{best}(i) = X(i)$

$g_{best}(i) =$ Best particle found in $X(i)$

(5) Archive initialization

Store the non-dominated solutions found in X into Archive

(6) If iterations cannot achieve $t_{\max}$

- (a) Find $g_{best}(i)$ from Archive

  Line L_j solver for non-dominated particle $X(j)$ from archive.

  Distance d_{ij} from $X(i)$ to line L_j.

  While $d_{ik} = \min\{d_{ij}\}$, particle k in archive will be selected as the global best guide, $g_{ibest} = X(k)$.
- (b) Particle speed and position update

$$\begin{aligned} v_{t+1}(i) &= \omega \times v_t(i) + c_1 R_1 \left(pbest_t(i) - X_t(i)\right) \\ &\quad + c_2 R_2 \left(gbest_t(i) - X_t(i)\right), \\ X_{t+1}(i) &= X_t(i) + v_t(i), \end{aligned} \tag{3}$$

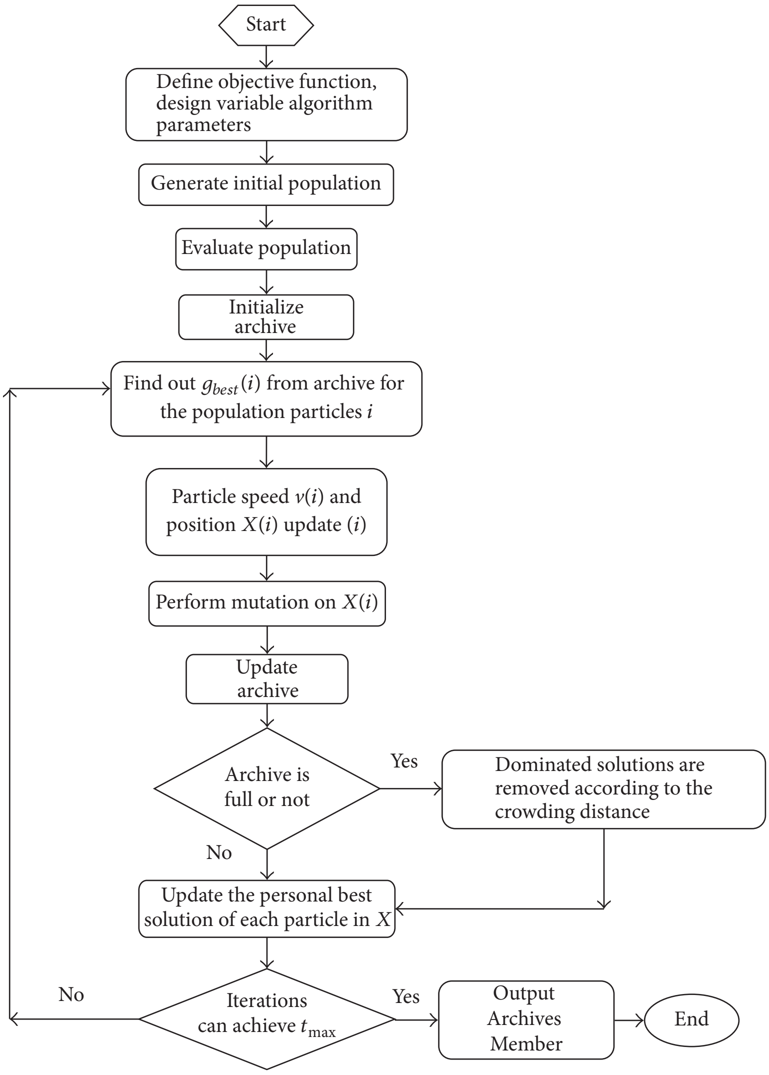

Start
Define objective function, design variable algorithm parameters
Generate initial population
Evaluate population
Initialize archive
Find out $g_{best}(i)$ from archive for the population particles i
Particle speed $v(i)$ and position $X(i)$ update (i)
Perform mutation on $X(i)$
Update archive
Archive is full or not
Yes
Dominated solutions are removed according to the crowding distance
No
Update the personal best solution of each particle in X
No
Iterations can achieve $t_{\max}$
Yes
Output Archives Member
End

Figure 1: Flowchart of the proposed algorithm for multiobjective optimization problems.

where t is the number of iterations, and R_1 and R_2 are random numbers in the range $[0\cdots 1]$.

If $X(i)$ extends beyond the boundaries, it is reintegrated by setting the decision variable equal to the value of its corresponding lower or upper boundary, and its velocity is multiplied by −1 so that it searches in the opposite direction.

(c) Perform mutation on $X(i)$.

(d) Evaluate $X(i)$

(e) Archive update

Insert the new non-dominated solution in X into Archive if they are not dominated by any of the stored solutions. All solutions in the archive dominated by the new solution are removed from the archive. If the archive is full, the solution to be replaced is determined according to the crowding distance values.

(f) Update the personal best solution of each particle in X. If the current $p_{best}(i)$ dominates the position in memory, the particles position is updated using $p_{best}(i) = X(i)$.

(g) Increment iteration counter t

(7) Cycle count increase until iteration requirements are achieved.

The flowchart of the proposed algorithm for MDPL-MOPSO algorithm is shown in Figure 1.

2.2. Finding the Global Best Guide. As mentioned previously, several important MOPSO methods exist [15, 16, 20]. In each of these methods, there is a suggestion for finding the global best guide. In most of these methods, there is an inspiration of multiobjective evolutionary algorithms. In this section, we discuss some of these methods, their advantages, and disadvantages and introduce a new method for finding the global best guide.

2.2.1. Overview of the Method of Finding the Global Best Guide

(1) Handling Multiple Objectives with Particle Swarm Optimization [15]. In this method, the objective space is divided to hypercubes before selecting the global best guide for each particle. Next, a fitness value is assigned to each hypercube depending on the number of elite particles within it. The more elite the particles that are in a hypercube, the lower the fitness value. Then, roulette-wheel selection is applied to the hypercube, and one is selected. Finally, the global best guide is a random particle selected from the selected hypercube. Therefore, the global best guide is selected using the roulette-wheel selection method, which is a random selection. It is possible that a particle does not select a suitable guide as its global guide.

(2) Multiobjective Optimization Using Dynamic Neighborhood Particle Swarm Optimization [16]. In this method, a dynamic neighborhood strategy is used. This paper showed that, for two-objective optimization, the global best guide for the particle is found in the objective space. First, the distance between particle i and other particles is calculated in terms of the first objective value, which is called a fixed objective. Then, k local neighbors based on the calculated distances are found. The local optima among the neighbors in terms of the second objective value are the global best guide g_{ibest} for particle i. In this method, fixed objective selection must be performed with a priori knowledge about the objective functions; and one-dimensional optimization is used to handle multiple objectives. Therefore, selecting the global best guide depends on only one objective.

(3) An Effective Use of Crowding Distance in Multiobjective Particle Swarm Optimization (CD-MOPSO) [20]. In this method, a crowding distance strategy is used. The crowding distance value of a solution provides an estimate of the density of solutions surrounding that solution. First, a bounded external archive stores nondominated solutions found in previous iterations. The nondominated solutions in the archive can be used as the global best guide of the particles in the swarm. Next, the nondominated solutions in the archive are sorted by decreasing crowding distance. Finally, the global best guide of the particles is selected from among the nondominated solutions with the highest crowding distance values. Selecting different guides for each particle in a specified top part of the sorted repository is implemented. Therefore, the global best guide is selected using the crowding distance value method, which is a random selection.

2.2.2. Minimum Distance of Point to Line Method. To overcome the disadvantages of the aforementioned methods, a MDPL-MOPSO is proposed to find the global best guide for each particle. First, the basic concept of the minimum distance of point to line (MDPL) is introduced. Later, this paper explains how this method determines the global best guide for each particle of the population in the objective.

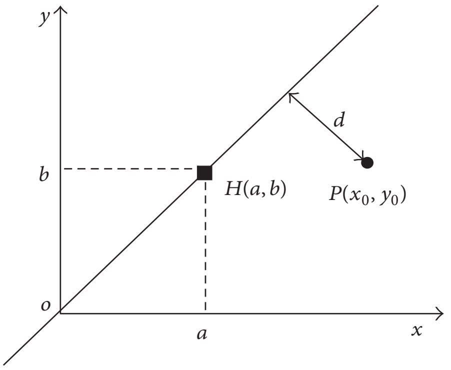


FIGURE 2: Distance of point to line 2D objective space.

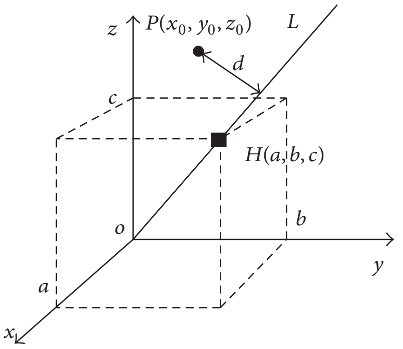


FIGURE 3: Distance of point to line in 3D objective space.

In the two-dimensional coordinate system, a straight line, line L, can be determined using the origin point $O(0,0)$ and any point $H(a,b)$ as shown in Figure 2. Line L is defined as follows:

$$\frac{x}{a} = \frac{y}{b}. \tag{4}$$

Point $P(x_0, y_0)$ is any point outside line L, and the distance d between point p and line L is defined as follows:

$$d = \frac{\left|bx_0 - ay_0\right|}{\sqrt{a^2 + b^2}}. \tag{5}$$

Similarly, in the three-dimensional coordinate system, a straight line can be produced through the origin point $O(0,0,0)$ and any point $H(a,b,c)$, as shown in Figure 3 as line L. Line L is defined as follows:

$$\frac{x}{a} = \frac{y}{b} = \frac{z}{c}. \tag{6}$$

Point $P(x_0, y_0, z_0)$ is any point outside line L, and the distance d between point p and line L is defined as follows:

$$\begin{aligned} d &= \frac{\left|\overrightarrow{OP} \times \overrightarrow{OH}\right|}{\left|\overrightarrow{OH}\right|} \\ &= \frac{\sqrt{\left(cy_0 - bz_0\right)^2 + \left(cx_0 - az_0\right)^2 + \left(bx_0 - ay_0\right)^2}}{\sqrt{a^2 + b^2 + c^2}}. \end{aligned} \tag{7}$$

Here, we use two-objective optimization as an example to illustrate how to find global optimal guide.

Using the basic concept of a distance of point to line and considering the objective space, finding the global best

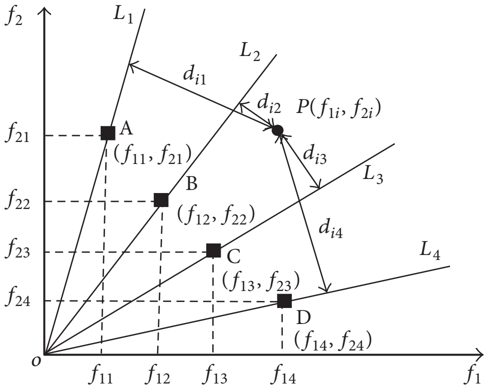


Figure 4: Finding the global best guide in two-objective space.

guide g_{best} among the archive members for particle i of two-objective optimization populations is as follows:

First, we draw line L_j of point $H(j)$ with coordinates (f_{1j}, f_{2j}) in the two-objective spaces. $X(j)$ is the related nondominated particle in the archive. L_j is defined as follows:

$$\frac{f_1}{f_{1j}} = \frac{f_2}{f_{2j}} \quad (j = 1, 2, \ldots, M). \tag{8}$$

Second, we calculate distance d_{ij} from point $P(i)$ with coordinates (f_{1i}, f_{2i}) in the objective space related to the population particle $X(i)$ to line L_j $(j = 1, 2, \ldots, M)$. d_{ij} is defined as follows:

$$d_{ij} = \frac{\left|f_{2j} f_{1i} - f_{1j} f_{2i}\right|}{\sqrt{f_{1j}^2 + f_{2j}^2}} \quad (j = 1, 2, \ldots, M,\ j \neq i). \tag{9}$$

Finally, archive particle $X(k)$ will be considered $g_{ibest} = X(k)$ in terms of optimization, while distance d_{ik} from $P(i)$ to archive line L_k is the minimum. d_{ik} is defined as follows:

$$d_{ik} = \min \quad \left\{d_{ij} \mid j = 1, 2, \ldots, m\right\}. \tag{10}$$

In other words, each particle with a minimum distance to the line of the archive member must select that archive member as the global best guide. Therefore, MDPL-MOPSO can determine the most appropriate guide as its global guide for each particle in the population.

As shown in Figure 4, point $H(j)$ $(j = 1, 2, 3, 4)$ with coordinates (f_{11}, f_{21}), (f_{12}, f_{22}), (f_{13}, f_{23}), and (f_{14}, f_{24}) is the Pareto optimal front four points of the two-objective space whose nondominated particles in the archive are $X(j)$ $(j = 1, 2, 3, 4)$. Line L_j $(j = 1, 2, 3, 4)$ is achieved according to (14). The corresponding point of particle $X(i)$ in the objective space is point $P(i)$ with coordinates (f_{1i}, f_{2i}). The distance d_{ij} $(j = 1, 2, 3, 4)$ is the distance between point $P(i)$ and line L_j. If it is assumed that the distance d_{12} is the minimum value, the nondominated particle $X(2)$ in the archive is selected. Particle $X(2)$ is the global best guide of population particle $X(i)$.

Figure 5 shows how the algorithm determines the global best guide among the archive members for each particle of the population for a two-dimensional objective space. The reason particle k is selected from the archive members as the global best guide is the minimum distance between the population particles i and line L of particle k. This method can guide the population particles to move directly toward the Pareto optimal front.

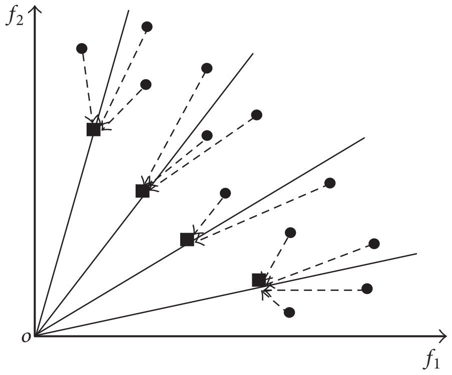


Figure 5: Finding the global best guide for each particle.

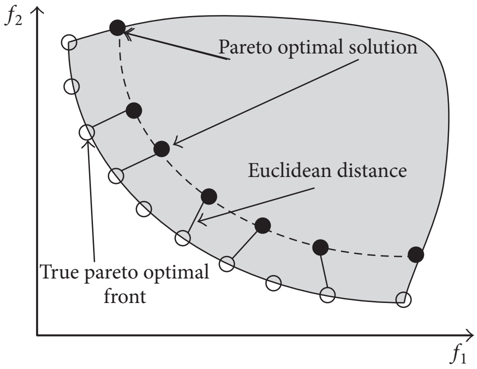


Figure 6: Convergence measure GD.

3. Performance Evaluations

3.1. Testing Functions and Performance Targets. To evaluate the performance of the proposed MDPL-MOPSO, six representative test functions denoted as DT1, ZDT2, ZDT3, ZDT4, and ZDT6 are employed, which are detailed in Table 1. These functions were defined and used by Zitzler et al. [21] to demonstrate the effectiveness of their evolutionary algorithms in handling the common issues of multiobjective converging to the Pareto optimal front and maintaining diversity within the population.

In addition, convergence and diversity are the significant criteria for algorithm performance. Generational distance (GD) is one measure to represent convergence metric, and Spacing (S) is a measure of equilibrium of distribution and diversity [7].

(1) Generational distance (GD): convergence measure (GD) is a measure of the convergence range of the multiobjective optimization algorithm. First, we find a set of solutions from the true Pareto optimal front in the objective space. Second, for each solution obtained with an algorithm, we compute the minimum Euclidean distance from chosen solutions on the Pareto optimal front. The root mean square is expressed as the convergence measure GD. As shown in Figure 6, the shaded area represents the possible solution

Table 1: Characteristics of test functions.

Problem	n	Bounds	Objective functions	Optimal solutions	Comments
ZDT1	30	[0, 1]	$f_1(x) = x_1$ $f_2(x) = 1 - \sqrt{\frac{x_1}{g(x)}}$ $g(x) = 1 + \frac{9\left(\sum_{i=2}^{n} x_i\right)}{(n-1)}$	$x_1 \in [0,1]$ $x_i = 0$ $i = 2, \ldots, n$	Convex
ZDT2	30	[0, 1]	$f_1(x) = x_1$ $f_2(x) = 1 - (\frac{x_1}{g(x)})^2$ $g(x) = 1 + \frac{9\left(\sum_{i=2}^{n} x_i\right)}{(n-1)}$	$x_1 \in [0,1]$ $x_i = 0$ $i = 2, \ldots, n$	Nonconvex
ZDT3	30	[0, 1]	$f_1(x) = x_1$ $f_2(x) = 1 - \sqrt{\frac{x_1}{g(x)}} - \frac{x_1 \sin(10\pi x_1)}{g(x)}$ $g(x) = 1 + \frac{9\left(\sum_{i=2}^{n} x_i\right)}{(n-1)}$	$x_1 \in [0,1]$ $x_i = 0$ $i = 2, \ldots, n$	Convex disconnected
ZDT4	10	$x_1 \in [0,1]$ $x_i \in [-5,5]$	$f_1(x) = x_1$ $f_2(x) = 1 - \sqrt{\frac{x_1}{g(x)}}$ $g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}\left[x_i^2 - 10\cos(4\pi x_i)\right]$	$x_1 \in [0,1]$ $x_i = 0$ $i = 2, \ldots, n$	Nonconvex
ZDT6	10	[0, 1]	$f_1(x) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$ $f_2(x) = 1 - \left(\frac{x_1}{g(x)}\right)^2$ $g(x) = 1 + 9\left[\frac{\left(\sum_{i=2}^{n} x_i\right)}{(n-1)}\right]^{0.25}$	$x_1 \in [0,1]$ $x_i = 0$ $i = 2, \ldots, n$	Nonconvex Not uniformly spaced

area. The solid line is the true Pareto optimal front, and the hollow point is the selected theoretical optimal solution. The dashed line is the Pareto optimal front obtained by the multiobjective optimization algorithm. The real point is the Pareto optimal solution of the archive. The smaller the value, the better the convergence of the algorithm

$$\mathrm{GD} = \left(\frac{1}{n}\sum_{i=1}^{n} d_i^2\right)^{1/2}, \tag{11}$$

where n is the quantity of Pareto front solutions using an optimized algorithm and d_i denotes the distance between Pareto front particle i and the closest Euclid of the optimum solution. GD theoretically represents the distance between the Pareto front based on algorithm optimization and the optimized solution for testing functions. A smaller GD value indicates better convergence of the algorithm.

(2) Diversity measure S is a measure of the distribution and diversity of the multiobjective optimization algorithm. It is more important to obtain a set of optimal solutions distributed throughout the entire Pareto front. As shown in Figure 7, the shaded areas represent the possible solution areas, the dashed lines are the Pareto optimal front obtained by the multiobjective optimization algorithm, and the solid points are the Pareto optimal solutions for the external archives. The smaller the value, the better the convergence of the algorithm

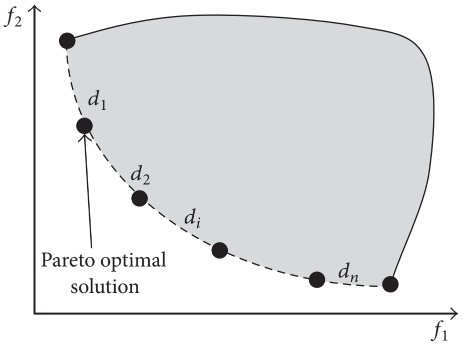


Figure 7: Diversity measure S.

$$s = \left[\frac{1}{n}\sum_{i=1}^{n}\left(d_i - \overline{d}\right)^2\right]^{1/2}, \tag{12}$$

$$\overline{d} = \frac{1}{n}\sum_{i=1}^{n} d_i, \tag{13}$$

where n is the quantity of Pareto front solutions using an optimized algorithm and d_i denotes the Euclidean distance (in the objective space) between the members i in the Pareto optimal front and its nearest member in the Pareto optimal front. A smaller value of S implies a more uniform distribution of solutions in the Pareto optimal front.

Table 2: GD results of optimal solutions for every problem.

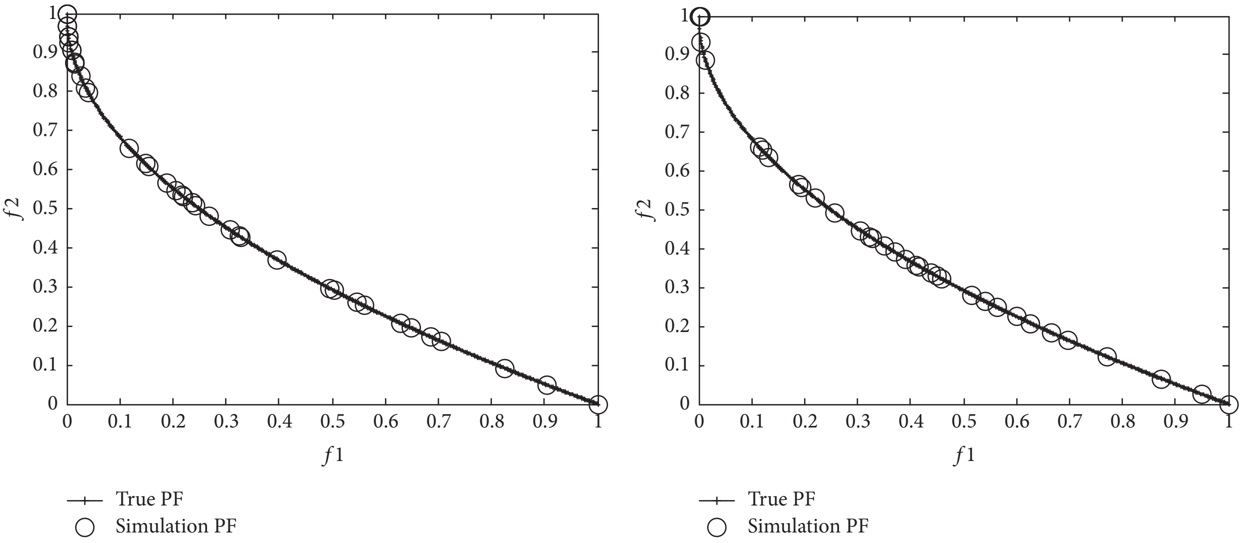
Optimization algorithm	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
MPDL-MOPSO	0.001165*	0.000183	0.017570	0.001670	0.115910
	0.000075**	0.000025	0.001605	0.000263	0.017119
CD-MOPSO	0.001565*	0.001017	0.017990	0.002667	0.096690
	0.000243**	0.000060	0.000983	0.000347	0.009128

Note. *Mean value; **variance value.

Table 3: S results of optimal solutions for every problem.

Optimization algorithm	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
MPDL-MOPSO	0.001094*	0.000155	0.016820	0.001634	0.093060
	0.000074**	0.000016	0.001564	0.000253	0.012801
CD-MOPSO	0.001539*	0.000994	0.017550	0.002613	0.084730
	0.000233**	0.000059	0.000959	0.000273	0.007736

Note. *Mean value; **variance value.



(a) MDPL-MOPSO

(b) CD-MOPSO

Figure 8: Nondominated solutions with MDPL-MOPSO and CD-MOPSO on ZDT1.

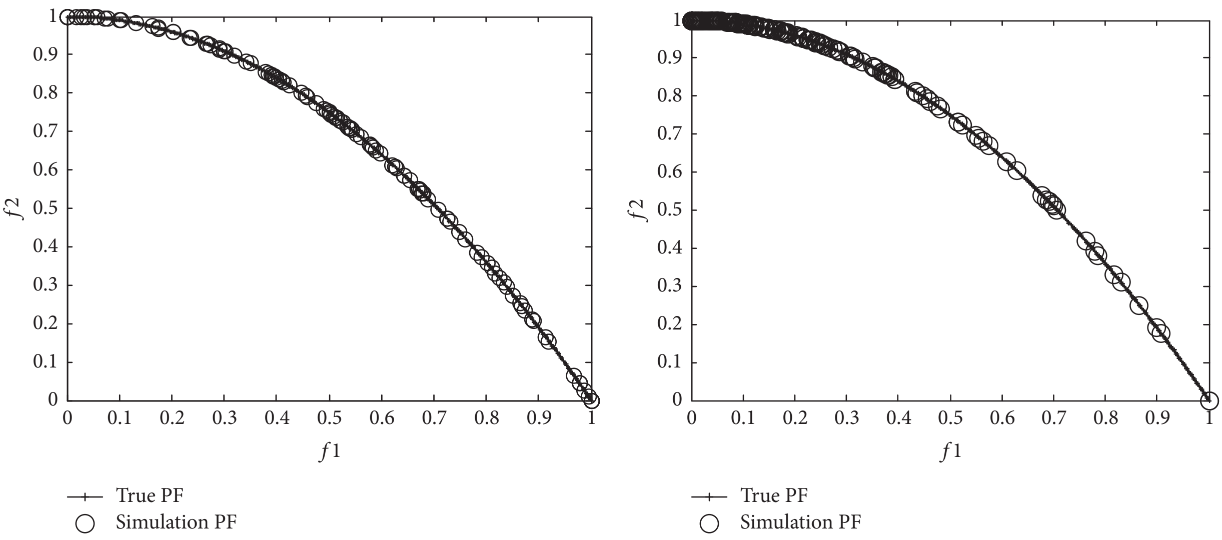
3.2. Results and Discussion. To verify the performance of the proposed algorithm, a comparison analysis is provided between the results and CD-MOPSO that are conducive to archive update and maintenance as well as g_{best} selection. Through a ten-cycle testing function simulation, GD for convergence and S of diversity could be calculated. This can be summarized as a mean value for algorithm performance and variance for stability.

Raquel and Naval Jr. [20] proposed a CD-MOPSO algorithm. There are 30 particles in the particle swarm, the maximum number of iterations is 400, and the learning factor is 2. The MDPL-MOPSO algorithm has the same parameters as the CD-MOPSO algorithm. In the following analysis, results obtained by performing 30 independent tests for each algorithm are compared.

Figures 8–12 indicate a comparison analysis between the Pareto front generated from five testing functions and a true Pareto front. Tables 2 and 3 refer to two different algorithms of convergence GD value and diversity S value for the five testing functions. Row 1 is the mean value and Row 2 is the variance value in the figures.

From Figures 8 to 12, both MDPL-MOPSO and CD-MOPSO can cover the Pareto front of ZDT1, ZDT2, and ZDT4, despite ZDT3 and ZDT6 being partially covered. Compared with CD-MOPSO, MDPL-MOPSO is closer to the Pareto front in terms of ZDT3, while CD-MOPSO is more suitable regarding ZDT6. This can also be observed from the information in Tables 2 and 3.

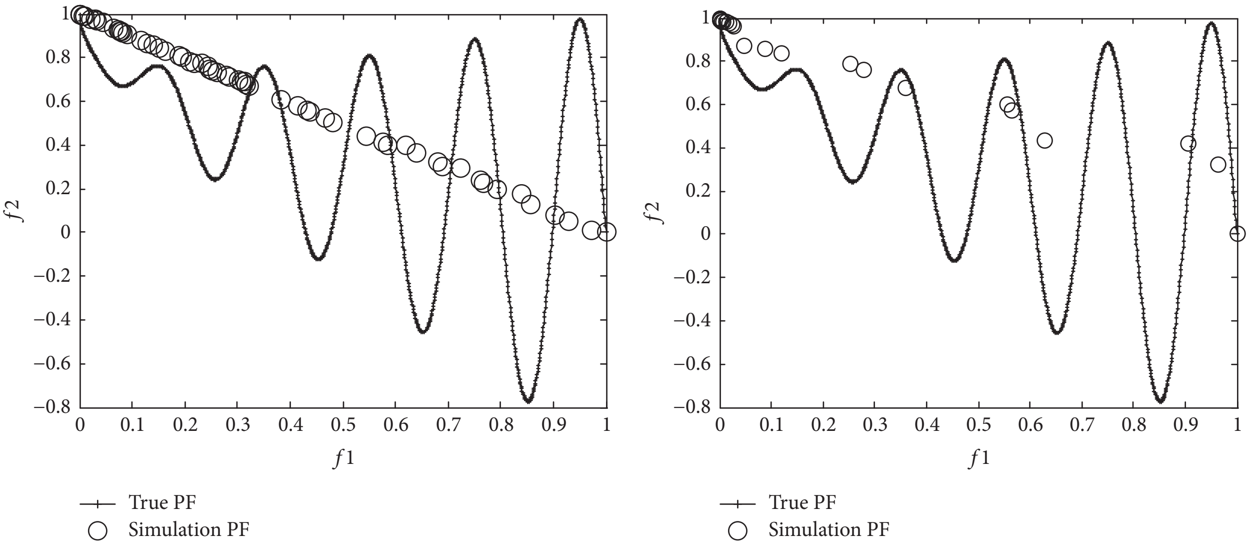
Table 2 refers to convergence GD of testing function optimum solutions, in which lower values indicate better convergence. Furthermore, the GD results from MDPL-MOPSO are less than the GD in CD-MOPSO for functions ZDT1, ZDT2, ZDT3, and ZDT4, which indicates that the convergence and stability of MDPL-MOPSO are better than



(a) MDPL-MOPSO

(b) CD-MOPSO

Figure 9: Nondominated solutions with MDPL-MOPSO and CD-MOPSO on ZDT2.



(a) MDPL-MOPSO

(b) CD-MOPSO

Figure 10: Nondominated solutions with MDPL-MOPSO and CD-MOPSO on ZDT3.

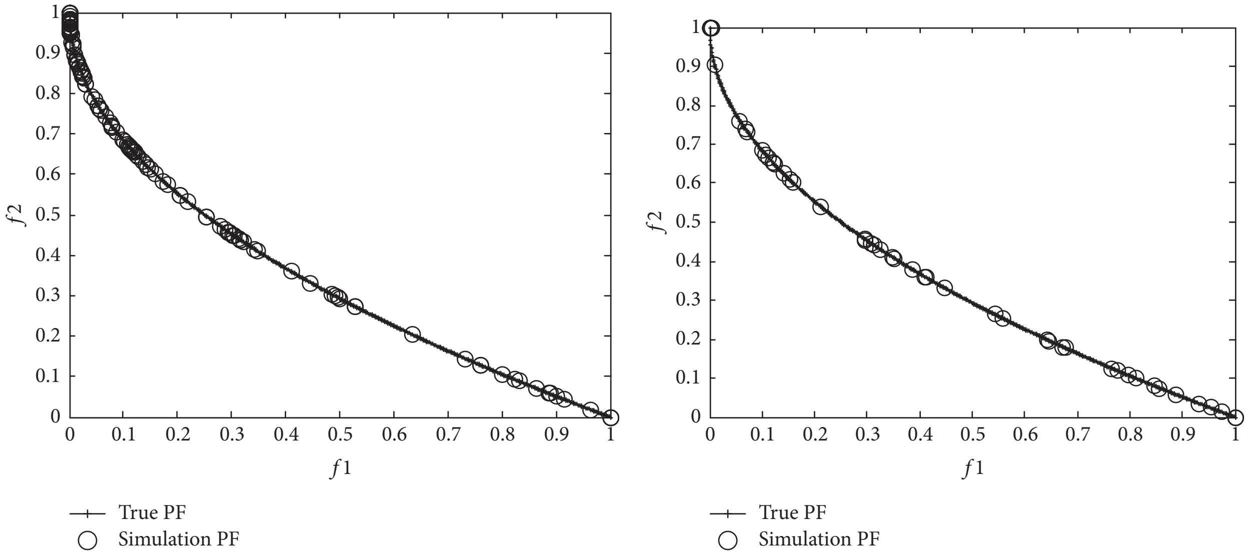
CD-MOPSO, and the mean value and variance value are less than in CD-MOPSO. Considering GD mean value increased by 16% as function ZDT6, the convergence of MDPL-MOPSO is slightly worse than CD-MOPSO.

Table 3 is the diversity *S* for testing function optimum solutions. For lower values of *S*, the diversity is improved. Of the Table 3 information, the *S* results and GD convergence follow the same trend. However, the diversity of the MDPL-MOPSO algorithm improves as ZDT1–ZDT4 is considered. As for ZDT6, the diversity of MDPL-MOPSO is slightly less conducive than CD-MOPSO because the *S* mean value increases by approximately 9%.

Therefore, it can be concluded that MDPL-MOPSO has improved convergence and diversity performance compared with that of the CD-MOPSO algorithm.

4. Multiobjective Optimization Design of Vehicle Driving Dynamics

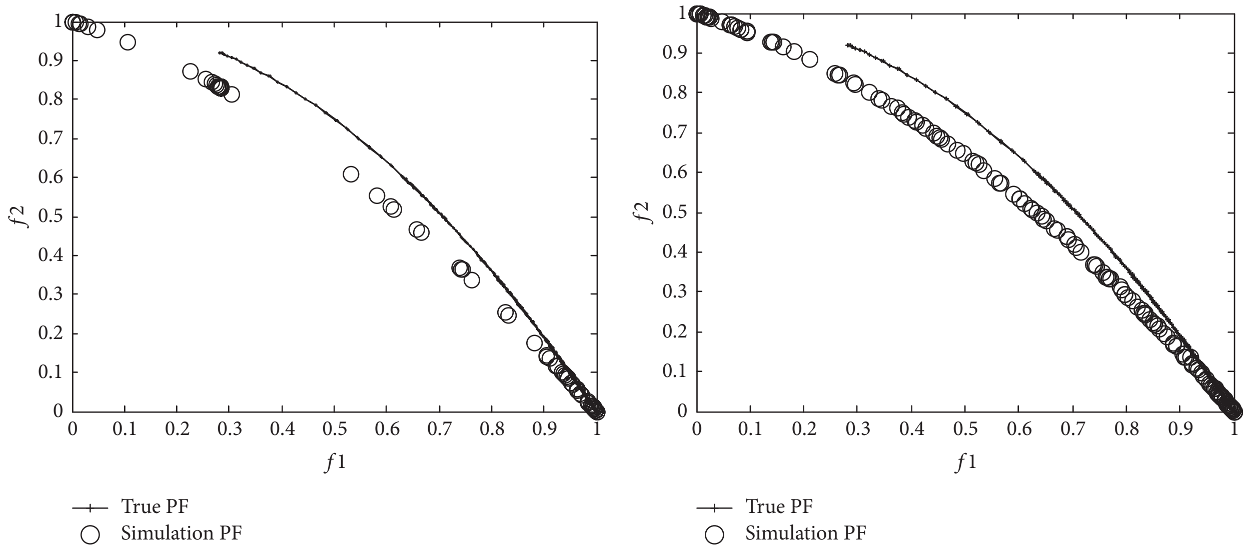
Nariman-Zadeh et al. [1] and Mahmoodabadi et al. [2] performed two-objective and five-objective optimization using advanced evolutionary algorithms and particle swarm optimization, which resulted in numerous effective trade-offs between conflicting objectives, including seat acceleration,



(a) MDPL-MOPSO

(b) CD-MOPSO

Figure 11: Nondominated solutions with MDPL-MOPSO and CD-MOPSO on ZDT4.



(a) MDPL-MOPSO

(b) CD-MOPSO

Figure 12: Nondominated solutions with MDPL-MOPSO and CD-MOPSO on ZDT6.

front tire velocity, rear tire velocity, relative displacement between sprung mass and front tire, and relative displacement between sprung mass and rear tire, for a five-degree-of-freedom (five-DOF) vehicle vibration model.

In this paper, an improved multiobjective particle swarm optimization (MDPL-MOPSO) algorithm is used to optimize the five-DOF vehicle driving dynamics in [1, 2] and is compared with the references.

4.1. Five-Degree-of-Freedom Vehicle Model. A five-degree-of-freedom vehicle with passive suspension, which is adopted from [2], is shown in Figure 13.

The fixed parameters and design variables of the vehicle are shown in Table 4.

Kinetic energy of vehicle vibration five-degree-of-freedom model:

$$T = \frac{1}{2} m_1 \dot{Z}_1^2 + \frac{1}{2} m_2 \dot{Z}_2^2 + \frac{1}{2} m_s \dot{Z}_s^2 + \frac{1}{2} m_c \dot{Z}_c^2 + \frac{1}{2} J_s \dot{\theta}^2. \quad (14)$$

Potential energy of vehicle vibration five-degree-of-freedom model:

$$\begin{aligned} U = {} & \frac{1}{2} K_{s1} \left(Z_s - L_1\theta - Z_1\right)^2 + \frac{1}{2} K_{s2} \left(Z_s + L_2\theta - Z_2\right)^2 \\ & + K_{ss} \left(Z_c - Z_s + r\theta\right)^2 + \frac{1}{2} K_{p1} \left(Z_1 - q_1\right)^2 \\ & + \frac{1}{2} K_{p2} \left(Z_2 - q_2\right)^2 . \end{aligned} \quad (15)$$

Table 4: Vehicle parameters.

Parameters	Symbol	Unit	Value
Forward tire mass	m_1	kg	40
Rear tire mass	m_2	kg	35.5
Seat mass	m_c	kg	75
Sprung mass	m_s	kg	730
Momentum inertia of sprung mass	J_s	kg·m^2	130
Forward tire relation to the center of mass	l_1	m	1.011
Rear tire relation to the center of mass	l_2	m	1.803
Forward tire stiffness coefficient	K_{p1}	N·m^{-1}	175500
Rear tire stiffness coefficient	K_{p2}	N·m^{-1}	175500
Seat stiffness coefficient	K_{ss}	N·m^{-1}	50000–150000
Forward suspension stiffness coefficients	K_{s1}	N·m^{-1}	10000–20000
Rear suspension stiffness coefficients	K_{s2}	N·m^{-1}	10000–20000
Seat damping coefficient	C_{ss}	(N·s) m^{-1}	1000–4000
Forward suspension damping coefficient	C_{s1}	(N·s)·m^{-1}	500–2000
Rear suspension damping coefficient	C_{s2}	(N·s)·m^{-1}	500–2000
Seat position in relation to the center of mass	r	m	0–0.5

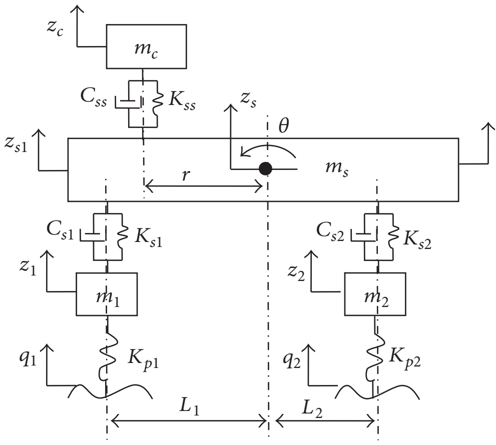


Figure 13: A five-degree-of-freedom vibration model for a vehicle with passive suspension adopted from [2].

Dissipation energy of vehicle vibration five-degree-of-freedom model:

$$D = \frac{1}{2}C_{s1}\left(\dot{Z}_s - L_1\dot{\theta} - \dot{Z}_1\right)^2 + \frac{1}{2}C_{s2}\left(\dot{Z}_s + L_2\dot{\theta} - \dot{Z}_2\right)^2 + C_{ss}\left(\dot{Z}_c - \dot{Z}_s + r\dot{\theta}\right)^2. \quad (16)$$

Substituting (13)–(15) into the Lagrange motion equation yields the vibration differential equations:

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_i}\right) - \frac{\partial T}{\partial q_i} + \frac{\partial U}{\partial q_i} + \frac{\partial D}{\partial \dot{q}_i} = 0. \quad (17)$$

Then, a matrix form of the vibration model of (17) can be written for efficiently obtaining solutions efficiently:

$$M\ddot{Z} + C\dot{Z} + KZ = K_F Q, \quad (18)$$

where matrices M, C, and K represent the mass matrix, damping matrix, and stiffness, K_F is the tire stiffness matrix, and Z, $\dot{Z}$, and $\ddot{Z}$ are displacement, speed, and acceleration column vectors. Q represents the road activation vector. Displacement vector $Z = [Z_1, Z_2, Z_s, Z_c, \theta]^{\mathrm{T}}$ and the road activation vector is $Q = [q_1, q_2, 0, 0]^{\mathrm{T}}$. The road activation vector is shown in Figure 14.

It is assumed that the vehicle moves at a constant velocity $v = 20$ m/s. It is further assumed that the rear tire follows the same trajectory as the front tire with a delay of $\Delta t = (L_1 + L_2)/v$.

4.2. Design Variables and Objective Functions

4.2.1. Design Variables. To optimize the suspension system, corresponding stiffness and damping parameters are used as the design variables, which are defined as $X = [x_1, x_2, x_3, x_4, x_5, x_6, x_7] = [K_{ss}, K_{s1}, K_{s2}, C_{ss}, C_{s1}, C_{s2}, r]$. Their bounds are defined in Table 4.

4.2.2. Objective Functions. The five conflicting objectives are seat acceleration $\ddot{Z}_c$, front tire velocity $\dot{Z}_1$, rear tire velocity $\dot{Z}_2$, relative displacement between sprung mass and front tire d_1, and relative displacement between sprung mass and rear tire d_2 for the five-DOF vehicle vibration model. For this purpose, four of the ten possible pairs of five objectives are considered in various two-objective optimization processes. The pairs of objectives to be optimized separately chosen were $(\ddot{Z}_c, \dot{Z}_1)$, $(\ddot{Z}_c, \dot{Z}_2)$, $(\ddot{Z}_c, d_1)$, and $(\ddot{Z}_c, d_2)$. All the objective functions must be minimized simultaneously.

Table 5: Objective function values and trade-off design variables of [1, 2] and this work shown in Figure 15.

Point	K_{ss}	K_{s1}	K_{s2}	C_{ss}	C_{s1}	C_{s2}	r	$\ddot{Z}_c$	$\dot{Z}_1$
A1 (by this work)	50000	10000	10000	4000	1318.9748	500	0.5	1.4796	0.1398
B1 (by this work)	150000	20000	19288.60	1235.31	2000.000	2000	0.5	2.94514	0.13248
C1 (by this work)	150000	15881.94	10000	4000	2000	1382.29	0.500	2.22834	0.13386
C1_1 (by [1])	111177	10000	10117.700	3858.82	1264.710	1852.940	0.49804	2.92268	0.40883
C1_1′ (by [2])	144263.6	10003.33	10029.590	3119.21	1232.827	1989.187	0.49635	2.91323	0.40863

Table 6: Objective function values and trade-off design variables of [1, 2] and this work shown in Figure 15.

Point	K_{ss}	K_{s1}	K_{s2}	C_{ss}	C_{s1}	C_{s2}	r	$\ddot{Z}_c$	$\dot{Z}_2$
A2 (by this work)	50000	10000	10000	1000	2000	500	0	1.81877	0.13561
B2 (by this work)	60399.789	10000	10000	1000	500	500	0	2.50843	0.12994
C2 (by this work)	88190.529	10013.127	10013.979	1003.96	1247.35	502.018	0.0008	2.02698	0.13234
C2_2 (by [1])	131961	10000	10000	3400	1282.35	1613.628	0.47112	2.94568	0.43178
C2_2′ (by [2])	139269.5	10000.420	10036.32	3485	1259.88	1511.770	0.46863	2.95293	0.43162

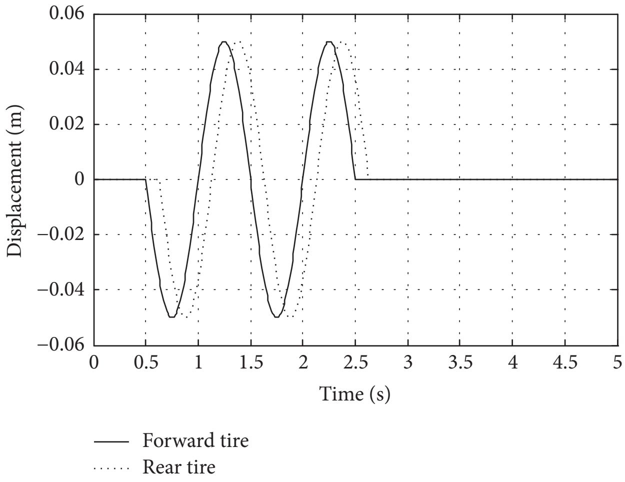


Figure 14: Road activation vector.

4.3. Multiobjective Optimization Result Analysis. The Pareto front for the vibration system is obtained using MDPL-MOPSO and is shown in Figures 15 and 16. Obviously, these two objectives are interinfluenced, as one objective is better optimized than another objective. The two end points demonstrate the two extreme cases of these conflicting objectives. However, the points between these extremes are the nondominated optimum points, any of which can be selected as design points. In other words, any other set of objectives will locate a point inferior to the corresponding Pareto front.

Figure 15 shows the Pareto front of seat acceleration and forward tire velocity representing different nondominated optimum points with respect to these conflicting objectives. In this figure, points A1, A2 and B1, B2 represent the best seat acceleration and best forward tire velocity, respectively. Note that all the optimum design points in this Pareto front are nondominated and could be chosen by a designer. Considering the actual engineering optimization problem, the engineer chooses a trade-off solution according to the actual request. In Figure 15, points C1 and C2 are trade-off design points.

Nondominated Pareto fronts for another chosen set of objective functions are shown in Figure 16. Design points A3 and A4 represent the best seat acceleration ($\ddot{Z}_c$), while points B2, B3, and B4 represent the best $\dot{Z}_2$, d_1, and d_2, respectively. Similarly, C3 and C4 are selected as the trade-off design points.

The corresponding values of objective functions and design variables of these optimum design points are provided in Tables 5–8. In these tables, points C_1, C_2, C_3, C_4, and C_1′, C_2′, C_3′, C_4′ represent the optimum design points reported in [1, 2]. It is evident that those design points are significantly dominated by proposed trade-off designing points.

The time behavior of the seat acceleration of the trade-off design points of these figures and the optimum point proposed in [1, 2] is shown for comparison in Figures 17–20. It is obvious from these figures that the seat acceleration for the design points obtained in this paper is better than the acceleration using the design points given in [1, 2].

5. Conclusions

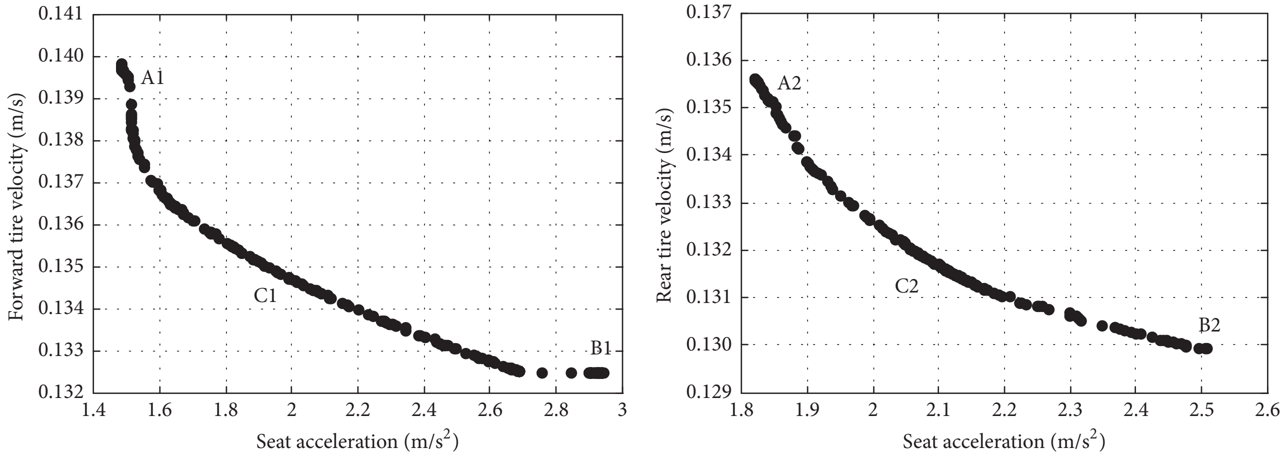
In this work, an improved multiobjective particle swarm optimization algorithm was introduced. A new method was proposed to determine the global best guide particle for the population particle based on the basic concept of the minimum distance of point to line (MDPL). To consider the optimization performance of the proposal algorithm, five two-objective test functions were used. The results were compared with the results of the CD-MOPSO algorithm.

Table 7: Objective function values and trade-off design variables of [1, 2] and this work shown in Figure 16.

Point	K_{ss}	K_{s1}	K_{s2}	C_{ss}	C_{s1}	C_{s2}	r	$\ddot{Z}_c$	d_1
A3 (by this work)	50000	10000	10000	4000	1483.59	500	0.5	1.0652	0.01566
B3 (by this work)	50000	20000	20000	1000	2000	500	0.0813	2.0172	0.007398
C3 (by this work)	50000	10865.21	10865.21	3740.44	1963.46	500	0.0745	1.3981	0.008484
C3_3 (by [1])	134706	10000.000	10078.400	3768.22	1999.95	1432.873	0.49299	2.80956	0.089026
C3_3′ (by [2])	147210.7	10001.550	10003.770	2505.88	2000	1511.770	0.50000	2.82	0.09

Table 8: Objective function values and trade-off design variables of [1, 2] and this work shown in Figure 16.

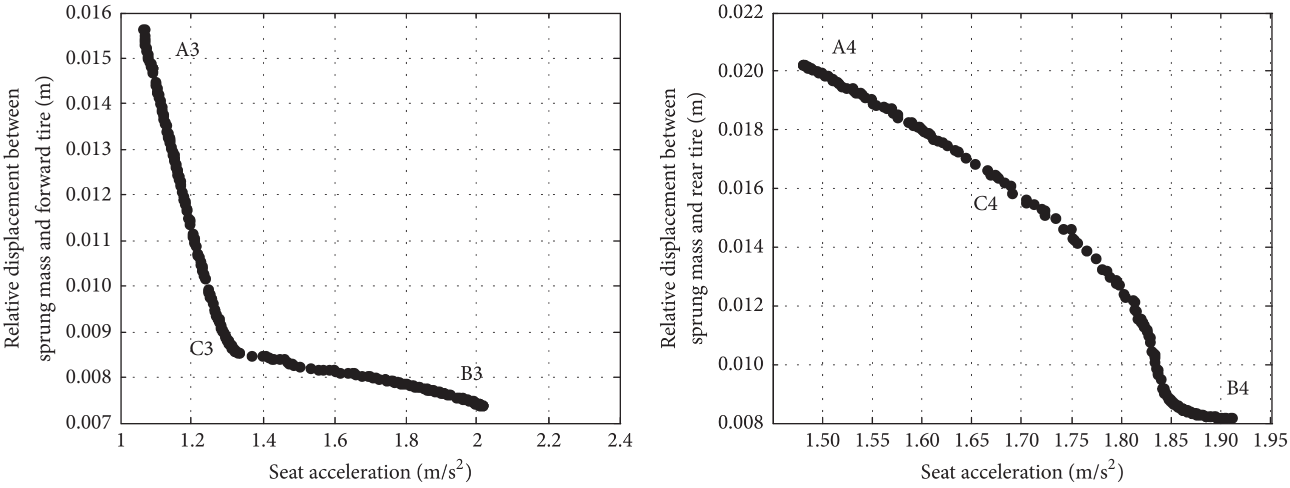
Point	K_{ss}	K_{s1}	K_{s2}	C_{ss}	C_{s1}	C_{s2}	r	$\ddot{Z}_c$	d_2
A4 (by this work)	50000	10000	10000	4000	1308.95	500	0.5	1.47959	0.02025
B4 (by this work)	50000	10000	20000	1967.74	1550.77	2000	0.5	1.91086	0.00819
C4 (by this work)	50000	10000	17870.18	4000	1219.67	1118.15	0.5	1.80211	0.01241
C4_4 (by [1])	98235.30	10000.25	19999.92	3482.94	1336.78	1999.61	0.49911	2.95040	0.05156
C4_4′ (by [2])	146265.70	10000	18313.70	2917.65	1311.77	1994.12	0.5	2.97674	0.05345



(a) Forward tire

(b) Rear tire

Figure 15: Obtained Pareto front for seat acceleration and forward/rear tire velocity.



(a) Forward tire

(b) Rear tire

Figure 16: Pareto front for seat acceleration and relative displacement between sprung mass and forward/rear tire.

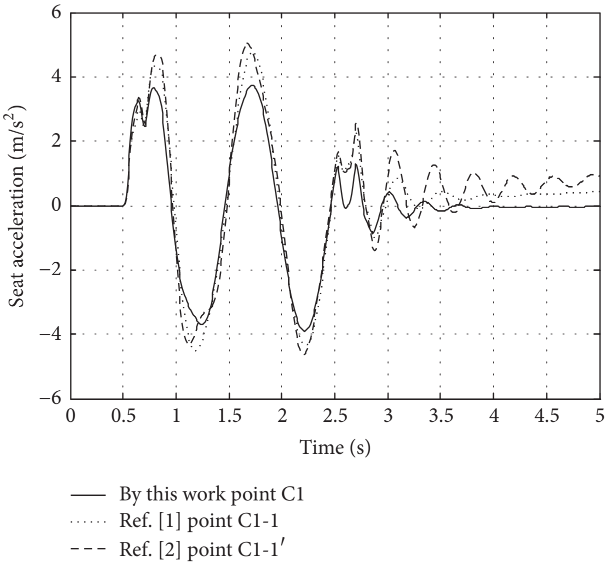


Figure 17: Time response of seat acceleration of design points C1, C1_1, and C1_1′.

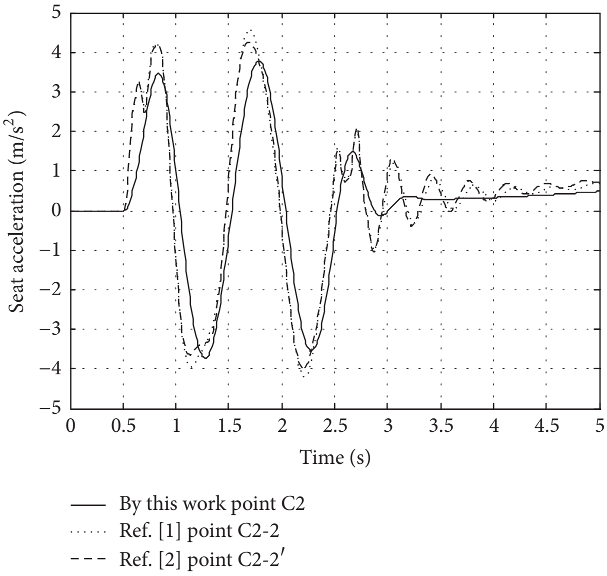


Figure 18: Time response of seat acceleration of design points C2, C2_2, and C_2′.

The results indicate that the improved multiobjective particle swarm optimization algorithm is a successful method. Moreover, MDPL-MOPSO has been used to optimally design a five-degree-of-freedom vehicle vibration model, which resulted in numerous effective trade-offs between conflicting objectives, including seat acceleration, front tire velocity, rear tire velocity, relative displacement between sprung mass and front tire, and relative displacement between sprung mass and rear tire. A comparison of the obtained results and the literature demonstrates the superiority of this work.

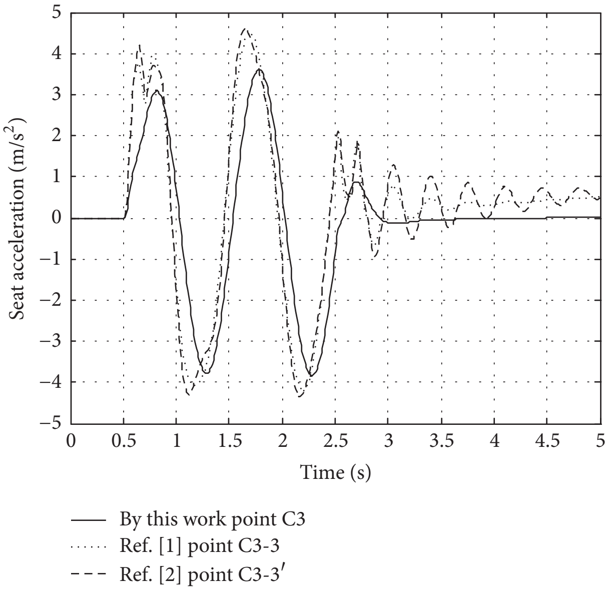


Figure 19: Time response of seat acceleration of design points C3, C3_3, and C3_3′.

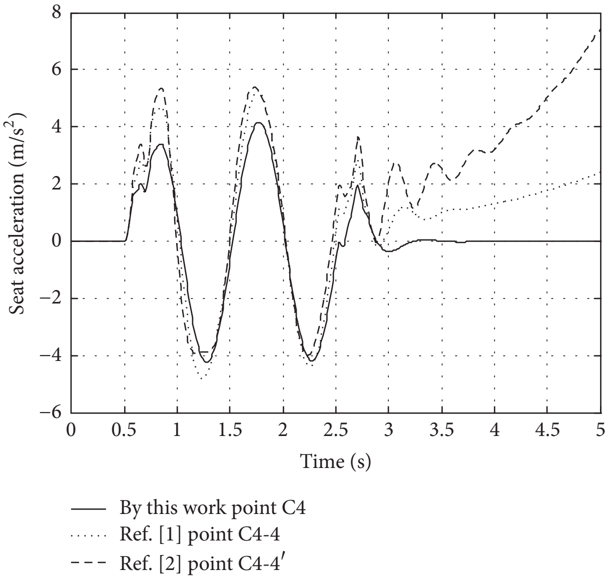


Figure 20: Time response of seat acceleration of design points C4, C4_4, and C4_4′.

Appendix

Matrices of Model

Matrices **M**, **C**, and **K** represent the mass matrix, damping matrix, and stiffness; $\mathbf{K_F}$ is the tire stiffness matrix. **M** is described by

$$M = \begin{bmatrix} m_1 & 0 & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 & 0 \\ 0 & 0 & m_s & 0 & 0 \\ 0 & 0 & 0 & m_c & 0 \\ 0 & 0 & 0 & 0 & J_s \end{bmatrix}. \quad \text{(A.1)}$$

The stiffness matrix **K** is described by

$$K = \begin{bmatrix} K_{s1} + K_{p1} & 0 & -K_{s1} & 0 & L_1 K_{s1} \\ 0 & K_{s2} + K_{p2} & -K_{s2} & 0 & -L_2 K_{s2} \\ -K_{s1} & -K_{s2} & K_{s1} + K_{s2} + K_{ss} & -K_{ss} & -L_1 K_{s1} + L_2 K_{s2} - rK_{ss} \\ 0 & 0 & -K_{ss} & K_{ss} & rK_{ss} \\ L_1 K_{s1} & -L_2 K_{s2} & -L_1 K_{s1} + L_2 K_{s2} - rK_{ss} & rK_{ss} & L_1^2 K_{s1} + L_2^2 K_{s2} + r^2 K_{ss} \end{bmatrix}. \quad \text{(A.2)}$$

The damping matrix **C** is described by

$$C = \begin{bmatrix} C_{s1} & 0 & -C_{s1} & 0 & L_1 C_{s1} \\ 0 & C_{s2} & -C_{s2} & 0 & -L_2 C_{s2} \\ -C_{s1} & -C_{s2} & C_{s1} + C_{s2} + C_{ss} & -C_{ss} & -L_1 C_{s1} + L_2 C_{s2} - rC_{ss} \\ 0 & 0 & -C_{ss} & C_{ss} & rK_{ss} \\ L_1 C_{s1} & -L_2 C_{s2} & -L_1 C_{s1} + L_2 C_{s2} - rC_{ss} & rC_{ss} & L_1^2 C_{s1} + L_2^2 C_{s2} + r^2 C_{ss} \end{bmatrix}. \quad \text{(A.3)}$$

The tire stiffness matrix $\mathbf{K_F}$ is described by

$$K_F = \begin{bmatrix} -K_{p1} & 0 & 0 & 0 & 0 \\ 0 & -K_{p2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \quad \text{(A.4)}$$

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] N. Nariman-Zadeh, M. Salehpour, A. Jamali, and E. Haghgoo, "Pareto optimization of a five-degree of freedom vehicle vibration model using a multi-objective uniform-diversity genetic algorithm (MUGA)," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 4, pp. 543–551, 2010.

[2] M. J. Mahmoodabadi, A. Adljooy Safaie, A. Bagheri, and N. Nariman-Zadeh, "A novel combination of particle swarm optimization and genetic algorithm for pareto optimal design of a five-degree of freedom vehicle vibration model," *Applied Soft Computing Journal*, vol. 13, no. 5, pp. 2577–2591, 2013.

[3] L. Weiping, W. Lei, Z. Baozhen et al., "Optimizing vehicle ride comfort based on uncertainty theory and fuzzy theory," *Mechanical Science and Technology for Aerospace Engineering*, vol. 32, no. 5, pp. 636–640, 2013.

[4] L. Pengmin, H. Limei, and Y. Jinmin, "Optimization of vehicle suspension parameters based on comfort and tyre dynamic load," *China Journal of Highway and Transport*, vol. 20, no. 1, pp. 112–117, 2007.

[5] Z. Zhonglang and L. Pengmin, "Optimization method of suspension parameters for articulated vehicle based on ride comfort and road-friendliness," *Journal of Traffic and Transportation Engineering*, vol. 9, no. 5, pp. 49–54, 2005.

[6] L. R. C. Drehmer, W. J. P. Casas, and H. M. Gomes, "Parameters optimisation of a vehicle suspension system using a particle swarm optimisation algorithm," *Vehicle System Dynamics*, vol. 53, no. 4, pp. 449–474, 2015.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[8] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the pareto archived evolution strategy," *Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.

[9] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: improving the strength Pareto evolutionary algorithm," in *Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pp. 95–100, Springer, Berlin, Germany, 2002.

[10] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks (ICNN '95)*, vol. 4, pp. 1942–1948, 1995.

[11] T. Ray and K. M. Liew, "A swarm metaphor for multiobjective design optimization," *Engineering Optimization*, vol. 34, no. 2, pp. 141–153, 2002.

[12] K. E. Parsopoulos and M. N. Vrahatis, "Particle swarm optimization method in multiobjective problems," *Frontiers in Artificial Intelligence & Applications*, vol. 76, no. 1, pp. 214–220, 2002.

[13] X. Hu and R. Eberhart, "Multi-objective optimization using dynamic neighbourhood particle swarm optimization," in *Proceedings of the IEEE International Conference on Evolutionary Computation*, vol. 2, pp. 1677–1681, 2002.

[14] C. A. C. Coello and M. S. Lechuga, "MOPSO: a proposal for multiple objective particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, pp. 1051–1056, May 2002.

[15] C. A. Coello Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 256–279, 2004.

[16] X. Hu, R. C. Eberhart, and Y. Shi, "Particle swarm with extended memory for multiobjective optimization," in *Proceedings of the 2003 IEEE Swarm Intelligence Symposium, SIS 2003*, pp. 193–197, April 2003.

[17] J. E. Fieldsend and S. Sing, "A multi-objective algorithm based upon particle swarm optimization, an efficient data structure and turbulence," in *Proceedings of the 2002 U.K. Workshop on Computational Intelligence*, pp. 37–44, Birmingham, UK, 2002.

[18] S. Mostaghim and J. Teich, "Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO)," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '03)*, pp. 26–33, Indianapolis, Ind, USA, April 2003.

[19] X. Li, "A non-dominated sorting particle swarm optimizer for multiobjective optimization," *Lecture Notes in Computer Science*, vol. 2723, pp. 37–48, 2003.

[20] C. R. Raquel and P. C. Naval Jr., "An effective use of crowding distance in multiobjective particle swarm optimization," in *Proceedings of the 7th Annual conference on Genetic and Evolutionary Computation*, pp. 257–264, ACM, June 2005.

[21] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multi-objective evolutionary algorithms: empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.