RESEARCH ARTICLE

Self-powered acceleration sensors arrayed by swarm intelligence for table tennis umpiring system

Ke Lu[1☯], Chaoran Liu[1☯*], Haiyang Zou[2☯], Yishao Wang[1☯], Gaofeng Wang[1☯], Dujuan Li[1☯], Kai Fan[1☯], Weihuang Yang[1☯], Linxi Dong[1☯*], Ruizhi Sha[3☯*], Dongyang Li[4☯]

1 Ministry of Education Engineering Research Center of Smart Microsensors and Microsystems, College of Electronics and Information, Hangzhou Dianzi University, Hangzhou, China, **2** School of Materials Science and Engineering, Georgia Institute of Technology, Atlanta, GA, United States of America, **3** State Key Laboratory for Manufacturing Systems Engineering, System Engineering Institute, Xi'an Jiaotong University, Xi'an, China, **4** Department of Electronics and Information Engineering, Tongji University, Shanghai, China

☯ These authors contributed equally to this work.
* liucr@hdu.edu.cn (CL); donglinxi@hdu.edu.cn (LD); ruizhi.sha@gmail.com (RS)

Abstract

Table tennis competition is voted as one of the most popular competitive sports. The referee umpires the competition mainly based on visual observation and experience, which may make misjudgments on competition results due to the referee's subjective uncertainty or imprecision. In this work, a novel intelligent umpiring system based on arrayed self-powered acceleration sensor nodes was presented to enhance the competition accuracy. A sensor node array model was established to detect ball collision point on the table tennis table. This model clearly illuminated the working mechanism of the proposed umpiring system. And an improved particle swarm optimization (level-based competitive swarm optimization) was applied to optimize the arrayed sensor nodes distribution by redefining the representations and update rules of position and velocity. The optimized results showed that the number of sensors decreased from 58 to 51. Also, the reliability of the optimized nodes distribution of the table tennis umpiring system has been verified theoretically. The results revealed that our system achieved a precise detection of the ball collision point with uniform error distances below 3.5 mm. Besides, this research offered an in-depth study on intelligent umpiring system based on arrayed self-powered sensor nodes, which will improve the accuracy of the umpiring of table tennis competition.

Citation: Lu K, Liu C, Zou H, Wang Y, Wang G, Li D, et al. (2022) Self-powered acceleration sensors arrayed by swarm intelligence for table tennis umpiring system. PLoS ONE 17(10): e0272632. https://doi.org/10.1371/journal.pone.0272632

Editor: Wei Hong Lim, UCSI University, MALAYSIA

Received: April 11, 2022

Accepted: July 25, 2022

Published: October 17, 2022

Data Availability Statement: All relevant data are within the paper.

Funding: This work was financial supported by the National Natural Science Foundation of China (No. 62111530298, 61871167, U1909221, 61804038), The funders of this work are Professor Chaoran Liu and Professor Linxi Dong.

Competing interests: The authors have declared that no competing interests exist.

Introduction

As one of the hot events in the Olympic Games, table tennis competition has got increasing attention from all countries. However, the umpire in table tennis competition is still entirely determined by the professional ability and subjective judge of the referee [1, 2]. Table tennis competition has the characteristics of fast speed and small ball. It is hard for the referee to make accurate umpire when an edge ball or net ball appears [3]. Furthermore, in a highly strained mental state, professional referees are also prone to mental fatigue, which may lead them to make misjudgments. The collision point of table tennis on the tabletop is a key reference factor for referee to make the final umpire [4]. Therefore, precise positioning of collision point will effectively improve the accuracy of judgment.

Hawk-eye technology can be introduced into table tennis competition to accurately locate the collision point (CP) on the table caused by a ball, thereby reducing misjudgments. But it will cause long pauses, which may impact athlete's state [5]. Sensor network has been proved as an effective positioning system with potential applications in many fields, such as military affairs [6–8] and civil engineering [9–11]. To improve the sensing efficiency of the sensor and save costs, the sensors distribution optimization has been presented. Many approaches have been developed to optimize the sensors deployment, in which the sensors distribution based on particle swarm optimization (PSO) gains wide attentions [12–19]. PSO was first proposed by Eberhart and Kennedy [20] in 1995 to solve optimization problems. It is a population-based algorithm inspired from the behaviors of social animals such as birds. The individual in PSO is called particle and each particle has a position and velocity. The position of a particle is viewed as a candidate solution for the target optimization problem. Owing to simplicity and efficiency, PSO has become a vital tool to solve optimization problems [21, 22]. It has been successfully implemented to solve various optimization problems for decades, especially for those intractable NP-hard issues [23]. Unfortunately, the classical PSO easily leads to stagnation or premature convergence [24].

In this work, we presented a real-time and accurate intelligent umpiring system based on arrayed self-powered acceleration sensors to assist referee make precise umpiring. Firstly, a model was established to expound the designed umpiring system which was used to detect the collision point of the ball on the table tennis table. To simplify the arrayed sensor nodes, save costs and optimize nodes distribution, we presented a level-based competitive swarm optimization (LCSO). The representations and update rules of position and velocity could be redefined in LCSO. By LCSO algorithm, the number of sensors reduced from 58 to 51. Finally, we verified the reliability of optimized nodes distribution of the table tennis umpiring system theoretically. This research provided a new strategy for umpire of table tennis competitions, which would make umpiring precise and real-time.

Model

When two dissimilar materials come into contact, their surfaces will generate positive and negative electrostatic charges due to the contact electrification respectively. When the two materials are separated due to the action of mechanical force, the positive and negative charges generated by the contact electrification are also separated. And this charge separation will correspondingly generate an induced potential difference between the upper and lower electrodes of the material. If a load is connected between the two electrodes or the two electrodes are short circuit, the induced potential difference will drive electrons to flow between the two electrodes through an external circuit—this is triboelectric nanogenerator (TENG).

The referee umpires the score mainly based on the ball collision point (CP) on the tabletop [5] in a table tennis competition. It is hard for human eyes to precisely umpire when touching net or edge ball appears. Here, we present an intelligent collision point positioning system by distributing the acceleration sensor on the back surface of the table (Fig 1). We prefer the self-powered acceleration sensor [25–27] based on TENG to overcome battery shortcomings such as limited power storage, maintenance required, and large weight. The bottom-up components of the sensor consist of a TENG, an insulating layer, and a mass (Fig 1A). As the table tennis

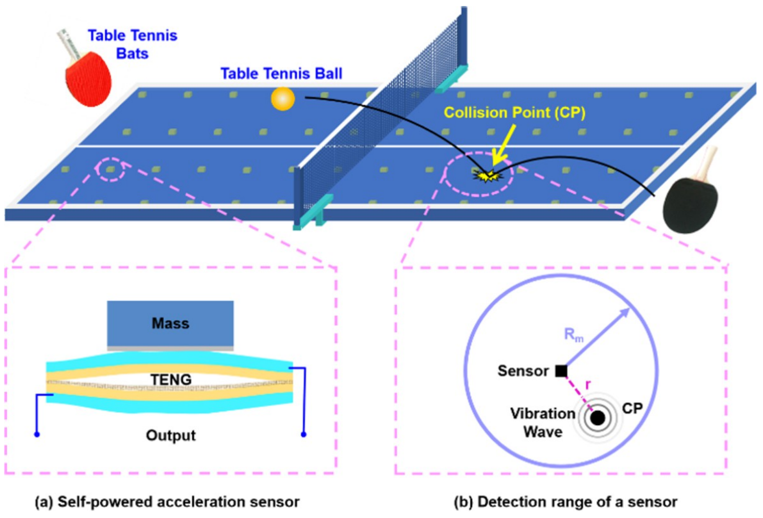


Fig 1. Schematic of the intelligent umpiring system by deploying the self-powered acceleration sensor on the back of the table. (a) Schematic of the self-powered acceleration sensor. (b) Detection range of a sensor with a maximum detection distance.

https://doi.org/10.1371/journal.pone.0272632.g001

hits the tabletop, the acceleration sensor will detect a tabletop vibration wave and output a voltage signal.

The sensor has a maximal detection distance (R_m). Within the detection range of the sensor, the vibration of any collision point can be detected (Fig 1B). Obviously, more than three sensor nodes are required to uniquely determine a collision point on a two-dimensional plane. One sensor cannot determine the location of the collision point (Fig 2A), while two sensors have a 50% chance to accurately locate the collision point (Fig 2B). Four or more sensors are wasteful, complicating the system and increasing cost (Fig 2D). Therefore, three sensors are the best.

The sensor nodes distribution should be optimized to simplify and efficiently detect all possible collision points (CPs) on the entire tabletop. Assuming that infinitely many sensor nodes are installed on the back of the table, a CP position must be located inside a triangle formed by the three nearest sensors. And the distances between the CP and each of the three nearest sensors are less than R_m of sensor. We define the three nearest sensors as the detection sensor set (DSS). It is obvious that different CPs may have the same DSS. And these CPs with the same DSS will completely fill in the triangle formed by their DSS. Based on the assumption, we regard the tabletop as a 2-D rectangle composed of triangles. When we try to expand these triangles synchronously, the number of sensors installed on the back of the table will gradually decrease. The triangle cannot be expanded infinitely because the distance between any CP in triangle and vertexes must be less than R_m. Therefore, the sensor distribution optimization can be converted as the optimization of the triangle area.

Commonly, the distances between any point in a triangle and vertexes are less than the length of the triangle longest edge. The triangles cannot cover the maximum area on the tabletop if they are not isosceles triangles. Therefore, we reduce the number of sensor nodes by optimizing the shape of isosceles triangle. Furthermore, the included angles α and β were explored to optimize the sensor distribution (Fig 3A).

Based on the above theories, the sensor distribution optimization can be modeled as a function: $minsensor = f(\alpha, \beta, x, y)$. α is the base angle of the isosceles triangle, which determines the area of the triangle. β is the included angle between the waist of the isosceles triangle and the edge of the table (Fig 3A). x and y are abscissa and ordinate of the initial sensor respectively.

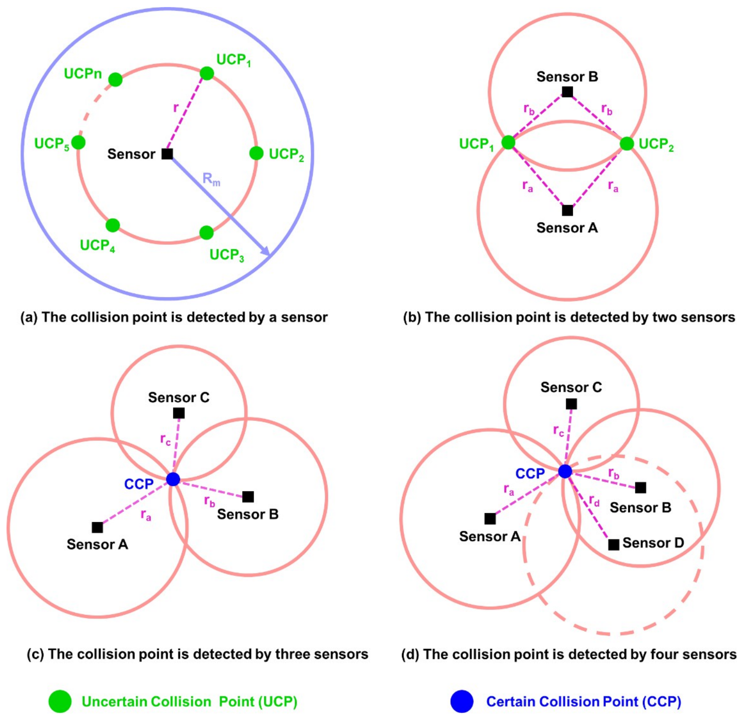

UCPn
UCP$_1$
UCP$_5$
Sensor
UCP$_2$
r
R$_m$
UCP$_4$
UCP$_3$
(a) The collision point is detected by a sensor
Sensor B
Sensor A
UCP$_1$
UCP$_2$
r$_b$
r$_a$
(b) The collision point is detected by two sensors
Sensor C
Sensor B
Sensor A
CCP
r$_a$
r$_b$
r$_c$
(c) The collision point is detected by three sensors
Sensor C
Sensor B
Sensor A
Sensor D
CCP
r$_a$
r$_b$
r$_c$
r$_d$
(d) The collision point is detected by four sensors
Uncertain Collision Point (UCP)
Certain Collision Point (CCP)

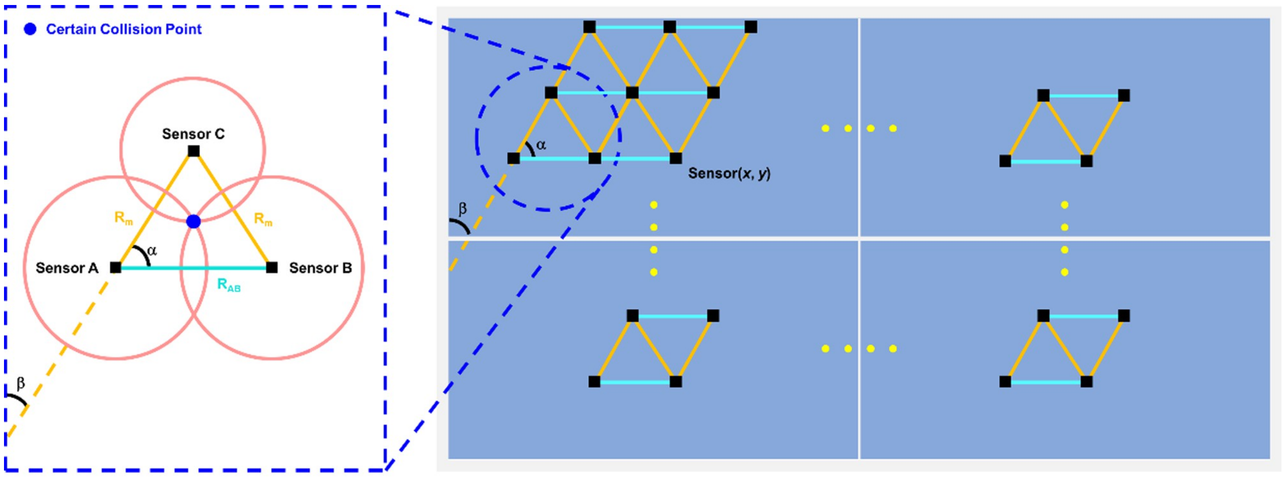
Fig 2. The possible location of the collision point when the point is detected by (a) a sensor, (b) two sensors, (c) three sensors and (d) four sensors at the same time, respectively.

https://doi.org/10.1371/journal.pone.0272632.g002

With the function *minsensor*, we will get the positions of other sensors and the number of sensors inside the table after obtaining the angles of α and β and the abscissa x and ordinate y of initial sensor (Fig 3B). The pseudo code of the function *minsensor* = $f(\alpha, \beta, x, y)$ is shown as following:

Algorithm 1: The function *minsensor* = *f* (*α*, *β*, *x*, *y*)

```
Coordinate system: Define the lower left of the table tennis
  table as the origin and the edge of the table as the coordi-
  nate axis to establish a rectangular coordinate system. (0 mm
  ≤ $x$ ≤ 2740 mm and 0 mm ≤ $y$ ≤ 1525 mm)
Input: base angle $\alpha$, intersection angle $\beta$, the initial sensor($x$,
  $y$)
Output: the number of sensors inside the table tennis table
  //sensors: store the coordinates of all sensors located inside
  the table;
01: Define global variables: sensors;
  //$U$ denotes a set of sensors;
02: $U = \varnothing$;
03: Store initial sensor($x$,$y$) into $U$;
04: while $U \neq \varnothing$ do
```



(a) Isosceles triangle formed by sensors

(b) Obtaining other sensors by *α*, *β* and initial sensor(*x*,*y*).

Fig 3. (a) Isosceles triangle formed by the three nearest sensors of the collision point. (b) Obtaining the position of all sensors in the table by *α*, *β* and initial sensor(*x*,*y*).

https://doi.org/10.1371/journal.pone.0272632.g003

```
05:  randomly select a sensor(x,y) from U;
06:  remove sensor(x, y) from U;
    //The sensor cannot be located outside the table
07:  if x > 2740 or x < 0 or y > 1525 or y < 0 then
08:    if x > 2740 then
09:      x = 2740;
10:    end if
11:    if x < 0 then
12:      x = 0;
13:    end if
14:    if y > 1525 then
15:      y = 1525;
16:    end if
17:    if y < 0 then
18:      y = 0;
19:    end if
20:    if sensor(x, y) is not the member of sensors then
21:      Store sensor(x, y) into sensors;
22:    end if
23:  else
24:    if sensor(x, y) is not the member of sensors then
```

```
            Store sensor($x$, $y$) into $sensors$;
            Calculate the coordinates of the six sensors around sensor($x$, $y$) based on $\alpha, \beta$ and $R_m$;
            Store these six sensors into $U$;
        end if
    end if
end while
if sensor(0, 0) is not the member of $sensors$ then
    Store sensor(0, 0) into $sensors$;
end if
if sensor(0, 1525) is not the member of $sensors$ then
    Store sensor(0, 1525) into $sensors$;
end if
if sensor(2740, 0) is not the member of $sensors$ then
    Store sensor(2740, 0) into $sensors$;
end if
if sensor(2740, 1525) is not the member of $sensors$ then
    Store sensor(2740, 1525) into $sensors$;
end if
return $sensors$ and the size of $sensors$;
```

Level-based competitive swarm optimizer

The idea of particle swarm optimization (PSO) originates from the movement of birds, where PSO can search the decision space for promising solutions just like birds searching for food [28]. Each particle in PSO has two attributes (position and velocity). These two attributes represent current feasible solution and moving direction of the particles, respectively. In the optimization process, the velocity and position of each particle are iteratively updated by using the following equations:

$$\begin{aligned} V_i(t+1) = \omega V_i(t) + c_1 r_1(t)(Pbest_i(t) - X_i(t)) \\ + c_2 r_2(t)(Gbest(t) - X_i(t)) \end{aligned} \tag{1}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{2}$$

Where t is the current number of iterations, $X_i = (x_i^1, x_i^2, \ldots, x_i^D)$ and $V_{\mathrm{i}} = (v_i^1, v_i^2, \ldots, v_i^D)$ are the position and the velocity of the *i-th* particle, respectively. $Pbest_i = (pbest_i^1, pbest_i^2, \ldots, pbest_i^D)$ is the best position where *i-th* has achieved so far, $Gbest = (gbest^1, gbest^2, \ldots gbest^D)$ represents the current best position in the particles swarm. D denotes the dimension size of the objective function, ω is termed as the inertia weight [29]. c_1 and c_2 are acceleration coefficients adjusted by users. r_1 and r_2 are two random number generated within (0, 1).

Refer to Eqs (1) and (2), a particle iteratively updates its position and traverses the search space to seek the potential optimum of the objective function. However, this learning strategy commonly causes stagnation and premature convergence. The reason is that PSO shows weak capacity of diversity preservation [30]. To address this issue, a large amount of improvements on PSO can be found in literatures [31–39]. The current methods can be mainly classified into the following two categories, adopting cooperatively coevolutionary framework and designing new learning strategies. Methods in the first category focus on the large-scale optimization problems, which divides the original problem into several simultaneous and independently optimized sub-problems, such as the random variable grouping-based algorithms [40], differential grouping-based algorithms [41, 42] and machine learning based algorithms [43]. However, such methods commonly have the problems with high computational cost and inaccurate variable grouping. Methods in the second category aim to improve the search ability of PSO under different learning strategies, including the comprehensive learning and competitive learning for diversity enhancement [31], adaptive parameter adjustments and level-based learning [44] for balancing the convergence and diversity [45]. Nevertheless, these methods still cannot help PSO to effectively address complex problems, since the diversity preservation continue challenging PSO.

Among these methods, CSO [46] and DLLSO [44] are two competitive algorithms proposed in recent years to address optimization problems with large scale dimensionality. With a deep insight into these two algorithms, we find that CSO and DLLSO show several special advantages when comparing with PSO. CSO selects the exploitation exemplars by using a pairwise competition mechanism. And it only updates half of the particles swarm at each generation, which can maximize the diversity of the exemplars and preserve the promising solutions. On the other hand, the level-learning strategy proposed in DLLSO can further diverse the exploration exemplars when compared with CSO. Inspired by CSO and DLLSO, a novel particle swarm optimization method is proposed in this paper. In the proposed method, neither the personal best position of each particle (*pbest*) nor the global best position (*gbest*) is involved in updating the particles swarm. Conversely, we introduce a level-based pairwise competition mechanism. In general, the particles swarm will be divided into several levels according to the fitness, similar to the design in DLLSO. Therefore, the particles in higher levels hold more beneficial information which is contributing to guide the particles swarm toward the global optimum area [47, 48]. More specifically, the proposed pairwise competition mechanism is executed in each level to distinguish the particles that need to be updated and the corresponding exploration exemplars. On the other hand, the particles in higher levels should be selected as the exploitation exemplars to guide particles in lower levels to search the solution space for exploitation.

The framework of LCSO is shown in Fig 4. Firstly, particles in the swarm are sorted by fitness in ascending order and divide into several levels. The better particles belong to higher levels and all levels have the same number of particles (L_1 is the highest level). Lower-level particles can learn from higher-level particles.

The number of candidate exemplars for particles in higher level is fewer, which matches the expectation that better particles should do more exploitation rather than exploration. Then, in each level, two particles are randomly chosen to contest until all particles have participated in at least one competition. After each competition, the winner that having a better fitness will be passed directly to the next generation particles swarm, and the loser will update its position and velocity by learning from the winner and higher-level particles. Since the particles in the first level are the best of the whole particles swarm in the current generation, better solutions are usually found near these ones. The particles in the highest level directly enter the next generation. The position and velocity of the particle that loses the competition will update as

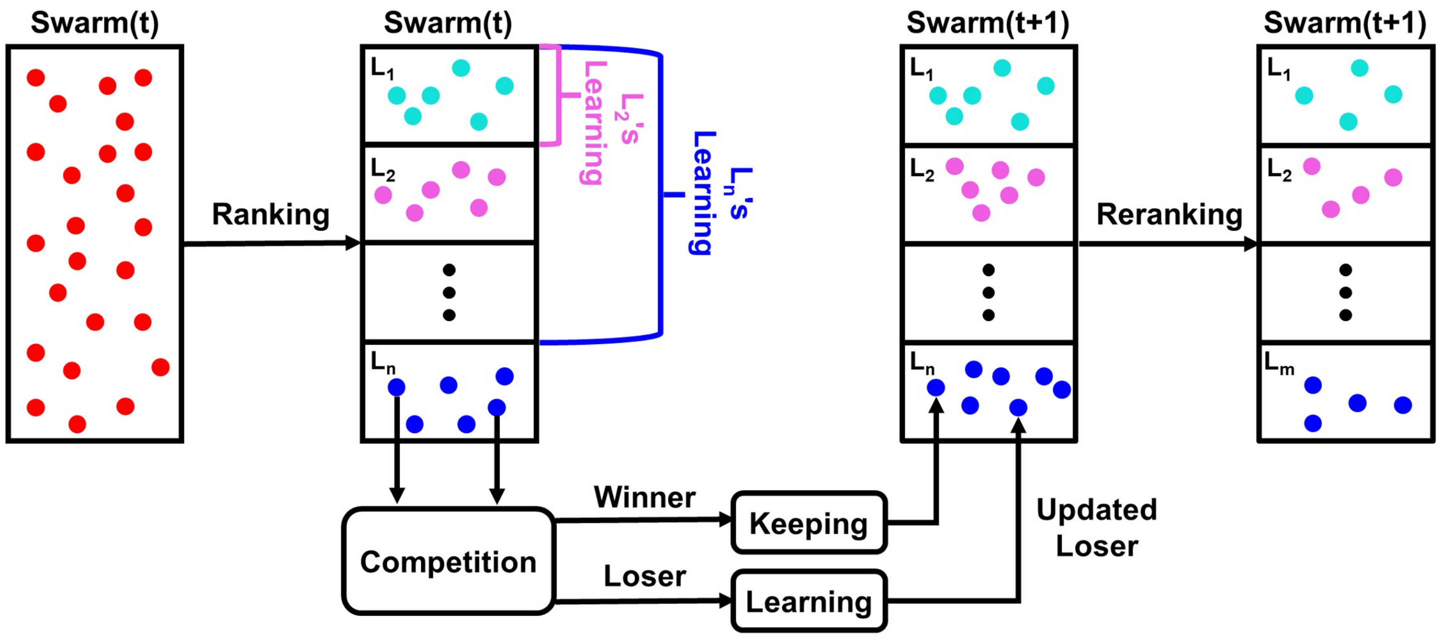


Fig 4. Framework of the proposed level-based competitive swarm optimization.

https://doi.org/10.1371/journal.pone.0272632.g004

follows:

$$V_{i,j}(t+1) = r_1(t)V_{i,j}(t) + r_2(t)\left(X_{m,n}(t) - X_{i,j}(t)\right) + \phi r_3(t)\left(X_{i,k}(t) - X_{i,j}(t)\right) \quad (3)$$

$$X_{i,j}(t+1) = X_{i,j}(t) + V_{i,j}(t+1) \quad (4)$$

Where t is the iteration number, $X_{i,j} = (x_{i,j}^1, x_{i,j}^2, \ldots, x_{i,j}^D)$ and $V_{i,j} = (v_{i,j}^1, v_{i,j}^2, \ldots, v_{i,j}^D)$ are the position vector and the velocity vector of the *j-th* loser particle from the *i-th* level (L_i) respectively. $X_{m,n} = (x_{m,n}^1, x_{m,n}^2, \ldots, x_{m,n}^D)$ is the randomly selected exemplars from *m-th* level (L_m) ($m\in[1,i\text{-}1]$), and $X_{i,k} = (x_{i,k}^1, x_{i,k}^2, \ldots, x_{i,k}^D)$ represents the winner after competing with $X_{i,j}$. $r_1(t)$, $r_2(t)$, and $r_3(t)$ are three random variables in the range [0, 1] and ϕ is the control parameter within [0, 1] which is in charge of the influence of the second exemplar.

The pseudo code of LCSO is outlined in Algorithm 2.

Algorithm 2: Framework of LCSO

```
Input: swarm P, swarm size NP, number of levels N, level size
  LS, maximum number of fitness evaluations MAX_FES.
Output: the final solution x and its fitness f(x)
  //t denotes the generation number;
01: t = 0;
02: fes = 0;
03: Initialize the swarm P(0) randomly;
```

```
04: while $fes < MAX\_FES$ do
05:   calculate the fitness of all particles in $P(t)$;
06:   $fes += NP$;
      //At each generation, $N$ may different;
07:   Sort particles in ascending order of fitness and divide them into $N$ levels;
      //Update particles in $L_N, \ldots, L_2$;
08:   for i = {$N, \ldots, 2$} do
        //$U_i$ denotes a set of $i$-th level particles that have not yet participated in a competition;
09:     while $U_i \neq \Phi$ do
10:       randomly choose two particles $X_1(t)$, $X_2(t)$ from the $i$-th level $L_i$;
11:       if $f(X_1(t)) < f(X_2(t))$ then
12:         $X_w(t) = X_1(t)$, $X_1(t) = X_2(t)$;
13:       else
14:         $X_w(t) = X_2(t)$, $X_1(t) = X_1(t)$;
15:       end if
16:       add $X_w(t)$ into $P(t + 1)$;
17:         Select a level from the top ($i$-1) levels: $L_m$;
18:       Randomly select a particles from $L_m$: $X_{m,n}$;
19:         Update particle $X_1(t)$ refer to (3) and (4);
20:       add the updated $X_1(t)$ to $P(t + 1)$;
21:       remove $X_1(t)$, $X_2(t)$ from $U_i$;
22:     end while
23:   end for
24:   add the first-level particles into $P(t + 1)$;
25:   $t = t + 1$;
26: end while
```

The difference between LCSO and PSO and its other variants is the proposed level-based pairwise competition mechanism (LCM) of LCSO. Firstly, LCM divides the swarm into several levels based on particles' fitness. Then, for each particle that lose the pairwise competition, two exemplars are selected from higher level particles and the winner of the pairwise competition to guide its updating for exploitation and exploration.

The complexity, exploration and exploitation of LCSO are theoretically analyzed as follows.

Computational complexity

Since the algorithm usually takes the number of fitness evaluations as the final criterion, this paper assumes using the same swarm size N to compare the computational complexity per generation between the proposed algorithm and PSO. Algorithm structure, computation cost at each generation and the required memory cost are taken into consideration to compare the computational complexity of LCSO and PSO.

Firstly, LSCO has the same structure as PSO, which is easy to implement. Secondly, the extra computational cost of LCSO at each generation are generated in Lines 7–18 in Algorithm 2 to divide the swarm into N levels and execute competition, which both cost O(N). In summary, LCSO only introduces O(2N) extra computational cost at each generation compared with PSO. Last but not least, LCSO has greater advantage in memory cost when comparing with PSO as it does not need to store the historical best particles.

In summary, LCSO is superior in terms of implementation and computational complexity.

Exploration ability

Exploration is crucial to Evolutionary Algorithms (EAs) since it is conducive to find more promising areas in the decision space to avoid premature convergence. One way to enhance the exploration ability of EAs is maintenance of swarm diversity, which can be achieved by diversing the exemplars of particles. The exploration exemplars in LCSO are selected by using a pairwise competition, indicating that all updated particles have different exploration exemplars. However, the exploration exemplars in DLLSO are randomly selected. Therefore, it can be concluded that the exploration exemplars in LCSO have more diversity than that in DLLSO, CSO and the canonical PSO.

Further, we rewrite Eq (3) as Eqs (5–7):

$$V_{i,j}(t+1) = r_1(t)V_{i,j}(t) + \theta_1\left(p_1 - X_{i,j}(t)\right) \tag{5}$$

$$\theta_1 = r_2(t) + \phi r_3(t) \tag{6}$$

$$p_1 = \frac{r_2(t)}{r_2(t)+\phi r_3(t)}X_{m,n}(t) + \frac{\phi r_3(t)}{r_2(t)+\phi r_3(t)}X_{i,k}(t) \tag{7}$$

Similarly, the equation of updated velocity of classical PSO, CSO and DLLSO can be rewritten to (8), (9), (10) respectively.

$$\begin{cases} V_i(t+1) = \omega V_i(t) + \theta_2(p_2 - X_i(t)) \\ \theta_2 = c_1r_1(t) + c_2r_2(t) \\ p_2 = \dfrac{c_1r_1(t)}{c_1r_1(t)+c_2r_2(t)}Pbest_i(t) + \dfrac{c_2r_2(t)}{c_1r_1(t)+c_2r_2(t)}Gbest(t) \end{cases} \tag{8}$$

$$\begin{cases} V_l(t+1) = r_1(t)V_l(t) + \theta_3(p_3 - X_l(t)) \\ \theta_3 = r_2(t) + \phi r_3(t) \\ p_3 = \dfrac{r_2(t)}{r_2(t)+\phi r_3(t)}X_w(t) + \dfrac{\phi r_3(t)}{r_2(t)+\phi r_3(t)}\bar{X}(t) \end{cases} \tag{9}$$

$$\begin{cases} V_{i,j}(t+1) = r_1(t)V_{i,j}(t) + \theta_4\Big(p_4 - X_{i,j}(t)\Big) \\ \theta_4 = r_2(t) + \phi r_3(t) \\ p_4 = \dfrac{r_2(t)}{r_2(t)+\phi r_3(t)} X_{rl_1,k_1}(t) + \dfrac{\phi r_3(t)}{r_2(t)+\phi r_3(t)} X_{rl_2,k_2}(t) \end{cases} \tag{10}$$

Where $X_w = (x_w^1, x_w^2, \ldots, x_w^D)$ is the position of the winner and $\bar{X} = (\bar{x}^1, \bar{x}^2, \ldots, \bar{x}^D)$ is the average position of the particles swarm in CSO. And in DLLSO, $X_{rl1,k1} = (x_{rl1,k1}^1, x_{rl1,k1}^2, \ldots, x_{rl1,k1}^D)$ randomly selected from *rl1-th* level (L_{rl1}) and $X_{rl2,k2} = (x_{rl2,k2}^1, x_{rl2,k2}^2, \ldots, x_{rl2,k2}^D)$ randomly selected from *rl2-th* level (L_{rl2}) are the two selected exemplars.

From Eqs (5), (8), (9) and (10), the difference between p_i (i = 1, 2, 3, 4) and the particle to be updated is the main source of diversity. As for classical PSO (p_2), *Pbest* of each particle is updated only when the particle finds a better position and *Gbest* is updated only when the swarm finds a better position. It indicates that *Pbest* and *Gbest* may be constant for many generations, which limits exploration ability. In CSO (p_3), the mean position of the swarm is employed as the exploration exemplar of all the particles. Although the mean position is updated at each generation, it is shared by all particles, which is not beneficial for further diversity enhancement. For DLLSO (p_4) based the level-learning strategy, the exploration exemplar $X_{rl2,k2}$ is randomly selected from higher levels for each particle in lower levels. However, a higher-level particle may be chosen as the exploration exemplar of multiple lower-level particles. In our proposed LCSO (p_1), a level-based pairwise competition mechanism is utilized to select the exploration exemplars, where the exploration exemplars are the winners of each pairwise competition in the same level. This mechanism can ensure that all the exploration exemplars are different.

In summary, the diversity preservation ability of the proposed LCSO is better than DLLSO, CSO, and PSO, which benefits for strengthening the exploration ability.

Exploitation ability

Exploitation is a key point for swarm to refine promising areas, especially when computational resources are limited. Generally, the exploitation ability can be reflected by the difference between the exploitation exemplars and the corresponding updated particles.

The first exemplars in LCSO and DLLSO are the exploitation exemplars, which are randomly selected from the higher level. The differences of the exploitation exemplar selections between LCSO and DLLSO are presented as follows. The exploitation exemplar for a particle in DLLSO is selected from a level that at least two levels higher than the updated particle. However, the exploitation exemplars in LCSO are selected from any higher level. This indicates that LCSO can exploit the exploitation exemplars within a smaller difference when comparing with DLLSO, which is potential in the exploitation of a local area. Similar to the analyses in [39], LCSO theoretically has a faster convergence ability than CSO and classical PSO.

The above analyses clearly verify that LCSO shows superior exploration and exploitation abilities.

To verify the superiority and efficiency of the proposed LCSO, we have compared LCSO with five representative PSO variants [37, 44–46, 49] by a series of experiments conducted on the CEC'2013 benchmark sets [50] (Table 1). For fairness, the key parameters in each algorithm are set according to the suggestions in the corresponding papers. The symbols above the p values, "+", "-", and "=", mean that the performance of LCSO is significantly better than, worse than, or statistically equivalent to the compared algorithm on the corresponding benchmark respectively (Table 2). In addition, the bold p means that LCSO is significantly better than other

Table 1. Summary of the 28 CEC'13 test functions.

No.	Functions	Unimodal/Multimodal	Non-separable/Separable
1	Sphere Function	Unimodal	Separable
2	Rotated High Conditioned Elliptic Function	Unimodal	Non-separable
3	Rotated Bent Cigar Function	Unimodal	Non-separable
4	Rotated Discus Function	Unimodal	Non-separable
5	Different Powers Function	Unimodal	Separable
6	Rotated Rosenbrock's Function	Multimodal	Non-separable
7	Rotated Schaffers F7 Function	Multimodal	Non-separable
8	Rotated Ackley's Function	Multimodal	Non-separable
9	Rotated Weierstrass Function	Multimodal	Non-separable
10	Rotated Griewank's Function	Multimodal	Non-separable
11	Rastrigin's Function	Multimodal	Separable
12	Rotated Rastrigin's Function	Multimodal	Non-separable
13	Non-Continuous Rotated Rastrigin's Function	Multimodal	Non-separable
14	Schwefel's Function	Multimodal	Non-separable
15	Rotated Schwefel's Function	Multimodal	Non-separable
16	Rotated Katsuura Function	Multimodal	Non-separable
17	Lunacek Bi_Rastrigin Function	Multimodal	--
18	Rotated Lunacek Bi_Rastrigin Function	Multimodal	Non-separable
19	Expanded Griewank's plus Rosenbrock's Function	Multimodal	Non-separable
20	Expanded Scaffer's F6 Function	Multimodal	Non-separable
21	Composition Function 1 (n = 5,Rotated)	Multimodal	Non-separable
22	Composition Function 2 (n = 3,Unrotated)	Multimodal	Separable
23	Composition Function 3 (n = 3,Rotated)	Multimodal	Non-separable
24	Composition Function 4 (n = 3,Rotated)	Multimodal	Non-separable
25	Composition Function 5 (n = 3,Rotated)	Multimodal	Non-separable
26	Composition Function 6 (n = 5,Rotated)	Multimodal	Non-separable
27	Composition Function 7 (n = 5,Rotated)	Multimodal	Non-separable
28	Composition Function 8 (n = 5,Rotated)	Multimodal	Non-separable

https://doi.org/10.1371/journal.pone.0272632.t001

algorithms. Furthermore, w/l/t at the bottom of the table represents the times that LCSO wins/loses/ties in the competitions compared with the corresponding algorithms. Comparing with five state-of-the arts algorithms, LCSO outperforms the compared algorithms on most of the 28 functions, which clearly demonstrates that the proposed LCSO has a better performance.

Sensors arrayed by proposed LCSO

To optimize the umpiring system, the number of sensor nodes should be minimized by LCSO under the premise that the detection range of the designed intelligent umpiring system covers the entire tabletop. The maximum detection distance (R_m) of the sensor is a key factor for the final result of sensor array optimization, which is decided by the resolution of the sensor and the minimal vibration caused by the table tennis hitting the table during the competition. Obviously, the table will generate a minimal vibration when the ball hits the top of the net and then drops in free fall. The instantaneous velocity (v_1) of the ball when colliding with the table can be defined as:

$$v_1 = \sqrt{2gh} \tag{11}$$

Table 2. Comparison results of the compared algorithms on CEC'2013 functions with 6 × 105 fitness evaluations.

Function	Quality	CSO	DLLSO	LIPS	HPSO_TVAC	ALC-PSO	LCSO
F1	Mean	2.28E-13	6.37E-13	8.71E-01	2.72E-08	7.47E-12	2.27E-13
	Std	3.74E-16	4.14E-13	2.07E+00	9.92E-09	5.78E-12	1.02E-13
	p-value	**5.00E-03^{+}**	**1.38E-08^{+}**	**5.35E-10^{+}**	**5.35E-10^{+}**	**5.25E-10^{+}**	-
F2	Mean	3.17E+06	1.76E+07	4.09E+08	2.65E+07	1.32E+08	3.10E+06
	Std	7.43E+05	4.61E+06	1.62E+08	6.23E+06	5.30E+07	8.66E+05
	p-value	5.61E-01$^{=}$	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F3	Mean	1.33E+08	9.15E+09	3.24E+13	2.16E+10	4.64E+10	4.25E+08
	Std	1.40E+08	4.25E+09	3.22E+13	6.23E+09	2.49E+10	3.09E+08
	p-value	6.75E-06^{-}	**1.42E-09^{+}**	**1.41E-09^{+}**	**1.41E-09^{+}**	**1.41E-09^{+}**	-
F4	Mean	8.76E+04	7.35E+04	2.52E+05	6.66E+04	5.20E+04	2.33E+04
	Std	9.42E+03	1.92E+04	2.14E+04	1.12E+04	8.44E+03	4.54E+03
	p-value	**1.41E-09^{+}**	**1.60E-09^{+}**	**1.41E-09^{+}**	**1.41E-09^{+}**	**2.02E-09^{+}**	-
F5	Mean	1.14E-13	1.03E-05	14.46E+02	2.77E-04	3.02E-12	3.55E-13
	Std	3.13E-16	6.62E-13	1.78E+01	4.93E-05	4.76E-13	2.70E-13
	p-value	4.16E-11^{-}	**7.25E-10^{+}**	**7.40E-10^{+}**	**7.40E-10^{+}**	**7.18E-10^{+}**	-
F6	Mean	2.00E+02	2.73E+02	2.95E+03	2.40E+02	6.83E+02	1.99E+02
	Std	3.38E+01	4.16E+01	1.01E+03	4.91E+01	1.55E+02	3.17E+01
	p-value	**1.41E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F7	Mean	1.10E+01	6.90E+01	2.66E+03	5.17E+04	1.80E+02	2.00E+00
	Std	2.76E+00	1.48E+01	1.83E+03	4.18E+04	7.01E+01	7.63E+00
	p-value	**1.41E-09^{+}**	**1.60E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F8	Mean	2.20E+01	2.10E+01	2.20E+01	2.20E+01	2.20E+01	2.10E+01
	Std	3.36E-02	6.31E-02	2.77E-02	2.37E-02	3.20E-02	3.63E-02
	p-value	**5.01E-05^{+}**	5.35E-01$^{=}$	**1.17E-02^{+}**	**1.30E-02^{+}**	8.77E-01$^{=}$	-
F9	Mean	4.20E+01	5.90E+01	1.18E+02	1.37E+02	1.26E+02	2.70E+01
	Std	4.57E+00	4.23E+00	5.54E+00	5.55E+00	8.02E+00	6.14E+00
	p-value	**1.60E-03^{+}**	**2.21E-07^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F10	Mean	1.64E-01	2.50E+01	1.37E+03	2.60E+01	1.20E+01	1.28E-01
	Std	1.09E-01	1.12E+01	1.32E+03	6.57E+00	9.21E+00	6.15E-02
	p-value	4.15E-01$^{=}$	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F11	Mean	5.20E+01	1.18E+02	7.45E+02	3.22E+02	2.57E+02	7.20E+01
	Std	7.63E+00	2.23E+01	9.39E+01	5.53E+01	2.55E+01	1.47E+01
	p-value	2.30E-06^{-}	**8.28E-09^{+}**	**1.41E-09^{+}**	**1.41E-09^{+}**	**1.41E-09^{+}**	-
F12	Mean	8.04E+02	1.40E+02	7.99E+02	1.72E+03	6.70E+02	7.91E+02
	Std	1.87E+01	2.42E+01	1.02E+02	1.57E+02	1.45E+02	1.71E+01
	p-value	**9.90E-03^{+}**	1.41E-09^{-}	7.415E-01$^{=}$	**1.41E-09^{+}**	3.31E-04^{-}	-
F13	Mean	8.01E+02	3.75E+02	1.30E+03	2.11E+03	1.05E+03	7.84E+02
	Std	1.42E+01	6.19E+01	1.15E+02	1.88E+02	1.58E+02	2.34E+01
	p-value	**3.00E-03^{+}**	1.42E-09^{-}	**1.42E-09^{+}**	**1.42E-09^{+}**	**7.38E-09^{+}**	-
F14	Mean	1.45E+03	5.19E+03	1.11E+04	3.77E+03	6.58E+03	2.31E+03
	Std	3.34E+02	8.15E+02	5.92E+02	7.13E+02	7.03E+02	6.37E+02
	p-value	4.67E-06-	**2.03E-09^{+}**	**1.42E-09^{+}**	**1.31E-07^{+}**	**1.42E-09^{+}**	-
F15	Mean	2.92E+04	9.30E+03	1.41E+04	2.11E+04	2.39E+04	2.88E+04
	Std	5.18E+02	1.50E+03	1.10E+03	4.35E+03	6.12E+03	6.35E+02
	p-value	**4.78E-02^{+}**	1.41E-09^{-}	1.42E-09^{-}	1.60E-09^{-}	1.89E-02^{-}	-

(*Continued*)

Table 2. (Continued)

Function	Quality	CSO	DLLSO	LIPS	HPSO_TVAC	ALC-PSO	LCSO
F16	Mean	5.00E+00	2.00E+00	1.00E+00	3.00E+00	3.00E+00	4.00E+00
	Std	1.97E-01	1.78E+00	6.56E-01	3.83E-01	3.08E-01	2.86E-01
	p-value	**1.41E-09^{+}**	3.07E-04^{-}	1.04E-08^{-}	4.54E-07^{-}	5.50E-03^{-}	-
F17	Mean	7.80E+02	1.98E+02	1.39E+03	8.40E+02	5.50E+02	4.87E+02
	Std	4.44E+01	1.17E+01	1.98E+02	1.07E+02	7.75E+01	6.35E+01
	p-value	**1.41E-09^{+}**	1.42E-09^{-}	**1.42E-09^{+}**	**1.80E-09^{+}**	**2.80E-03^{+}**	-
F18	Mean	9.10E+02	3.09E+02	1.51E+03	2.53E+03	1.20E+03	8.90E+02
	Std	1.94E+01	2.00E+02	2.11E+02	2.25E+02	1.87E+02	1.59E+01
	p-value	**2.50E-03^{+}**	1.60E-09^{-}	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.99E-07^{+}**	-
F19	Mean	1.10E+01	1.80E+01	5.70E+02	8.20E+01	4.90E+01	1.10E+01
	Std	1.49E+00	4.63E+00	6.71E+02	1.27E+01	1.11E+01	1.29E+00
	p-value	5.35E-01$^{=}$	**5.21E-09^{+}**	**1.41E-09^{+}**	**1.41E-09^{+}**	**1.41E-09^{+}**	-
F20	Mean	4.99E+01	5.00E+01	4.99E+01	4.99E+01	5.00E+01	4.95E+01
	Std	4.81E-01	0.00E+00	2.24E-01	0.00E+00	0.00E+00	2.66E-09
	p-value	**4.28E-02^{+}**	**1.04E-02^{+}**	3.97E-01$^{=}$	1.18E-01$^{=}$	**1.04E-02^{+}**	-
F21	Mean	3.70E+02	4.10E+02	4.40E+02	4.20E+02	3.30E+02	3.80E+02
	Std	4.49E+01	3.95E+01	4.69E+01	2.97E+01	5.48E+01	3.67E+01
	p-value	3.17E-01$^{=}$	**5.12E-07^{+}**	**4.39E-09^{+}**	**5.17E-08^{+}**	5.86E-01$^{=}$	-
F22	Mean	1.40E+03	5.48E+03	1.56E+04	5.05E+03	6.97E+03	2.15E+03
	Std	4.01E+02	1.12E+03	1.30E+03	6.52E+02	9.59E+02	5.34E+02
	p-value	8.86E-06^{-}	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F23	Mean	2.88E+04	1.10E+04	2.01E+04	2.57E+04	2.38E+04	2.15E+04
	Std	6.13E+02	1.37E+03	1.13E+03	4.04E+03	5.09E+03	1.05E+04
	p-value	1.04E-01$^{=}$	2.98E-02^{-}	2.98E-02^{-}	6.55E-01$^{=}$	7.86E-01$^{=}$	-
F24	Mean	3.00E+02	3.70E+02	5.80E+02	6.10E+02	5.50E+02	2.10E+02
	Std	2.17E+01	1.83E+01	2.15E+01	2.30E+01	2.26E+01	2.49E+01
	p-value	**2.57E-09^{+}**	**1.80E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F25	Mean	4.10E+02	4.60E+02	7.10E+02	6.00E+02	6.30E+02	3.70E+02
	Std	1.41E+01	1.58E+01	2.09E+01	1.95E+01	1.88E+01	1.47E+01
	p-value	**2.87E-08^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F26	Mean	3.90E+02	4.40E+02	5.40E+02	6.80E+02	6.20E+02	3.10E+02
	Std	1.81E+01	1.27E+01	1.33E+02	1.36E+01	2.24E+01	1.35E+01
	p-value	**1.60E-09^{+}**	**1.42E-09^{+}**	**5.91E-05^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F27	Mean	1.33E+03	1.99E+03	3.75E+03	4.45E+03	3.56E+03	8.80E+02
	Std	1.64E+02	1.12E+02	1.44E+02	1.99E+02	1.65E+02	1.69E+02
	p-value	**6.80E-07^{+}**	**2.87E-08^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	**1.42E-09^{+}**	-
F28	Mean	2.89E+03	3.38E+03	8.90E+03	1.76E+04	4.72E+03	2.77E+03
	Std	7.82E+02	9.91E+02	1.25E+03	1.28E+03	1.36E+03	1.07E+03
	p-value	**8.87E-05^{+}**	**4.67E-06^{+}**	**1.41E-09^{+}**	**1.41E-09^{+}**	**1.99E-07^{+}**	-
w/l/t		18/5/5	20/7/1	23/3/2	24/2/2	22/3/3	-

https://doi.org/10.1371/journal.pone.0272632.t002

Where h is the height of net, g is the acceleration of gravity. We set downward direction as the positive reference direction, the vibration intensity (a_1) at the collision point is given by:

$$a_1 = \frac{(m_1 v_1 + m_1 v_2) S_2}{t_c S_1 m_2} \tag{12}$$

Where m_1 is the weight of the table tennis ball, t_c and S_1 are the contact time and the maximum contact area between the ball and the table respectively, m_2 and S_2 are the weight and area of the table respectively. v_2 is the rebound velocity of the ball, defined as following:

$$v_2 = COR \times v_1 \tag{13}$$

Where COR is the coefficient of restitution [51].

To get the maximum detection distance (R_m), we have carried out the following equation derivation process.

The transfer process of mechanical vibration inside the ping pong table is analyzed by elastic dynamics method. The physical equation, geometric equation and balance equation in polar coordinates are established as Eqs (20–22).

$$physical\ equation: \begin{cases} \varepsilon_r = \dfrac{1}{E}[\sigma_r - \mu(\sigma_\theta + \sigma_z)] \\ \varepsilon_\theta = \dfrac{1}{E}[\sigma_\theta - \mu(\sigma_r + \sigma_z)] \\ \varepsilon_z = \dfrac{1}{E}[\sigma_z - \mu(\sigma_r + \sigma_\theta)] \\ \gamma_{zr} = \dfrac{2(1+\mu)}{E}\tau_{zr} \end{cases} \tag{14}$$

$$geometric\ equation: \begin{cases} \varepsilon_r = \dfrac{\partial u}{\partial r} \\ \varepsilon_\theta = \dfrac{u}{r} \\ \varepsilon_z = \dfrac{\partial w}{\partial z} \\ \gamma_{zr} = \dfrac{\partial u}{\partial z} + \dfrac{\partial w}{\partial r} \end{cases} \tag{15}$$

$$balance\ equation: \rho\frac{\partial^2 u}{\partial t^2} = \frac{\partial \sigma_r}{\partial r} + \frac{\partial \tau_{zr}}{\partial z} + \frac{\sigma_r - \sigma_\theta}{r} + R \tag{16}$$

Where u, w are the displacement components of one point in the ping pong table in the r-axis and z-axis directions. σ_θ, σ_r and σ_z are the normal stresses in the θ-axis, r-axis and z-axis directions respectively. τ_{zr} is a upward shear stress perpendicular to the zr-plane. ε_θ, ε_r and ε_z are the normal strains in the θ-axis, r-axis and z-axis directions respectively. γ_{zr} is a upward shear strain perpendicular to the zr-plane. R is the periodic stress along the r-axis applied to the outer layer of the table. M and K are frequency and amplitude of the vibration source. The constants E, ρ, μ represent the Young's modulus, density and Poisson's ratio of the table material, respectively.

To simplify the model, make $R = 1e8 \times \sin(20\pi t)$, z = 0. After substituting the geometric equations into the physical equations, we can obtain:

$$\begin{cases} (1+\mu)\sigma_r = \frac{\mu E}{1-2\mu}(\frac{\partial u}{\partial r} + \frac{u}{r} + \frac{\partial w}{\partial z}) + E\frac{\partial u}{\partial r} \\ (1+\mu)\sigma_\theta = \frac{\mu E}{1-2\mu}(\frac{\partial u}{\partial r} + \frac{u}{r} + \frac{\partial w}{\partial z}) + E\frac{u}{r} \\ (1+\mu)\sigma_z = \frac{\mu E}{1-2\mu}(\frac{\partial u}{\partial r} + \frac{u}{r} + \frac{\partial w}{\partial z}) + E\frac{\partial w}{\partial z} \\ \gamma_{zr} = \frac{\partial u}{\partial z} + \frac{\partial w}{\partial r} \end{cases} \tag{17}$$

As z approaches infinity, we can ignore the axial displacement w, shear stress τ_{zr}, and the axial time-domain equilibrium equation. Then the equation set-up (17) can be reduced to the equation set-up (18).

$$\begin{cases} (1+\mu)\sigma_r = \frac{\mu E}{1-2\mu}(\frac{\partial u}{\partial r} + \frac{u}{r}) + E\frac{\partial u}{\partial r} \\ (1+\mu)\sigma_\theta = \frac{\mu E}{1-2\mu}(\frac{\partial u}{\partial r} + \frac{u}{r}) + E\frac{u}{r} \\ (1+\mu)\sigma_z = \frac{\mu E}{1-2\mu}(\frac{\partial u}{\partial r} + \frac{u}{r}) \\ \gamma_{zr} = \frac{E}{2(1+\mu)}(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial r}) = 0 \end{cases} \tag{18}$$

Substitute equation set-up (18) into equation set-up (15) to get:

$$\begin{cases} \rho\frac{\partial^2 u}{\partial t^2} - K\sin(Mt) = A(\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} - \frac{u}{r^2}) \\ (A = \frac{(1-u)E}{(1-2u)(1+u)}, K = 10^8) \end{cases} \tag{19}$$

According to the separation of variables method, assume $u = y(r) \times \sin(Mt)$, then Eq (19) can be simplified to Eq (20), and its analytical Eq (21) can be obtained through MATLAB.

$$Ay'' + Ay' - A\frac{y}{r^2} + \rho M^2 y + K = 0 \tag{20}$$

$$y(r) = -\frac{K\pi}{2\rho M^2} struveH(1, Mr\sqrt{\frac{\rho(1-2\mu)(1+\mu)}{E(1-\mu)}}) \tag{21}$$

Where $struveH(1, x)$ is an intrinsic function of software Mathematica. And $struveH(n, x)$ has branch-cut discontinuity in the range of $-\infty$~0 on the complex plane. For some specific variable values, *struveH* can automatically calculate the exact value and obtain the value of any numerical precision.

Finally, R_m will be obtained by solving the equation set-up (22) [52].

$$\begin{cases} y(r) = -\dfrac{K\pi}{2\rho M^2} struveH(1, Mr\sqrt{\dfrac{\rho(1-2\mu)(1+\mu)}{E(1-\mu)}}) \\ \dfrac{1}{2}a_1\left(\dfrac{t_c}{2}\right)^2 = x(0) \\ \dfrac{1}{2}a_2\left(\dfrac{t_c}{2}\right)^2 = x(R_m) \end{cases} \tag{22}$$

Where *StruveH* is a function of software Mathematica. E, ρ, and μ represent the table's Young's modulus, density, and Poisson's ratio, respectively. M and K denote the frequency and the amplitude of the external stress, respectively. r is the propagation distance of the vibration. a_2 is the resolution of the sensor.

Based on the designed model, the sensors distribution optimization has been converted to search the minimum value of the function $minsensor = f(\alpha, \beta, x, y)$ via LCSO. In this work, each particle in the LCSO represents a candidate solution and the return value of the function $minsensor = f(\alpha, \beta, x, y)$ represents particle's fitness. The position of the particle is expressed in discrete form and divided into 4 parts, representing α, β, x, and y. During the swarm iteration, the particles that lose the competition learn from the winner and the higher-level particles and update their position and velocity. The winner and the highest-level particles directly enter the next generation. The parameters involved in the experiment are listed in Table 3. Fig 5A–5C show the sensor distribution in different iteration swarms. After LCSO optimization and 1000 iterations, the number of the sensors reduces to 51 (Fig 5D). Overall, the relationship between number of sensors and time of iteration is shown in Fig 5E. It can be observed that the number of sensors decreases gradually as the number of iterations increases. These test results illustrate that the minimum number of sensors is successfully obtained by LCSO, and the optimal solution (α = 60.012°, β = 90.0880°, x = 1371, y = 197) meets the expectation of the largest triangle area.

The reliability of the designed intelligent umpiring system is crucial for its application. To verify this performance, we randomly generate some points on the table tennis table by Matlab

Table 3. Parameters applied in sensors distribution by LCSO.

Symbol	Quantity	Value
NP	Swarm size	100
D	Particle dimension	40
R_m	Maximum detection distance	39.25 cm
c_1	Acceleration coefficients	1
c_2	Acceleration coefficients	0.5
h	Net hight	15.25 cm
m_1	Ball weight	2.53 g
m_2	Table weight	118 kg
S_1	Maximum contact area	50.89 mm^2
S_2	Table surface area	4.18 m^2
t_c	Contact time	0.4192 ms
a_2	Sensor resolution	0.1 m/s^2
E	Young's modulus	2.2 GPa
K	External stress amplitude	20.3 N
ρ	Density	880 Kg/m^3
μ	Poisson's ratio	0.24

https://doi.org/10.1371/journal.pone.0272632.t003

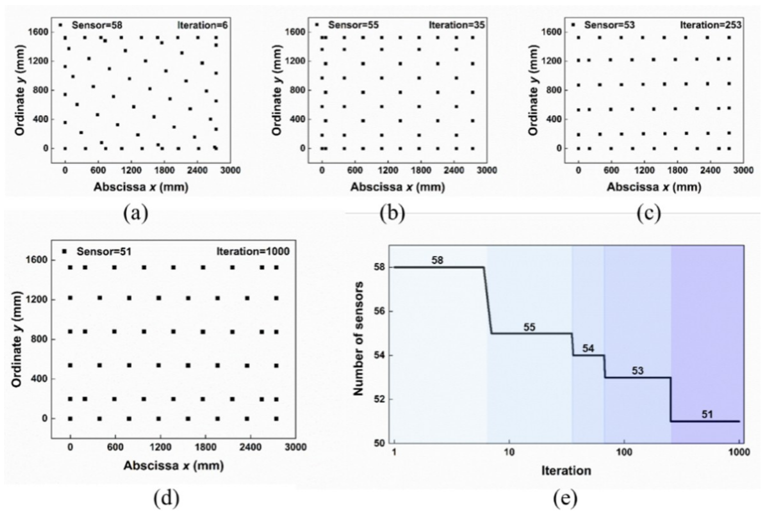


Fig 5. Best sensor distribution in different iteration ((a) 6, (b) 35 and (c) 253) swarms. (d) The optimal sensor distribution after LCSO optimization. (e) The number of sensors with different iterations.

https://doi.org/10.1371/journal.pone.0272632.g005

and calculate the coordinates of all points by the designed umpiring system. Based on the final sensor distribution (Fig 5D), the random points and their three nearest sensors are shown in Fig 6A. The coordinate (x_n, y_n) of each point can be obtained by solving equation set-up (23).

$$\begin{cases} (x_{an} - x_n)^2 + (y_{an} - y_n)^2 = r_{an}{}^2 \\ (x_{bn} - x_n)^2 + (y_{bn} - y_n)^2 = r_{bn}{}^2 \\ (x_{cn} - x_n)^2 + (y_{cn} - y_n)^2 = r_{cn}{}^2 \end{cases} \quad (n = 1, 2, \cdots, 15) \tag{23}$$

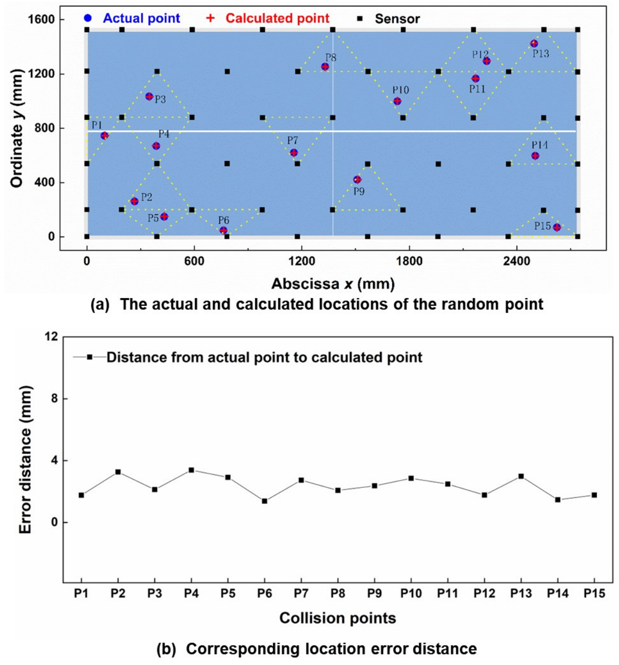


Fig 6. (a) Comparison of the calculated location and actual location of the random point and (b) corresponding location error.

https://doi.org/10.1371/journal.pone.0272632.g006

Table 4. Parameters applied in equation set-up (23).

Point 1	ra1	229	Nearest three sensors	Sensor A1 (0,539)
	rb1	165		Sensor B1 (0,879)
	rc1	163		Sensor C1 (195,978)
Point 2	ra2	95	Nearest three sensors	Sensor A2 (194,199)
	rb2	288		Sensor B2 (390,0)
	rc2	325		Sensor C2 (586,198)
Point 3	ra3	870	Nearest three sensors	Sensor A3 (586,198)
	rb3	850		Sensor B3 (194,199)
	rc3	498		Sensor C3 (391,538)
Point 4	ra4	131	Nearest three sensors	Sensor A4 (391,538)
	rb4	284		Sensor B4 (195,878)
	rc4	288		Sensor C4 (587,878)
Point 5	ra5	767	Nearest three sensors	Sensor A5 (195,878)
	rb5	746		Sensor B5 (587,878)
	rc5	1070		Sensor C5 (392,1218)
Point 6	ra6	51	Nearest three sensors	Sensor A6 (782,0)
	rb6	232		Sensor B6 (586,198)
	rc6	262		Sensor C6 (979,197)
Point 7	ra7	48	Nearest three sensors	Sensor A7 (1176,537)
	rb7	344		Sensor B7 (980,877)
	rc7	366		Sensor C7 (1372,877)
Point 8	ra8	157	Nearest three sensors	Sensor A8 (1177,1217)
	rb8	242		Sensor B8 (1569,1216)
	rc8	512		Sensor C8 (1766,1525)
Point 9	ra9	258	Nearest three sensors	Sensor A9 (1371,197)
	rb9	348		Sensor B9 (1764,196)
	rc9	133		Sensor C9 (1568,536)
Point 10	ra10	127	Nearest three sensors	Sensor A10 (1765,876)
	rb10	272		Sensor B10 (1569,1216)
	rc10	315		Sensor C10 (1962,1216)
Point 11	ra11	292	Nearest three sensors	Sensor A11 (2157,875)
	rb11	214		Sensor B11 (1962,1216)
	rc11	189		Sensor C11 (2354,1215)
Point 12	ra12	393	Nearest three sensors	Sensor A12 (2551,1525)
	rb12	281		Sensor B12 (1962,1216)
	rc12	145		Sensor C12 (2354,1215)
Point 13	ra13	245	Nearest three sensors	Sensor A13 (2354,1215)
	rb13	332		Sensor B13 (2740,1214)
	rc13	277		Sensor C13 (2740,1525)
Point 14	ra14	162	Nearest three sensors	Sensor A14 (2353,535)
	rb14	245		Sensor B14 (2740,535)
	rc14	280		Sensor C14 (2550,875)
Point 15	ra15	135	Nearest three sensors	Sensor A15 (2740,0)
	rb15	280		Sensor B15 (2352,0)
	rc15	145		Sensor C15 (2549,195)

https://doi.org/10.1371/journal.pone.0272632.t004

Where n is the serial number of random point, (x_{an}, y_{an}), (x_{bn}, y_{bn}), and (x_{cn}, y_{cn}) are the coordinates of the three nearest sensors respectively, r_{an}, r_{bn}, and r_{cn} are distances between the point P_n to each of its three nearest sensors. The parameters involved in equation set-up (23) are listed in Table 4. The corresponding experimental results are shown in Fig 6A and 6B. All points on the table are accurately located and the location errors are less than 3.39 mm. This proves that our umpiring system has accurate positioning and high stability. In addition, to deeply understand the actual collision output performance of the sensor, we also measured the output voltage of the self-powered acceleration sensor when the sensor sensed actual collision point made by table tennis. In our previous study, we have tested the sensitivity of the sensor and the sensitivity of the sensor is up to 20.4 V/(m/s^2) when the range of acceleration is 1–11 m/s^2 [25]. The relationship between acceleration and collision distance is shown in Fig 7. The measured acceleration increases with the increasing collision distance. This proves that our sensor has a superior linear output.

Conclusion

In summary, we presented a real-time intelligent umpiring system based on arrayed self-powered acceleration sensor nodes to assist referee to make precise umpiring. To avoid the periodic charging and reduce the number of sensors, high sensitivity self-powered acceleration sensors were installed on the back surface of the table tennis table after arrayed sensor nodes conducted a novel particle swarm optimization (LCSO). A model was established to optimize sensor nodes distribution. To minimize the function $minsensor = f(\alpha, \beta, x, y)$ based on the model, we presented an improved particle swarm optimization—LCSO. The number of sensors reduced from 58 to 51 via the LCSO algorithm. The simulation results on Matlab showed that the designed intelligent umpiring system had a high accuracy (errors below 3.5 mm). This work proposed a novel sensor positioning model and an effective method to optimize sensor nodes, which would make the umpiring of the table tennis competition precise and real-time.

Although the proposed algorithm showed competitive performance on both numerical comparison and application, the experimental results showed that the proposed algorithm still left room for further improvement. This can be explained by "No free lunch" theory. Thus, in our future work, we will continue to study the proposed algorithm with respect to specific problems. Targeting different kinds of problem, we will design adaptive parameters to improve the robustness of LCSO.

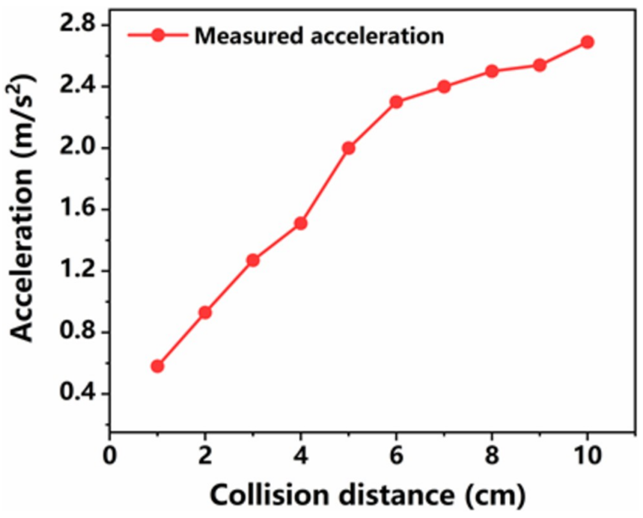


Fig 7. The measured acceleration by freeing the Ping-Pong ball with different collision distance.

https://doi.org/10.1371/journal.pone.0272632.g007

Author Contributions

Conceptualization: Chaoran Liu.

Data curation: Ke Lu, Ruizhi Sha.

Formal analysis: Dujuan Li.

Funding acquisition: Gaofeng Wang, Linxi Dong.

Investigation: Ke Lu, Kai Fan.

Methodology: Chaoran Liu.

Software: Yishao Wang, Dongyang Li.

Supervision: Weihuang Yang.

Validation: Ke Lu, Chaoran Liu, Yishao Wang.

Writing – original draft: Ke Lu.

Writing – review & editing: Chaoran Liu, Haiyang Zou, Linxi Dong, Ruizhi Sha.

References

1. Wong PKC. Developing an intelligent table tennis umpiring system: identifying the ball from the scene Asia. 2008 Second Asia International Conference on Modelling & Simulation (AMS). 2008; https://doi.org/10.1109/AMS.2008.110
2. Wong P, Dooley L. Tracking Table Tennis Balls in Real Match Scenes for Umpiring Applications. British Journal of Mathematics & Computer Science. 2011; 1(4): 228–241. https://doi.org/10.5281/zenodo.8730
3. Zhao WG. The analysis and thoughts of "Hawk Eye" technology in table tennis matches. Sichuan Sports Science. 2015. https://doi.org/10.13932/j.cnki.sctykx.2015.02.21
4. Tang JJ, Liu FD, Cai XL, et al. Research on the influences of rule changes of table tennis on the appreciation of the competition. J. Bjing Sport University. 2007.
5. Zhang Q, Tang D. Feasibility analysis on introduction of "Hawk-Eye" system for high-level table tennis tournament. Sichuan Sports Science. 2016.
6. Akyildiz IF, Su WL, et al. A survey on sensor networks. IEEE Communications Magazine. 2002; 40(8): 102–114. https://doi.org/10.1109/MCOM.2002.1024422
7. Diamond SM, Ceruti MG. Application of Wireless Sensor Network to Military Information Integration. 2007 5th IEEE International Conference on Industrial Informatics. 2007; https://doi.org/10.1109/INDIN.2007.4384776
8. Hamdi M, Boudriga N, Obaidat MS. WHOMoVeS: An optimized broadband sensor network for military vehicle tracking. International Journal of Communication Systems. 2008; 21(3): 277–300. https://doi.org/10.1002/dac.890
9. Elhabyan R, Shi W, et al. Coverage protocols for wireless sensor networks: Review and future directions. Journal of Communications and Networks. 2019; 21(1): 45–60. https://doi.org/10.1109/JCN.2019.000005
10. Antunes P, Lima H, et al. Optical fiber sensors for static and dynamic health monitoring of civil engineering infrastructures: Abode wall case study. Measurement. 2012; 45(7): 1695–1705. https://doi.org/10.1016/j.measurement.2012.04.018
11. Caetano E, Silva S, Bateira J. A vision system for vibration monitoring of civil engineering structures. Experimental Techniques. 2011; 35(4): 74–82. https://doi.org/10.1111/j.1747-1567.2010.00653.x
12. Kulkarni RV, Venayagamoorthy GK. Particle swarm optimization in wireless-sensor networks: A brief survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews). 2011; 41(2): 262–267. https://doi.org/10.1109/TSMCC.2010.2054080
13. Kukunuru N, Davuluri RL, et al. Sensor deployment using particle swarm optimization. International Journal of Engineering Science and Technology. 2010; 2(10).
14. Wang X, Wang S, Ma JJ. An improved co-evolutionary particle swarm optimization for wireless sensor networks with dynamic deployment. Sensors. 2007; 7: 354–370. https://doi.org/10.3390/s7030354

15. Kuila P, Jana PK. Energy efficient clustering and routing algorithms for wireless sensor networks: Particle swarm optimization approach. Engineering Applications of Artificial Intelligence. 2014; 33: 127–140. https://doi.org/10.1016/j.engappai.2014.04.009

16. Singh B, Lobiyal DK. A novel energy-aware cluster head selection based on particle swarm optimization for wireless sensor networks. Human-centric Computing and Information Sciences. 2012.

17. Aziz NA, Mohemmed AW, et al. A wireless sensor network coverage optimization algorithm based on particle swarm optimization and Voronoi diagram. 2009 International Conference on Networking, Sensing and Control. 2009. https://doi.org/10.1109/ICNSC.2009.4919346

18. Ni QJ, Du H, et al. An improved dynamic deployment method for wireless sensor network based on multi-swarm particle swarm optimization. Natural Computing. 2017; 16: 5–13. https://doi.org/10.1007/s11047-015-9519-0

19. Li P. Node localization method for massive sensor networks based on clustering particle swarm optimization in cloud computing environment. International Journal of Wavelets, Multiresolution and Information Processing. 2019; 18(1), 1941016. https://doi.org/10.1142/S0219691319410169

20. Eberhart R, Kennedy J. A new optimizer using particle swarm theory. MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science. 1995. https://doi.org/10.1109/MHS.1995.494215

21. Li YH, Zhan ZH, et al. Competitive and cooperative particle swarm optimization with information sharing mechanism for global optimization problems. Information Sciences. 2015; 293: 370–382. https://doi.org/10.1016/j.ins.2014.09.030

22. Xiong T, Bao YK, et al. Forecasting interval time series using a fully complex-valued RBF neural network with DPSO and PSO algorithms. Information Sciences. 2015; 305: 77–92. https://doi.org/10.1016/j.ins.2015.01.029

23. Cai Q, Gong MQ, et al. Discrete particle swarm optimization for identifying community structures in signed social networks. Neural Networks. 2014; 58: 4–13. https://doi.org/10.1016/j.neunet.2014.04.006 PMID: 24856248

24. El-Abd Mohammed. Preventing premature convergence in a PSO and EDA hybrid. 2009 IEEE Congress on Evolutionary Computation. 2009. https://doi.org/10.1109/CEC.2009.4983330

25. Liu CR, Wang YS, et al. A self-powered and high sensitivity acceleration sensor with V-Q-a model based on triboelectric nanogenerators (TENGs). Nano Energy. 2020; 67, 104228. https://doi.org/10.1016/j.nanoen.2019.104228

26. Liu CR, Fang LX, et al. Theoretical investigation and experimental verification of the self-powered acceleration sensor based on triboelectric nanogenerators (TENGs). Extreme Mechanics Letters. 2021; 42, 101021. https://doi.org/10.1016/j.eml.2020.101021

27. Miao QQ, Liu CR, et al. Toward Self-Powered Inertial Sensors Enabled by Triboelectric Effect. ACS Applied Electronic Materials. 2020; 2(10): 3072–3087. https://doi.org/10.1021/acsaelm.0c00644

28. Gong MQ, Yan JN, et al. Influence maximization in social networks based on discrete particle swarm optimization. Information Sciences. 2016; 367: 600–614. https://doi.org/10.1016/j.ins.2016.07.012

29. Shi Y, Eberhart R, et al. A modified particle swarm optimizer. 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360). 1998. https://doi.org/10.1109/ICEC.1998.699146

30. Parsopoulos KE, Vrahatis MN. On the computation of all global minimizers through particle swarm optimization. IEEE Transactions on Evolutionary Computation. 2004; 8(3): 211–224. https://doi.org/10.1109/TEVC.2004.826076

31. Liang JJ, Qin AK, et al. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Transactions on Evolutionary Computation. 2006; 10(3): 281–295. https://doi.org/10.1109/TEVC.2005.857610

32. Oca MAMD, Stutzle T, et al. Incremental social learning in particle swarms. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2011; 41(2): 368–384. https://doi.org/10.1109/TSMCB.2010.2055848 PMID: 20875976

33. Zhan ZH, Zhang J, et al. Orthogonal learning particle swarm optimization. IEEE Transactions on Evolutionary Computation. 2011; 15(6): 832–847. https://doi.org/10.1109/TEVC.2010.2052054

34. Ren ZG, Zhang AM, et al. A scatter learning particle swarm optimization algorithm for multimodal problems. IEEE Transactions on Cybernetics. 2014; 44(7): 1127–1140. https://doi.org/10.1109/TCYB.2013.2279802 PMID: 24108491

35. Li J, Zhang JQ, et al. Composite particle swarm optimizer with historical memory for function optimization, IEEE Transactions on Cybernetics. 2015; 45(10): 2350–2363. https://doi.org/10.1109/TCYB.2015.2424836 PMID: 26390177

36. Qin QD, Shi C, et al. Particle swarm optimization with interswarm interactive learning strategy. IEEE Transactions on Cybernetics. 2016; 46(10): 2238–2251. https://doi.org/10.1109/TCYB.2015.2474153 PMID: 26357418

37. Chen WN, Zhang J, et al. Particle swarm optimization with an aging leader and challengers. IEEE Transactions on Evolutionary Computation. 2013; 17(2): 241–258. https://doi.org/10.1109/TEVC.2011.2173577

38. Li DY, Guo WA, et al. Niching particle swarm optimizer with entropy-based exploration strategy for global optimization. Advances in Swarm Intelligence. 2019; 11655: 118–127. https://doi.org/10.1007/978-3-030-26369-0_11

39. Li DY, Guo WA, et al. An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization. Swarm and Evolutionary Computation. 2021; 60, 100789. https://doi.org/10.1016/j.swevo.2020.100789

40. Li XD, Yao X. Cooperatively coevolving particle swarms for large scale optimization. IEEE Transactions on Evolutionary Computation. 2012; 16(2): 210–224. https://doi.org/10.1109/TEVC.2011.2112662

41. Omidvar MN, Li XD, et al. Cooperative co-evolution with differential grouping for large scale optimization. IEEE Transactions on Evolutionary Computation. 2014; 18(3): 378–393. https://doi.org/10.1109/TEVC.2013.2281543

42. Omidvar MN, Yang M, et al. DG2: A faster and more accurate differential grouping for large-scale black-box optimization. IEEE Transactions on Evolutionary Computation. 2017; 21(6): 929–942. https://doi.org/10.1109/TEVC.2017.2694221

43. Chen WX, Weise T, et al. Large-scale global optimization using cooperative coevolution with variable interaction learning. International Conference on Parallel Problem Solving from Nature. Springer, Berlin, Heidelberg. 2010: 300–309. https://doi.org/10.1007/978-3-642-15871-1_31

44. Yang Q, Chen WN, et al. A level-based learning swarm optimizer for large-scale optimization. IEEE Transactions on Evolutionary Computation. 2018; 22(4): 578–594. https://doi.org/10.1109/TEVC.2017.2743016

45. Ratnaweera A, Halgamuge SK, et al. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Transactions on Evolutionary Computation. 2004; 8(3): 240–255. https://doi.org/10.1109/TEVC.2004.826071

46. Cheng R, Jin YC. A competitive swarm optimizer for large scale optimization. IEEE Transactions on Cybernetics. 2015; 45(2): 191–204. https://doi.org/10.1109/TCYB.2014.2322602 PMID: 24860047

47. Cheng R, Jin YC. A social learning particle swarm optimization algorithm for scalable optimization. Information Sciences. 2015; 291: 43–60. https://doi.org/10.1016/j.ins.2014.08.039

48. Krohling RA, Mendel E. Bare bones particle swarm optimization with gaussian or cauchy jumps. 2009 IEEE Congress on Evolutionary Computation. 2009. https://doi.org/10.1109/CEC.2009.4983361

49. Qu BY, Suganthan PN, et al. A distance-based locally informed particle swarm model for multimodal optimization. IEEE Transactions on Evolutionary Computation. 2013; 17(3): 387–402. https://doi.org/10.1109/TEVC.2012.2203138

50. Liang JJ, Qu BY, Suganthan PN, et al. Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report. 2013; 201212(34): 281–295.

51. Zhang XW, Tao Z, et al. Dynamic behaviors of visco-elastic thin-walled spherical shells impact onto a rigid plate. Latin American Journal of Solids and Structures. 2014; 11(14): 2607–2623. https://doi.org/10.1590/S1679-78252014001400009

52. Liu CR, Yue JZ, et al. Vibration attenuation analysis of compressional gas cushion press nanoimprint lithography system. Journal of Mechanical Engineering Science. 2014; 228 (9): 1634–1642.