Optimal Route Driving for Leader-Follower Using Dynamic Particle Swarm Optimization

B. Tutuko, S. Nurmaini,* P. Sahayu
Intelligent System Research Group, Universitas Sriwijaya, Indonesia
beng_tutuko@yahoo.com, sitinurmaini@gmail.com, sahayuputri@gmail.com

***Abstract*—The mobile robots rely on trajectory generation problem when they are navigating in several environments, for achieving the best path. One of the solution by using a heuristic method, named Particle Swarm Optimization (PSO). In the previous study, by using such method, the mobile robot can find the best route towards the target without collision, moreover, its simplicity in algorithms, implement easily and has few parameters to regulate. However, the PSO original algorithm can't guarantee to produce an optimal solution. Local optimum still occurs especially in complex and dynamic environments, due to premature convergence. It causes the mobile robot collisions with obstacles and generates the long path to the target. In this paper, dynamic PSO is developed by using dynamic inertia function in setting parameter to accelerate convergence and re-initialization of particles performed to overcome the premature convergence. The comparison with three algorithms, such as OPSO, GPSO, and DPSO have analyzed in this paper. The proposed DPSO algorithm produce the optimum solution faster with the convergence of fewer than 150 iterations in static obstacles and 200 iterations on the moving obstacle, 4% shorter traveled lengths, 13% more smooth, with fast processing and it guaranteed to avoid collisions and stable movement to achieve the target.**

Keywords— Route Optimization, Non-holonomic, Leader-Follower, Particle Swarm Optimization

I. INTRODUCTION

The distributed robot's coordination and control in a group have attracted many researchers over the past few years. One of many research topics is the problem of coordination between robots in controlling the formation of some robots on the some applications, such as unmanned ground robot, unmanned aerial robot, unmanned underwater robot, flying, and satellite [1][2][3][4][5][6]. Various strategies have been proposed with a variety of approaches to control the formation of a group of robots, including behavior-based, virtual structure and leader-follower [5][6][7][8].

In some environment like a factory, the leader-follower approach becomes an important research, due to they must be communication during the process to accomplish the tasks. In the leader-follower application, one or more robots are appointed as the leaders, and the others are the followers to achieve the target. The leader is a reference to follower robots, who need to position themselves and maintain the desired relative position with respect to the leader [8][9]. In such approach, to determine formation maneuvers, it is only necessary to determine the leader's path and, the desired relative position and orientation between the leaders and the followers. When the direction of the leader's movement is known, the desired position (distance and angle) of the followers relative to the leader can be achieved by using the local control of each follower. However, if a leader's robot fails it can lead to a failure of the entire controlling process.

Hence, controlling the leader-follower robot in terms of position and orientation for achieving the targets, in rapid convergence time and high accuracy in dynamic environments is desirable. In such condition, the optimization route must be implemented on robotic control in a simple algorithm. A less computational resources very important requirement in leader-follower approach, due to swarm characteristic. Several approaches have been proposed with good performance results [8][10][11]. They implement leader-follower robot, with global information for sharing with each other. However, the computational cost is large, due to the complexities of the methods. When the algorithm is implemented in a simple robot with onboard sensor and processor becoming a major problem.

Particle swarm optimization (PSO) algorithm one of the most efficient optimization strategies for continuous nonlinear optimization problems based-on global information about the environment. It can be designed with simple algorithms and produce smooth and efficient trajectory [12][13][14]. Unfortunately, the original PSO algorithm is difficult to balance between exploration and exploitation capability. To overcome the limitation, several authors proposed different methods to achieve better accuracy and convergence [15][16][17]. Only a few researchers propose a method in a multi-robot control system, especially on leader-follower configuration based on a kinematic model. Hence, this research becomes important to be done in the development of motion control and optimization route on leader-follower robot based on a non-holonomic kinematic model in the Cartesian representation.

The structure of this paper as follows. In section 2, the process of declining kinematic models using Cartesian coordinates for the control of the formation of two non-holonomic mobile robots is described. In section 3, the design of route optimization with PSO method is explained. Some simulation results are included in section 4 to verify the feasibility of the model and the controller. While the conclusions and future work will be described in section 5.

II. LEADER-FOLLOWER KINEMATIC SYSTEM

In this section, the Cartesian coordinates for leader-based formation controls explain the kinematics model of the three-wheeled robot team. In such model, left wheels and the right wheel are controlled and one freewheel for balancing. The

values of robot movement parameters are obtained, $X-Y$ is world coordinate, and $x-y$ is a fixed Cartesian Coordinate of the leader robot. The parameters (X_L, Y_L) and (X_F, Y_F) are the global position of leaders and followers, where the subscript '*L*' represents the leader and the subscript '*F*' represents the follower. v_L is linear velocity of the leader and v_F is linear velocity of the follower. θ_L the angular orientation of the leader and θ_F is the angular orientation of the follower [18].

We can assume that the leader and follower robot follow the kinematics model of a unicycle robot in the inertial frame (see Fig. 1). The kinematics of each robot can be expressed as follow,

$$\begin{pmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{pmatrix} = T_{NH}(q)\, u(t) \tag{1}$$

where $q(t)$ is the general variable of an initial position of the robot $q(t) = [x(t), y(t), \theta(t)]^{\mathrm{T}}$, T_{NH} is non-holonomic transformation matrix, and $u(t)$ is forward kinematic matrix which is used to estimate position and speed.

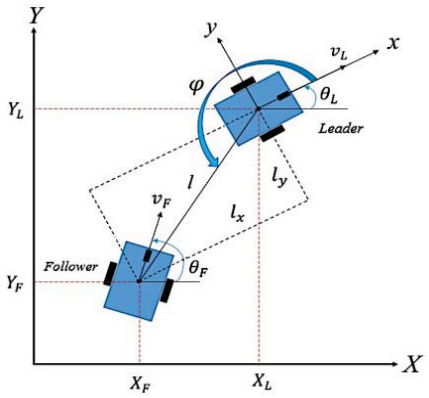


Fig. 1. Leader-Follower Kinematic System

The non-holonomic transformation of the mobile robot can be seen through the change of the three initial robotic position variables $q(t)$. By solving (1) to the change in the velocity of the right wheel and the left wheel, the single robot kinematic can be transformed into (2),

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} cos\theta(t) & 0 \\ sin\theta(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} \tag{2}$$

The modeling of the leader-follower system has been derived directly by the kinematic analysis of relative robot follower along the *x* and *y* coordinates associated with the robot leader. The leader *L* has configuration vector $[x_L, y_L, \theta_L]^T$ while the follower *F* has a vector $[x_F, y_F, \theta_F]^T$. The control inputs of the leader and the follower are the linear and angular velocities $[v_L, \omega_L]^T$ and $[v_F, \omega_F]^T$, respectively. The relative distance between leader and follower must be determined, thus they can be a move in the same trajectory. To illustrate the relative position between the robots, it's projected the relative distance along the *x* and *y* directions. In *x*-*y* Cartesian coordinates, the distance between the robot leader and the follower robot is l. By using the properties of trigonometric functions ie, $a.b=|a|\,.\,|b|\,cos\theta$, the rotation matrix for robot follower is obtained shown in (3) as follows:

$$\begin{bmatrix} x_F \\ y_F \\ \theta_F \end{bmatrix} = \begin{bmatrix} cos\theta_F & sin\theta_F & 0 \\ -sin\theta_F & cos\theta_F & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Based on Fig. 1, and assuming the relative distance can be derived using the matrix rotation in (3) the relative robot leader's distance to the follower robot is defined in (4). Where the relative position the follower robot along the *x*-direction is l_x and along the *y*-direction is l_y with relative orientation $\theta(t)$.

$$\begin{bmatrix} l_x \\ l_y \\ e_\theta \end{bmatrix} = \begin{bmatrix} -cos\theta_L & -sin\theta_L & 0 \\ sin\theta_L & -cos\theta_L & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} X_L - X_F \\ Y_L - Y_F \\ \theta_L - \theta_F \end{bmatrix} \tag{4}$$

If the position of the leader robot (X_L, Y_L) is determined and (l_x, l_y) are known and fixed to achieve and maintain the desired formation, a parameter (l_x, l_y) must be controlled, then the position with respect to the robot leader can be determined. By controlling $l_x \rightarrow l_x^d$ where l_x^d is the desired relative position along the *x* direction and $l_y \rightarrow l_y^d$, where l_y^d is the desired relative position along the *y*-direction. In the normal conditions, the relative distance between the leader robot and the follower robot is l^d, it needs to be simultaneously projected and to control the movement of the follower robot against the leader robot by using (5) to (8) as follows,

$$l_x^d = l^d \cos\varphi^d \text{ o}\rho, \tag{5}$$

$$l_x^d = l^d \cos\varphi^d - l^d\, \dot{\varphi}^d \sin\varphi^d \tag{6}$$

$$l_y^d = l^d \sin\varphi^d \text{ o}\rho, \tag{7}$$

$$l_y^d = l^d \sin\varphi^d - l^d\, \dot{\varphi}^d \cos\varphi^d \tag{8}$$

In general, the leader-follower kinematic model can be generated as follow:

$$\begin{bmatrix} l_x \\ l_y \\ \dot{e}_\theta \end{bmatrix} = \begin{bmatrix} cose_\theta & 0 & -1 & l_y \\ sine_\theta & 0 & 0 & l_x \\ 0 & 1 & 0 & -1 \end{bmatrix} \tag{9}$$

$$u = [v_F, \omega_F, v_L, \omega_L]^T \quad (10)$$

where ω_F is the angular velocity of the follower robot, v_F is the linear velocity of the follower robot, ω_L is angular velocity the leader robot and v_L is the linear velocity the leader robot. By using the leader-follower approach, ω_L and v_L is a time function that varies from input control ω and v.

III. ROUTE OPTIMIZATION

Particle swarm optimization (PSO) work based on the behavior of a herd of insects, such as ants, termites, bees or birds [15][17]. The algorithm mimics the social behavior of such organism. Social behavior consists of individual actions and the influence of other individuals in a group. The word "particle" denotes the individual. Each individual or particle behaves interconnected by using its own intelligence and also influenced the behavior of its collective group [15]. Thus, if one particle finds the right or short way to the target, the rest of the other group will also be able to follow the path immediately even though their location is far away in the group. There are two kinds of PSO algorithm such as original PSO and improved PSO [13][14][15][16]. In the original PSO (OPSO) algorithm the inertia weight (w) is set 1, thus the convergence speed of particles is fast, the adjustments of cognition and social component make particles search around in one point. It can produce a local minimum condition and the whole swarm will be converged at this position. However, if the inertia weight value is selected about $0 < w < 1$ the whole swarm is hard to jump out of the local optimum. It produced fatal weakness from this characteristic because no global optimum (g_{best}) is achieved. Hence, the dynamic inertia weight is desirable to regulate.

In this paper PSO algorithm is used to optimize the leader-follower robot tends to the target without collision. Therefore, it works not only optimization process, also to control the leader movement. The dynamic inertia weight is needed, due to the leader-follower robot move in the unstructured and dynamic environment. The dynamic PSO (DPSO) is created by using (11) and (12),

$$v_{i,j}^{n+1} = w * v_{i,j}^{n} + c_1 * r_1 * \left(P_{i,j}^{n} - x_{i,j}^{n}\right) + c_2 * r_2 * \left(P_{g,j}^{n} - x_{i,j}^{n}\right) \quad (11)$$

$$x_{i,j}^{n+1} = x_{i,j}^{n} + v_{i,j}^{n+1}, \quad j = 1,2, \ldots \ldots, d \quad (12)$$

Vector $P_i = (p_{i1}, p_{i2}, \ldots \ldots, p_{i,d})$ is the best previous position of the ith particle that gives the best fitness value, named personal best position p_{best} . Vector $P_g = (p_{g1}, p_{g2}, \ldots \ldots, p_{g,d})$ is the best particle among all the particles in the population, named global best (g_{best}). The weight of inertia w is used to balance the global exploration and local exploitation. r_1 , and r_2 are a random number uniformly distributed between [0,1]. The velocity $v_{i,j}$ is restricted to the range $[-v_{max}, v_{max}]$ in order to prevent the particles from flying out of the solution space. The acceleration coefficient c_1 and c_2 for making the better balance of the search space between the local exploitation the global exploration.

In the leader-follower case, the PSO must ensure the leader achieved the target, and the follower robot can follow the leader but it must keep the formation without collision. In this paper, the dynamic inertia weight w and the learning factor c_1 and c_2 are improved in (13) and (14).

$$w = w_i - \left(\left(\frac{i_{max} - i}{i_{max}}\right) * (w_{max} - w_{min})\right) \quad (13)$$

The DPSO algorithm serves to control PSO capabilities in local searches efficiently and convergent to global optimum solutions. The inertia weight w is updated to obtain an adaptive w value for each iteration, therefore the value can be dynamic and capable of improving the expected optimization result. The greater the value of iteration, the w value will be smaller, and preferably, if iteration is still early, then the value w will tend to be larger. If the w value gets bigger, then the particle is more focused towards exploration, but as it gets smaller, it is more focused towards exploitation [15]. $c1$ and $c2$ are the acceleration coefficients or learning rate of a single current particle for a better balance between global exploration by all particles in neighboring topology and local exploitation in the best fitness achieved. In this paper $c1$ and $c2$ is used for finding the target and to avoid the obstacle, due to it must change the vector of velocity and vector position. However, its targets should be found in a short period of time. In this paper, proposes the dynamic linear adjustment strategy for learning factors. The expression is given in (14) as follows,

$$c_1 = \frac{\Delta c_1 t}{t_{max}} + c_{1i} \text{ and } c_2 = \frac{\Delta c_2 t}{t_{max}} + c_{2i} \quad (14)$$

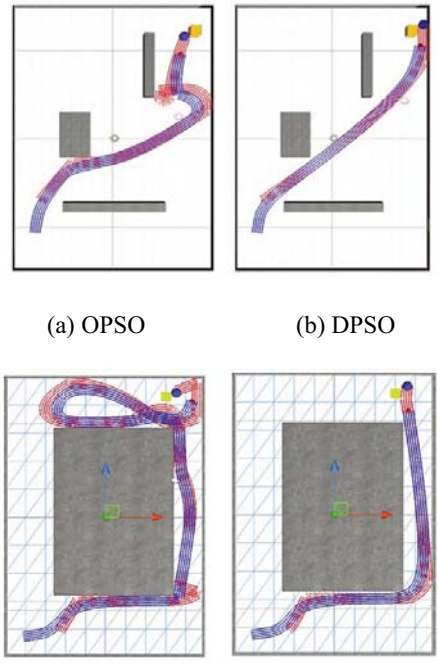
where, $\Delta c_1 = (c_f - c_i)$ and $\Delta c_2 = (c_f - c_i)$ c_f = final value, c_i = c initial value

If the number of iterations is increased, the cognitive ability of the individual is gradually reduced by improvement of the learning factors and improve the global search ability of particles. This strategy can improve particle's the global search ability in the whole search space in the early time and converge to the global optimum to the particles in the end.

IV. RESULT AND DISCUSSION

In this work, original PSO (OPSO), and Gaussian (GPSO) are compared to proposed dynamic PSO (DPSO) for the above-mentioned function minimization problems on leader-follower kinematic control. For such purpose, the number of swarm parameters about 50 particles and 1000 maximum iteration is taken. However, the number of particles is not very influential on the optimum solution generated PSO, but it affects the speed of the process. If the number of particles that are too small can get stuck on the local optimum even though the processing time is very fast. In contrast, large amounts of particles are rarely trapped in local optimum, but the process takes longer. Respective inertia weights and acceleration coefficient are selected for balancing between exploration and exploitation capability.

To verify the proposed algorithm, the new model of leader-follower based on a non-holonomic system for target seeking is simulated. Static and dynamic obstacles are utilized in the testing environment. In Fig. 2(a) and (b), the leader-follower robot moves to reach the target with a static obstacle. The leader robot is a blue line and the follower robot is a red line. Two types of PSO are used to view movement performance. The obstacle avoidance passing near and reaching the goal with a short trajectory is performed by the leader-follower robot, it avoids the obstacle coming from the right, deviates from the straight line. The leader-follower robot using the OPSO is able to reach the target, but the resulting trajectory is not smooth, long processing time, and generating large amounts of data to reach the target. Using the proposed DPSO approach, the leader-follower robot movement performs better, smoother movements, shorter travel times and produce less data generation.



(a) OPSO (b) DPSO

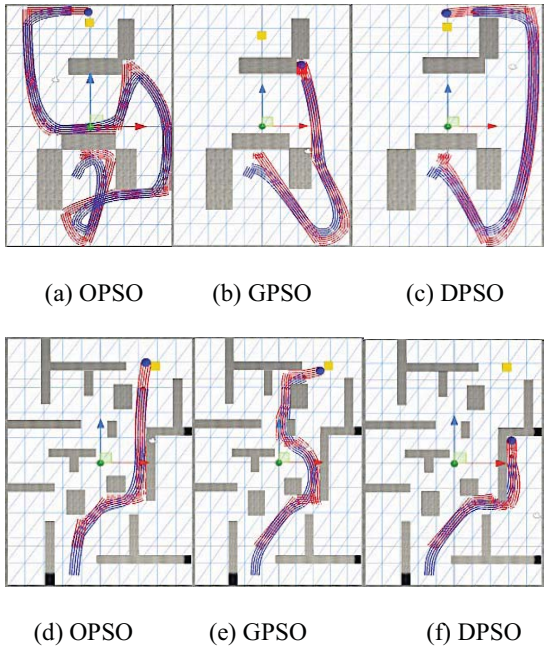
(c) OPSO (d) DPO

Fig. 2. Trajectory control in simple environment

The movement of the leader robot by using OPSO requires a fairly long route in reaching the target, therefore the execution time becomes longer about 29.14 sec. Conversely, if the robot leader using DPSO in optimizing the robot's route to be shorter, therefore the execution time becomes faster about 13.55. While the follower robot follows the movement of the robot leader in a relatively equal time with the time taken by the robot leader. Moreover, the movement of the leader robot is smooth, if DPSO is used and they can choose the simple way to find the target. In addition to test other environments a rectangular environment is created.

The robot must move from the initial position to target position it can be seen in Fig. 2 (c) and (d). The robot moves not smooth with long trajectory by using original PSO. The performance not satisfying due to the time processing to finish the route by using OPSO about 29.15 sec for the leader and about 29.18 sec for the follower and in some point of the trajectory the leader-follower robot crash the wall. Meanwhile by using the DPSO, the processing time about 13.50 sec for the leader and about 13.65 sec for the follower with short and smooth trajectory. Furthermore, the leader and the follower robot have the ability to maintain the position with the wall without collision.

From the robot's trajectory in Fig. 2 (c) and (d), it is seen that from the initial position the robot leader moves in search of the target and manages to find the target, the success of reaching the target is seen from the result of the route leading to a point. However, the route taken by the leader-follower robot using DPSO algorithm in achieving the target is more efficient than using the OPSO algorithm. It happens, due to DPSO algorithm, using parameter control of inertia function and coefficient acceleration to accelerate convergence and produce a global solution.



(a) OPSO (b) GPSO (c) DPSO

(d) OPSO (e) GPSO (f) DPSO

Fig. 3. Trajectory control in the cluttered environment

By using OPSO algorithm premature convergence always happens, the condition occurs when particles converge and particle velocity close to zero, but no global solution has been found. Based on the time taken by each robot using the OPSO algorithm from the starting point towards the target until the robot stops about 66 seconds while using the DPSO algorithm the time taken to reach the target faster only 31 seconds. DPSO algorithm using setting inertial parameters adaptable in accordance with the environmental dynamics that occur during the target search.

In the complex environment (see. Fig. 3 (a) – (f)) by using DPSO produce small processing time compare to OPSO and GPSO. Due to the proposed algorithm have the ability to change the position and orientation leader-follower robot in an adaptive manner. The leader-follower have the ability to achieve the target, but especially OPSO, the robots can't finish the task. They stop at one point, stack in the local minima (see Fig. 3 (a) and 3 (d)). However, by using GPSO and DPSO the task can be completed. The leader-follower use GPSO crash the wall (see. Fig. 3 (e) and 3 (b)), but still

move to the target. When they move based-on DPSO, the robot is able to reach the target without a collision, with a short route and fast processing time. Such condition also happens to the follower robot.

V. CONCLUSION

The social adaptation of knowledge for working and all individuals are considered the same generation based on the PSO algorithm. It has many advantages, such as the simple algorithm, good convergence performance, and the fewer control parameters. However, it does not provide a mechanism for escaping from local optimal solution and easy to fall into local extremum value. In case of leader-follower control, by using the original PSO, the leader always moving around in circles to find the target, and generate a lot of data to complete the task. Making the large search space for finding the possible solution space of the optimal solution by using inertia weight adjustment strategy into the original PSO. By using dynamic PSO, the leader-follower performs better, smoother movements, shorter travel times and produce less data generation by using the DPSO. The comparison results show that the proposed DPSO algorithm is more capable to obtain the global optimization solution and overcome the problem of local minima when the leader-follower move in the complex and dynamic environment.

ACKNOWLEDGMENT

Authors thank University of Sriwijaya and Ministry of Research Technology and Higher Education of Indonesia, for their support in our research work, with the financial of Penelitian Unggulan Perguruan Tinggi 2018 and Hibah Bersaing 2018.

REFERENCES

[1] G. Lee and D. Chwa, "Decentralized behavior-based formation control of multiple robots considering obstacle avoidance," *Intell. Serv. Robot.*, vol. 11, no. 1, pp. 127–138, 2018.

[2] L. He, P. Bai, X. Liang, J. Zhang, and W. Wang, "Feedback formation control of UAV swarm with multiple implicit leaders," *Aerosp. Sci. Technol.*, vol. 72, pp. 327–334, 2018.

[3] F. Berlinger, J. Dusek, M. Gauci, and R. Nagpal, "Robust Maneuverability of a Miniature, Low-Cost Underwater Robot Using Multiple Fin Actuation," *IEEE Robot. Autom. Lett.*, vol. 3, no. 1, pp. 140–147, 2018.

[4] M. A. Lewis and K.-H. Tan, "High precision formation control of mobile robots using virtual structures," *Auton. Robots*, vol. 4, no. 4, pp. 387–403, 1997.

[5] W. Ren and R. Beard, "Decentralized scheme for spacecraft formation flying via the virtual structure approach," *J. Guid. Control. Dyn.*, vol. 27, no. 1, pp. 73–82, 2004.

[6] Y. Abbasi, S. A. A. Moosavian, and A. B. Novinzadeh, "Formation control of aerial robots using virtual structure and new fuzzy-based self-tuning synchronization," *Trans. Inst. Meas. Control*, vol. 39, no. 12, pp. 1906–1919, 2017.

[7] T. Balch and M. Hybinette, "Behavior-based coordination of large-scale robot formations," in *MultiAgent Systems, 2000. Proceedings. Fourth International Conference on*, 2000, pp. 363–364.

[8] A. Loria, J. Dasdemir, and N. A. Jarquin, "Leader--follower formation and tracking control of mobile robots along straight paths," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 2, pp. 727–732, 2016.

[9] S. Nurmaini and B. Tutuko, "Intelligent Robotics Navigation System: Problems, Methods, and Algorithm," *Int. J. Electr. Comput. Eng.*, vol. 7, no. 6, pp. 3711–3726, 2017.

[10] H. Wang, D. Guo, X. Liang, W. Chen, G. Hu, and K. K. Leang, "Adaptive vision-based leader--follower formation control of mobile robots," *IEEE Trans. Ind. Electron.*, vol. 64, no. 4, pp. 2893–2902, 2017.

[11] A. N. Asl, M. B. Menhaj, and A. Sajedin, "Control of leader--follower formation and path planning of mobile robots using Asexual Reproduction Optimization (ARO)," *Appl. Soft Comput.*, vol. 14, pp. 563–576, 2014.

[12] C.-J. Lin, T.-H. S. Li, P.-H. Kuo, and Y.-H. Wang, "Integrated particle swarm optimization algorithm based obstacle avoidance control design for home service robot," *Comput. Electr. Eng.*, vol. 56, pp. 748–762, 2016.

[13] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Appl. Soft Comput.*, vol. 11, no. 4, pp. 3658–3670, 2011.

[14] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 240–255, 2004.

[15] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998, pp. 69–73.

[16] B. Tang, Z. Zhu, and J. Luo, "Hybridizing particle swarm optimization and differential evolution for the mobile robot global path planning," *Int. J. Adv. Robot. Syst.*, vol. 13, no. 3, p. 86, 2016.

[17] F. den Bergh and A. P. Engelbrecht, "A convergence proof for the particle swarm optimiser," *Fundam. Informaticae*, vol. 105, no. 4, pp. 341–374, 2010.

[18] A. K. Das, R. Fierro, V. Kumar, J. P. Ostrowski, J. Spletzer, and C. J. Taylor, "A vision-based formation control framework," *IEEE Trans. Robot. Autom.*, vol. 18, no. 5, pp. 813–825, 2002.