Simulation of Robot Movement in 2-Dimensional Space Using Fuzzy-Particle Swarm Optimization

Irianto¹**, dan Hauw-Sen, Tan²**

¹Faculty of Electrical and Electronic Engineering Technology, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia
²Industrial Engineering Department, Faculty of Engineering, Bina Nusantara University, Jakarta, Indonesia, 11480

*Email korespondensi: irianto@utem.edu.my, hauw.rimo@binus.ac.id
Dikirim: 28-02-2020, Diterima: 15-03-2020, Diterbitkan: 31-03-2020

Abstract
Now days, the use of a group of autonomous robots are grown increasingly, especially for an application dealing with hazardous material and or dangerous situation. In this case, autonomous robot movement where there is no interference from a human on the execution process is very important. The concern is how this group of autonomous robots could arrive as fast as possible to the target location to perform the tasks given. If it includes the movement of groups of autonomous robots then particle swarm optimization (PSO) is one of a simple yet powerful method available. Fuzzy logic as a logic system has been proven can be combined with various numbers of applications or methods to get a more optimal result. One of them is the combination of fuzzy logic with PSO method. This paper implemented the fuzzy-PSO optimization method to simulate a group of robots movement to the target location using scratch programming. The fuzzy-PSO optimization results, then compared to the results of classic PSO optimization. It is found that the robots with fuzzy-PSO optimization movement arrived at the location target in average more than 40% faster compared to the robots with classic PSO optimization movement.

Keywords: Particle Swarm Optimization; Fuzzy Logic; Robot Movement

1. Introduction

An evolutionary computation technique called particle swarm optimization (PSO) was first introduced by Eberhart and Kennedy [1] in 1995. PSO is inspired with social behavior of animal living in a group; such as bird flocking, fish schooling, a swarm of ants, etc [2]. PSO is implemented to find the optimum solution of optimization problems in some certain space search.
Implementation PSO in autonomous robot movement has been an interesting research topic. Annuar et al. [3] implemented PSO for simulating robot movement based on odor. The movement of PSO also can be implemented into others optimization problem. For examples, Irianto [4] used PSO for solving the Hamiltonian systems, while Yuhui Shi and Eberhart [5], Poli et al. [6] and Shehata et al. [7] described variations in the algorithm, current and ongoing research, applications and open problems in PSO. The simple calculation of PSO makes this method become one of the widely used method in optimization problem. It can quickly select optimal solutions with shorter training time than other stochastic learning methods.

Fuzzy logic forms a bridge between two areas of qualitative and quantitative modeling. Fuzzy logic was suggested by Zadeh [8, 9] as a method for mimicking the ability of human reasoning using a small number of rules and still producing a smooth output via a process of interpolation. Implementing fuzzy logic to PSO can create a more reliable method in terms of accuracy.

A group of autonomous robots is needed to handle dangerous material and or dangerous situation where the life of humans cannot be compromised. This group of autonomous robots should move and arrive at the location target as soon as possible to perform the tasks given. The idea of using a group of autonomous mobile robots movement for searching hazardous target or approaching dangerous location with little to no human intervention can be realized with PSO method. This concept is still restricted to simulation; since there are multiple real-world factors those limit the conventional programming.

Introducing fuzzy into the search area the finding is an efficient way for the PSO to move them towards real-world success. Particle Swarm Optimization (PSO) has shown to be an effective tool for potential applications to the Collective Robotic Search (CRS) problem, and implementing fuzzy will increase the effectiveness of PSO in CRS [10].

Mobile robots can move in a large variety of ways. There are robots that walk, crawl, jump, swim, fly, roll, etc. The simplest type of movement is based on three variables, its \((x, y)\) positions and its orientation. For this paper the simulation is based on 2 dimensional spaces hence it will involve with the position of the robot in \(x, y\) coordinate and the distance between robots.

Let \((x_1, y_1)\) and \((x_2, y_2)\) be the position of two robots in 2 dimensional Cartesian coordinate, then the distance \((d)\) is define by

\[
d = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}
\]  

2. Methodology

2.1. Particle Swarm Optimization (PSO) Algorithm

Particle Swarm Optimization (PSO) was developed based on the behavior of animals that lives in-group. For example, the way group of ants working is based on their movement towards their food source (i.e. sugar). If one of the ants close to the food source it will share information through whatever way of communication to all the other ants that there is a food source nearby, at the end all other ants will move towards to the food source. Once all the ants have gathered in one food source location we can consider that location is the optimal solution of our problems.

The example above can be translated into the following model. Let \(x_1, x_2, ..., x_n\) are the position of \(n\) number of agents spread randomly throughout search space, \(v_1, v_2, ..., v_n\) be the velocity of every agent, and let \(f(x)\) be the objective function in the search space that we need to find the optimum values. Using PSO method the movement of every agent will be formulated as:

\[
v_i^{(t+1)} = v_i^{(t)} + c_1R_1(x_i^{(t)} - pb_i^{(t)}) + c_2R_2(x_i^{(t)} - gb^{(t)})\quad (2)
\]

Where \(v_i^{(t)}\) is the velocity of agent- \(i\) at \(j\)-th iteration, \(x_i^{(t)}\) is the position of agent- \(i\) at \(j\)-th iteration, \(pb_i^{(t)}\) is the personal best location, and \(gb^{(t)}\) is the global best location.

Personal best location is the closest location of an agent to the optimum value, and global best location is the location of the agent that is closest to optimum value among all other agents. The tendency of movement for every agents are depend on the value of weight constant \(c\). If \(c_1 > c_2\), then the agent tends to move to its own personal best. Where if the opposite, \(c_1 < c_2\), means that the agent tends to move to the global best. The Illustration about the agent movement can be found in Figure 1. \(R\) is a random number to guarantee the randomness of the model. After calculation of the velocity the position of every agent will be updated with:
PSO Algorithm:

1. Generate \( n \) number of agents with random positions and velocities, 
2. Evaluate the objective function \( f(x) \) for every agent. 
3. Compare evaluation with agent’s previous best position \( pb^{(i)} \): if the current position is more optimum than \( pb^{(i)} \), then \( pb^{(i)} = \) current position, else \( pb^{(i)} = pb^{(i-1)} \). 
4. Compare all the position of all agents based on distance function in (1). The most optimum, in this case is the closest to the goal, is the \( gb^{(i)} \). 
5. Evaluate the \( i+1 \)-th iteration velocity using (2) 
6. Update agent location with (3): 
7. If the stopping criteria is met then stop, else go to step 2.

2.2. Fuzzy Logic

Logic is reasoning conducted or assessed according to strict principles of validity. Logic can be grouped into 2 groups; classical logic (crisp logic) and fuzzy logic (vague). The fuzzy logic is used in this paper.

The idea of fuzzy is creating a logical system that has the ability to mimic human reasoning using a small number of rules and still producing a smooth output via a process of interpolation. The downside of classical logic is that it just has 2 conditions: yes or no, true or false, accept or reject. There is no definition for the “grey area” where we consider might be yes or might be no, fuzzy logic create the definition for this grey area.

A fuzzy set is a soft-bordered space to which all elements with a certain quality belong, up to a certain extent. It forms rules that are based upon multi-valued logic and so introduced the concept of set membership. With fuzzy logic, an element could partially belong to a set and this is represented by the set membership function.

2.3. Fuzzy Particle Swarm Optimization

The classic particle swarm optimization agent movement is based on the stochastic movement towards its personal best and the global best position. Its tendency to move towards \( pb \) or \( gb \) is simply based on the values of \( c \) that we set at the beginning. The agent does not have any freedom to choose its tendency. Fuzzy particle swarm optimization create the fuzzy set for the value of \( c \), so the agent can “think” about to move more to \( pb \) rather than \( gb \) or vice versa during the iteration. The thinking part is based on the \( pb \) and \( gb \), and define as a fuzzy set as given in Figure 2 below.
Irianto dkk: Simulation of Robot Movement in 2-Dimensional Space Using Fuzzy-Particle Swarm Optimization

The vertical axis represents the values of \( c \), the blue line represents the \( c_1 \) values and the red one is \( c_2 \). The values of \( c_1 \) is equal to 1 if \( pb \) is significantly more optimum than \( gb \), and the opposite the value of \( c_2 \) is equal to 1 if \( gb \) is significantly more optimum than \( pb \). The crossing area in the middle is the fuzzy condition, i.e. if the blue line is more than red line then the \( pb \) is more optimum than \( gb \), but the movement of the agent is not only towards \( pb \) it also towards \( gb \) but with less tendency.

2.4. Robot Movement Simulation

Scratch programming software (https://scratch.mit.edu/) is used to develop the movement of autonomous mobile robot simulation in 2-dimensional space based on classic PSO and fuzzy-PSO methods. It is a program of your own interactive stories, games, and animations — and share your creations with others in the online community.

The simulation run 20 autonomous mobile robots which locations are generated randomly. The movement of 10 autonomous mobile robots (blue) is based on classic PSO and the others (yellow) is based on the fuzzy-PSO. The goal is to move the robots autonomously towards the target position as shown in Figure 3 below.

The Scratch coding for autonomous mobile robots movement using classic PSO method is shown in Figure 4, while the Scratch coding for autonomous mobile robots using Fuzzy-PSO method is shown in Figure 5 below.
3. Results and Discussion

This section discusses the result of the simulation. It has several data including the value of the positions of robots, the distance to the target and the number of iterations until the robots arrive at the target. The goal of this simulation is to find which robots have fewer iterations to arrive at the target. Robots which have fewer iterations means arrive at the target faster. The result of data was drawn from two (2) group of robots, first robots with classic PSO movement methodology and second robots with Fuzzy PSO movement methodology. The position of mobile robot agents was arranged randomly as they move to the target.

Table 1 and Table 2 show the data drawn from initial position (iteration 0) for each group of mobile robot agents.

Table 1: Initial position and distance of robot agents with classic PSO

| Agent | Position of Agent | Distance |
|-------|------------------|----------|
| 1     | 132.001, 38.595  | 285.566714 |
| 2     | 141.324, 45.254  | 286.783245 |
| 3     | 149.908, 49.353  | 296.285489 |
| 4     | 134.893, 33.84   | 291.347667 |
| 5     | 143.559, 45.122  | 298.793697 |
| 6     | 140.308, 47.506  | 286.731085 |
| 7     | 148.333, 43.291  | 292.516871 |
| 8     | 141.521, 38.291  | 288.085963 |
| 9     | 129.001, 55.997  | 285.726863 |
| 10    | 139.1, 34.087    | 288.60674 |

Table 2: Initial position and distance of robot agent with fuzzy-PSO

| Agent | Position of Agent | Distance |
|-------|------------------|----------|
| 1     | 124.397365, 32.011665 | 289.822911 |
| 2     | 136.231863, 55.3384252 | 288.207317 |
| 3     | 121.433417, 48.365621 | 291.138684 |
| 4     | 142.523946, 47.652682 | 288.82503 |
| 5     | 141.516261, 49.273568 | 288.542628 |
| 6     | 141.40355, 43.406074 | 286.144939 |
| 7     | 140.792614, 48.104273 | 287.413167 |
| 8     | 130.067065, 56.486163 | 289.337056 |
| 9     | 130.197852, 39.553939 | 287.441579 |
| 10    | 126.448506, 29.430313 | 288.520203 |

The simulation results for each iteration are captured as shown in Figure 6. It is shown that after 20 iterations all fuzzy-PSO mobile robot agents (yellow colour) were closer to the target compared with the classic
PSO mobile robot agents (blue colour). When the simulation is continued, all fuzzy-PSO mobile robot agents arrived at the target location in 40 iterations.

The simulation was repeated until 30 replications. Each simulation was limited to 50 iterations only and the mobile robot agents considered arrived at the target when all the mobile robot agents touch the target. The result of 30 simulation replications was summarized in Table 3 below.
Table 3: Simulation result of 30 replications

| Replication | Number of iterations until all mobile robot agents arrive at target |
|-------------|---------------------------------------------------------------|
|             | Fuzzy PSO agents | Classic PSO agents |
| 1           | 40               | >50                |
| 2           | 24               | >50                |
| 3           | 31               | >50                |
| 4           | 28               | >50                |
| 5           | 35               | >50                |
| 6           | 39               | >50                |
| 7           | 41               | >50                |
| 8           | 30               | >50                |
| 9           | 27               | >50                |
| 10          | 30               | >50                |
| 11          | 41               | >50                |
| 12          | 37               | >50                |
| 13          | 50**             | >50                |
| 14          | 34               | >50                |
| 15          | 29               | >50                |
| 16          | 48               | >50                |
| 17          | 41               | >50                |
| 18          | 29               | >50                |
| 19          | 35               | >50                |
| 20          | 24               | >50                |
| 21          | 36               | >50                |
| 22          | 26               | >50                |
| 23          | 32               | >50                |
| 24          | 32               | >50                |
| 25          | 21               | >50                |
| 26          | 20*              | >50                |
| 27          | 30               | >50                |
| 28          | 43               | >50                |
| 29          | 40               | >50                |
| 30          | 32               | >50                |

From 30 replications, the least number of iterations until all fuzzy-PSO mobile robot agents arrive at the target was 20 iterations, while the largest number of iterations until all fuzzy mobile robot agents arrived at target was 50 iterations. Classic PSO mobile robot agents still not arrived at the target location at 50 iterations for all replications.

4. Conclusion

Fuzzy PSO has been proven to be a more efficient algorithm in terms of convergence compare with the classic PSO. It is shown by the simulation that fuzzy-PSO can reach the goal within in average 30 iterations where the classical PSO the autonomous mobile robot agents still not arrived at the target location in 50 iterations. In average, the autonomous mobile robot fuzzy-PSO agents will arrive at the target location by more than 40% faster compared to classic PSO agents.

Although it is shown that fuzzy-PSO is a better approach in CRS, however, its reliability needs to be tested through other random positions or space with obstacles. For further research, a 3D simulation or real robot could be done to strengthen these findings.

ACKNOWLEDGMENTS

We thank to Universiti Teknikal Malaysia Melaka, Faculty of Electrical and Electronic Engineering Technology of Universiti Teknikal Malaysia Melaka and Faculty of Engineering, Department of Industrial Engineering, Bina Nusantara University for providing facilities to support this research. We also thank the reviewers for their comment on earlier manuscript, although any errors are our own and should not tarnish the reputations of these esteemed persons.

REFERENCES:

[1] R. Eberhart and J. Kennedy. "A new optimizer using particle swarm theory" in Micro Machine and Human Science, MHS95., Proceedings of the Sixth International Symposium
Irianto dkk: Simulation of Robot Movement in 2-Dimensional Space Using Fuzzy-Particle Swarm Optimization

on, IEEE, pp. 39-43. 1995.

[2] Kennedy, J. “Particle swarm optimization”, Encyclopedia of machine learning, Springer US, pp 760-766. 2001.

[3] K. A. M. Annuar, Irianto, M. H. Harun, M. F. M. A. Halim, I. M. Saadon, and N. A. Rahman. “Particle Swarm Optimization (PSO) for Simulating Robot Movement on Two-Dimensional Space Based on Odor Sensing”, Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 9(2-6), pp. 79-83. 2017.

[4] Irianto. "A Preliminary Report on the Utilization of PSO for Solving the Hamiltonian Systems", Australian Journal of Basic and Applied Sciences 8 (5) S, pp. 370-374. 2014.

[5] Y. Shi, and R. C. Eberhart. "Empirical study of particle swarm optimization" in Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, vol. 3, IEEE, pp. 1945-1950. 1999.

[6] R. Poli, J. Kennedy and T. Blackwell. "Particle swarm optimization", Swarm intelligence 1, no. 1, pp. 33-57, 2007.

[7] R. H. Shehata, S. F. Mekhamer, N. El-Sherif and M. A. L. Badr, “Particle Swarm optimization: Developments and application fields”, International Journal on Power Engineering and Energy, Vol. 5. 2014.

[8] L. A. Zadeh, "Fuzzy logic", Computer 21.4, pp. 83-93. 1988.

[9] L. A. Zadeh, "Fuzzy logic = computing with words", fuzzy systems 4, no. 2, pp. 103-111. 1996.

[10] Y. Shi, Yuhui, and R. C. Eberhart. "Fuzzy adaptive particle swarm optimization" in Evolutionary Computation, 2001. Proceedings of the 2001 Congress on, vol. 1, IEEE, pp. 101-106. 2001.