Study comparison between firefly algorithm and particle swarm optimization for SLAM problems

Mounia Janah[1*], Yasutaka Fujimoto[1**]
1 Yasutaka Fujimoto Laboratory, Electrical and computer engineering departement, Yokohama National University, Yokohama, Japan
*E-mail: mounjanah@gmail.com
**E-mail: fujimoto@ynu.ac.jp

***Abstract*—Simultaneous localization and mapping abbreviated in SLAM is a procedure used to elaborate autonomous mobile robots that can construct a map of an unknown surroundings and simultaneously use that map to calculate its own position. There is many implementations of methods for solving the SLAM problem. In this context, Firefly Algorithm is inspired by fireflies behavior in nature and is one of the latest models same as PSO that is the abbreviation for Particle Swarm Optimization which is inspired by bird flocking or fish schooling and described as population depend on optimization process. We conducted a sequence of experiments using each algorithm and the results of those experiements were evaluated and compared to find the best solutions.**

***Keywords*—FA, LRF, PSO, SLAM**

I. Introduction

The Simultaneous Localisation and Mapping problem request for a mobile robot if it is possible to construct a consistent map of an unknown surroundings and determine it's own location within the same map simultaneously. Over the past decade, The solution of the SLAM problem has been one of the remarkable successes of the robotics community.

SLAM has been developed and determined as a abstract problem in a number of different models. However, consequential problems continue in basically accomplishing more general SLAM solutions and particularly as part of a SLAM algorithm in building and using maps [1]. This paper aims to compare the firefly algorithm with Particle Swarm Optimisation, there are test problems that can be used to compare between them, experiments will be presented applying these algorithms for solving SLAM problem and comparing the results.

II. Particle Swarm Optimization (PSO) and Firefly optimization Algorithm (FA)

Particle Swarm Optimisation (PSO) constructed a set of possible problem solutions as particles that are moved through a problem area, a particle updates its velocity and finally its position in the search area Fig. 1.

(1) is the velocity and (2) is the position of the i^{th} particle at the k^{th} step :

$$v_i^k = wv_i^{k-1} + c_1 r_1(p_i - x_i^{k-1}) + c_2 r_2(p_n x_i^{k-1}) \quad (1)$$

$$x_i^k = x_i^{k-1} + v_i^k \quad (2)$$

where v_i^k and x_i^k describe velocity and position for the i^{th} particle in the k^{th} step of the PSO algorithm, p_i represents the local best location of the i^{th} particle and p_n represents the global best location of all particles. the acceleration coefficients parameters are c_1 and c_2, and the arbitrary numbers taken from an homogeneous distribution are r_1, r_2 [2]. The parameter w is called inertia weight and was initiated in [3] to manage how much devotes the current velocity of the particle to its velocity in the next iteration. It plays the role of adjusting the balance between the global and local searchs [2].

We set the following parameters:
- Number of particle: m = 100
- Weight coefficients: w = 0.729, c_1 = 1.4955, c_2 = 1.4955
- Number of iteration: K_{max} = 500

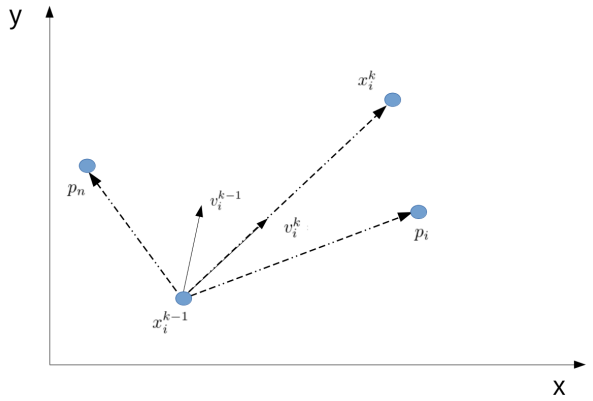


Fig. 1: PSO process

A. Procedure of the first test

To find the right parameters for particle swarm optimization algorithm we test our algorithm for different values of weight coefficients :

w = 0.6; 0.7; 0.729
c_1 = 0.8; 2.8 ; 1.4955
c_2 = 0.8 ; 1.3 ; 1.4955

We test the parameters by using sphere function (5) with : $-5.12 \leq x_i \leq 5.12 \Rightarrow$
$-5.12 \leq x \leq 5.12$; $-5.12 \leq y \leq 5.12$; $1 < i < n$

The results of Fig. 2 and Fig. 3 shows when we set w = 0.729, c_1 =1.4955, c_2 = 1.4955, the particle swarm optimization algorithm converge better

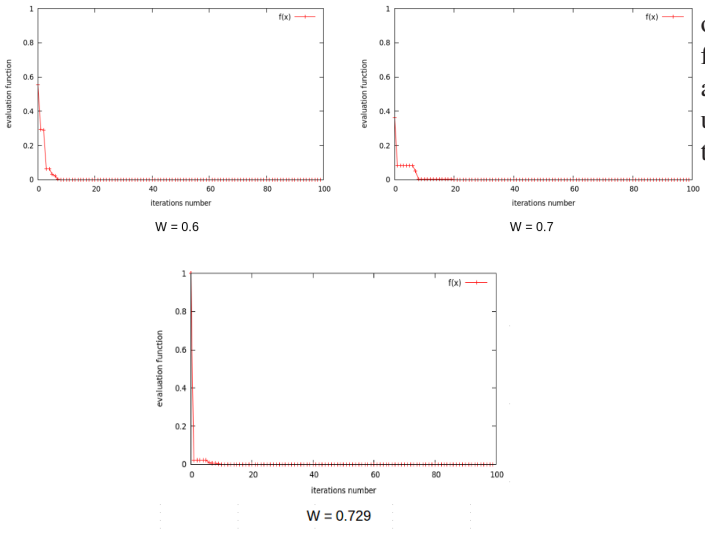


Fig. 2: Results from first test when we change the value of w

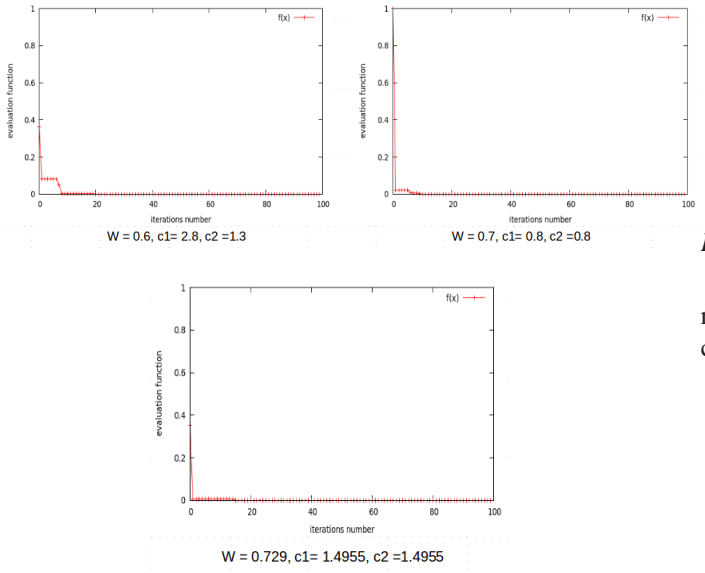


Fig. 3: Results from first test when we change the values of w, c_1 and c_2

In other hand, Firefly Algorithm (FA) depends on the glowing paradigm of fireflies and the attractiveness of a firefly is compatible to its lightness, and this decreases with the observed distance Fig. 4. The fluctuation of attractiveness with distance r is shown as below :

$$\beta = \beta_0 e^{-\gamma r^2} \tag{3}$$

where γ is the luminous abstraction coefficient of the medium and β_0 is the attractiveness at distance $r = 0$. The movement of i^{th} firefly to a brighter j^{th} firefly which is attracted to is

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}(x_j^t - x_i^t) + \alpha_t \epsilon_i^t \tag{4}$$

where respectively x_i^t and x_j^t are the positions of fireflies i and j at time t and α_t is the randomisation parameter that can be related to rate the random element and ϵ_i^t is a vector of random numbers [2]. This random vector is mostly peaked from either Gaussian or uniform distribution, but can also be applied to other probability distributions. The second term is used for the attraction to the lighter neighbour and the third term is dedicated to randomisation.

We set the following parameters:

- Number of firefly: m = 100
- Weight coefficients: $\beta_0 = 0.2$, $\gamma = 6.5$, $\alpha = \text{rand}(0, 1)$
- Number of iteration: K_{max} = 500

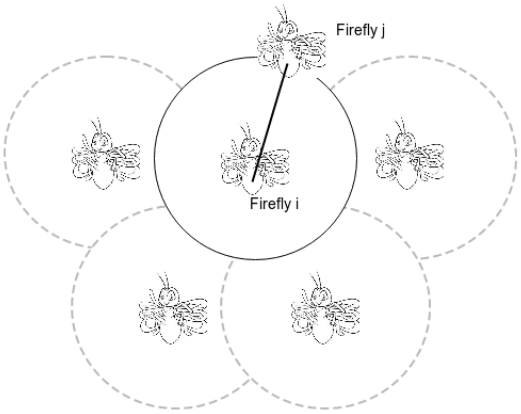


Fig. 4: FA process

B. *Procedure of the second test*

To find the right parameters for Firefly optimization algorithm we test our algorithm for different values of weight coefficients :

$\beta_0 = 0.2, 1$

$\gamma = 0.8, 1, 2.5, 4.5, 6.5, 7$

$\alpha = 0.2, \text{rand}(0, 1)$

We test the parameters by using sphere function (5)

With: $-5.12 \leq x_i \leq 5.12 \Rightarrow$

$-5.12 \leq x \leq 5.12$; $-5.12 \leq y \leq 5.12$; $1 < i < n$

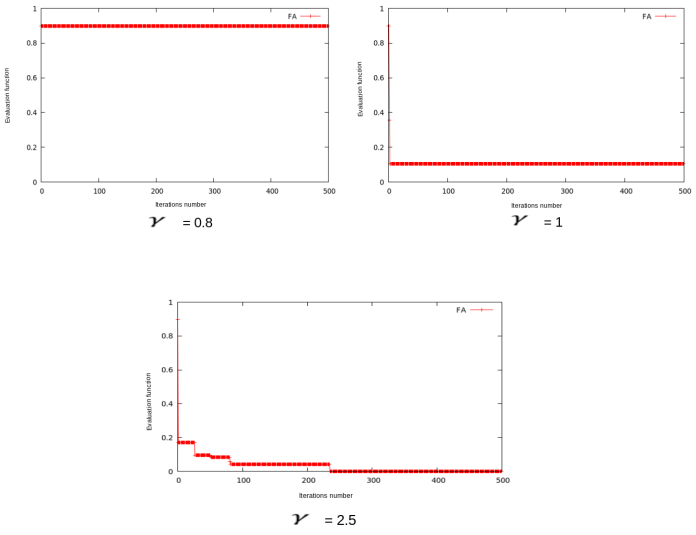


Fig. 5: Results from second test when we change the value γ = 0.8; 1; 2.5

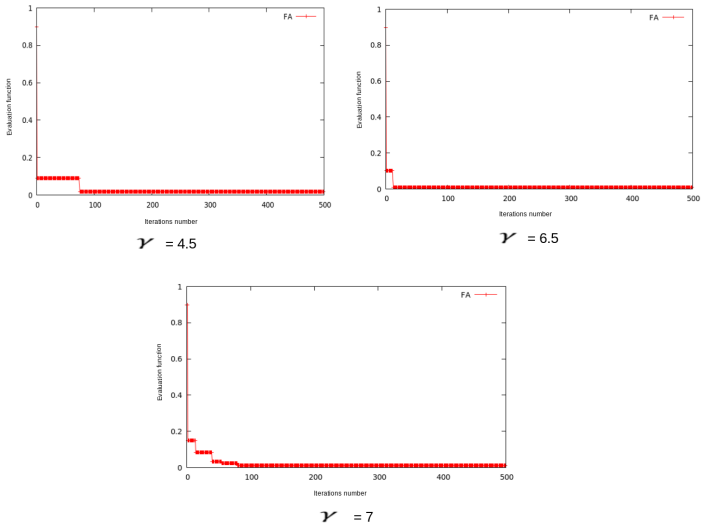


Fig. 6: Results from second test when we change the value γ = 4.5; 6.5; 7

The results of Fig. 5 and Fig. 6 shows, when we set $\gamma = 6.5$; $\beta_0 = 0.2$; $\alpha = \text{rand}(0, 1)$ the Firefly optimization algorithm converge better

Testing on Benchmark functions Fig. 7:

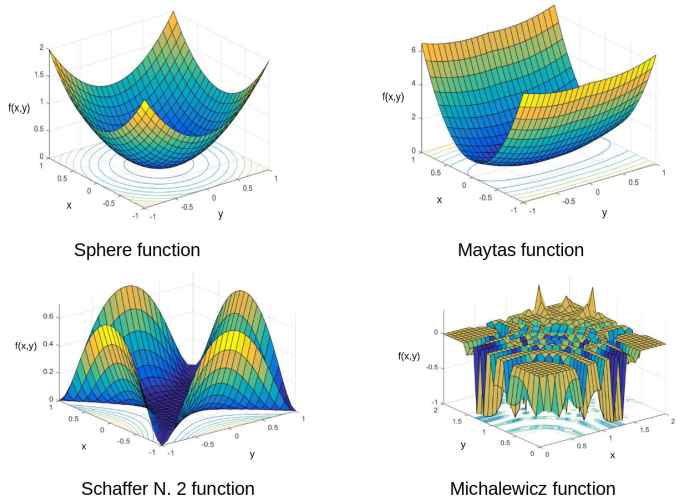


Fig. 7: 3D plotting of Benchmark functions

- Sphere function :

$$f(x) = \sum_{i=1}^{n} x_i^2 \tag{5}$$

- Maytas function:

$$f(x) = 0.6(x^2 + y^2) - 0.48xy \tag{6}$$

- Schaffer function N. 2:

$$f(x) = 0.5 + \frac{\sin^2(x^2 - y^2) - 0.5}{[1 + 0.001(x^2 + y^2)]^2} \tag{7}$$

TABLE I
TABLE OF EXECUTION TIME OF BOTH PROCESS

Functionsl	PSO	FA
Sphere fct	0.1s	0.45s
Maytas fct	0.4s	2.88s
Schaffer fct	0.5s	7.49s
Michalewicz fct	0.16s	0.76s

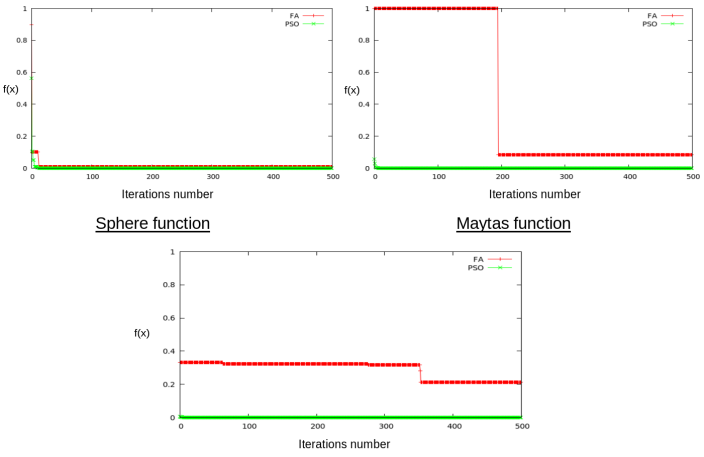


Fig. 8: Results of testing on Benchmark functions

- Michalewicz function :

$$f(x) = -\sum_{i=1}^{d} \sin(x_i)[\sin(\frac{ix_i^2}{\pi})]^{2n} \tag{8}$$

with n = 10 and d = 1, 2, ...; in the region $10 \leq x \leq 0, 0 \leq y \leq 10$.

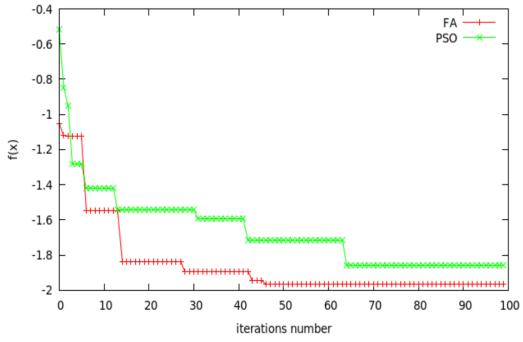


Fig. 9: Results of testing on Michalewicz function

From TABLE 1 and Fig. 8, PSO seems to be better in terms of speed of convergence and to find the global optimum.

III. Solving SLAM problem

In SLAM without the priority of knowing the location, the trajectory and the location are estimated SLAM is a procedure for an autonomous robot to construct simultaneously a map of an surroundings and use that map to estimate its position [4].

A grid map is rebuild from the information obtained from an LRF that belongs to the environment platform Fig. 17.

We modify the basic data extract from the Laser Range Finder sensor onto an occupied grid-map [5], the cells size of the grid-map is appointed at $20 \times 20\text{mm}^2$, and the cell value is described as MAP$(x, y) \in \{0, 1\}$ at the coordinates (x, y) and is initiated by 0. When MAP(x, y) = 1 that means there is an obstacle.

Respectively, the position and head angle of the robot are determined as x_0, y_0 and θ_0. The robots movement is described as $\delta x, \delta y, \delta\theta$ in a sampling period δt. Sampled data extracted from an LRF include movement information. hence, we can evaluate the robots movement by coordinating the latest data to a map built from past data. After the robot try to minimize Eq. (9) by using the PSO algorithm .

Depending on the number of coordinated points, the rate of points that have been matched $f(z)$ decided how well the local-map matches each of two the current-map or the global-map [5]:

$$f(z) = \frac{N_{valid} - N_{fit}(z)}{N_{valid}} \tag{9}$$

where $z = (\delta x, \delta y, \delta\theta)$ define the robots movement and N_{valid} describe the number of samples extracted from the LRF in each scan and detection distance is from 30 mm to 30,000 mm and N_{fit} is described as below [5]:

$$N_{fit} = \sum_{i=1}^{n_{max}} MAP(x_i, y_i) \tag{10}$$

$$x_i = d_i \cos(\theta_0 + \delta\theta) + x_0 + \delta x \tag{11}$$

$$y_i = d_i \sin(\theta_0 + \delta\theta) + y_0 + \delta y \tag{12}$$

where d_i is directly the distance from the sensor to the obstacle extracted from the LRF, (x_i, y_i) is the point coordinates on the map equivalent to the distance d_i and n_{max} is the number of valid points (719).

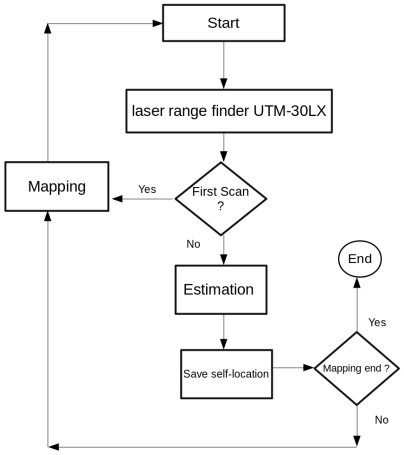


Fig. 10: System flow chart [6]

A. *Procedure of the third test*

We will test in different environment the simulator tool that we will use is Gazebo we will perform in ideal map Fig. 11

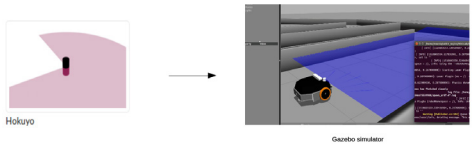


Fig. 11: Our environment in the simulator Gazebo

We can obviously add noise to the data extracted from Gazebo's sensors that by default examine the world completely to introduce a more practical environment because in the real world sensors display noise.

We will test on the case when we do a simple translation using the laser sensor with adding the gaussian noise : A mean of 0.0m and stddev of 0.02m, and the angular resolution by default equal to 1 deg.

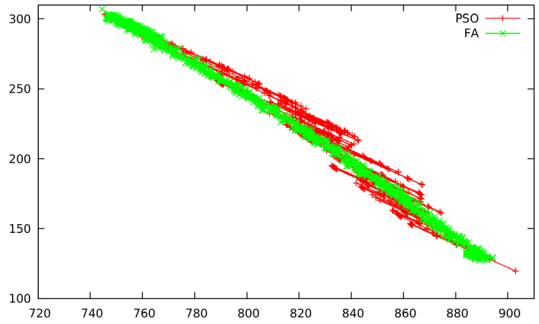


Fig. 12: trajectory of both (PSO/FA)

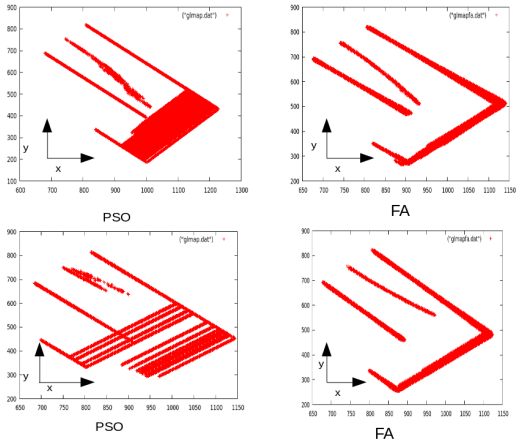


Fig. 13: results of the third test

From the simulation results Fig. 12, Fig. 13 we can see that FA outperfoms PSO.

When we conduct tests in terms of angular resolution and in terms of standard deviation of Gaussian noise we got the following results Fig. 14

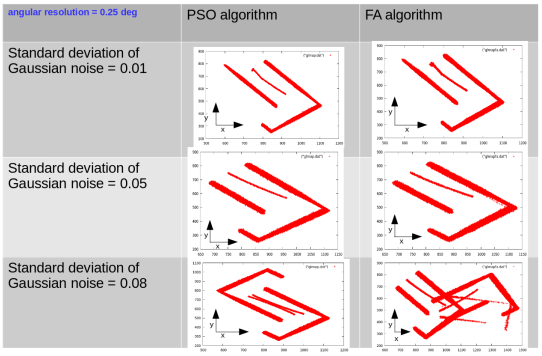


Fig. 14: results of the third test

B. *Procedure of the fourth test*

Testing in real world in two diffrent cases: The first and second examples Fig. 18 and Fig. 20 will test on the case when we do a simple translation, the third example Fig. 23, Fig. 24 will test on the case when we do (translation, rotation)

About the platform i am using 2D laser range finder UTM-30LX, iRobot Roomba 600, Raspberry Pi 3 and Power Bank Fig. 17

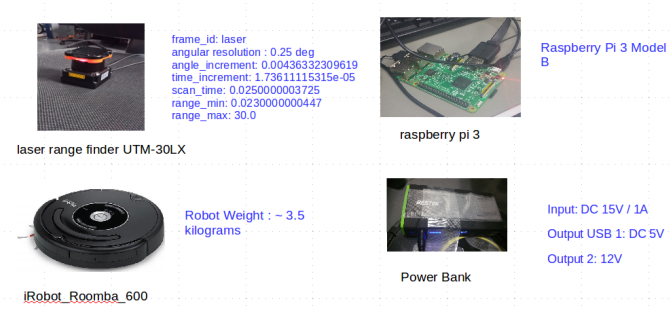


Fig. 15: Environment platform :2D laser range finder UTM-30LX + iRobot Roomba 600 + Raspberry Pi 3 + Power Bank

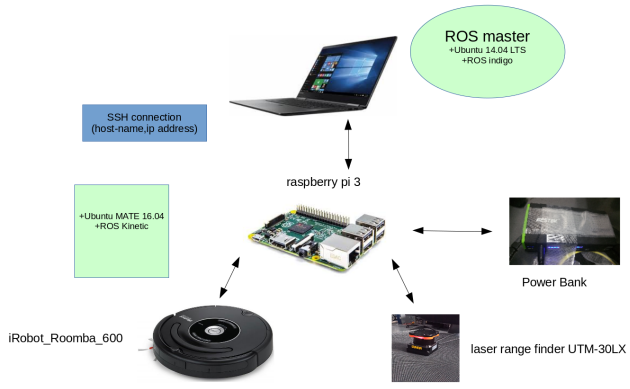


Fig. 16: Architecture of the environment platform

Extending the comparison between two algorithms for SLAM problem

Fig. 17: Environment platform :2D laser range finder UTM-30LX + iRobot Roomba 600

TABLE II
SPECIFICATIONS OF UTM-30LX

provision	specification
Size $(W \text{x} D \text{x} H)$	60 mm x 60 mm x 87 mm
Current	under 0.7 A
detection distance	3030,000 mm
angle of the scan	270 deg
angular resolution	0.25 deg
scanning time	25 ms/scan

Fig. 18: Example I environment

Number of iterations : we fixed the number of fireflies/particles to 40 and we varied the number of iterations from 5 to 500 Fig. 21

Number of particles/ fireflies : we fixed the number of iterations to 500 and we varried the number of fireflies/particles from 5 to 100 Fig. 22

In the benchmark functions, we have seen the PSO achieves better results than the FA Fig. 8, and does so in a small

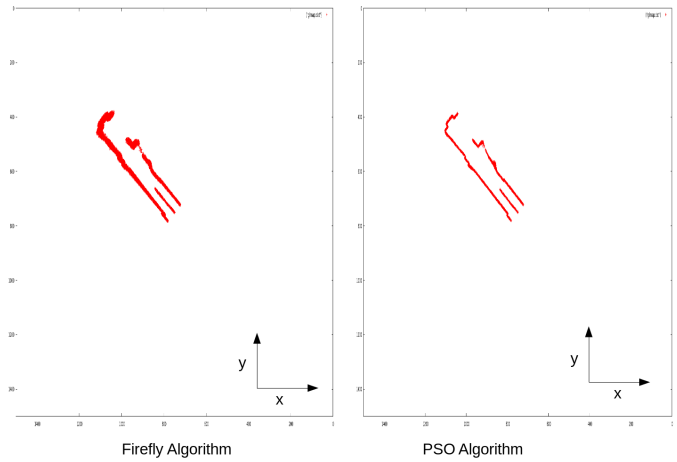


Fig. 19: Result of the first case (translation)

Fig. 20: Example II environment

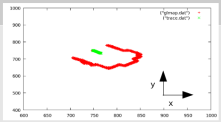
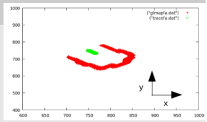
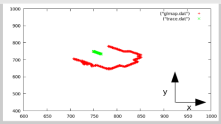
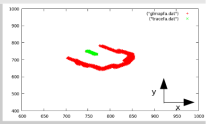
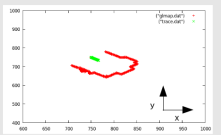
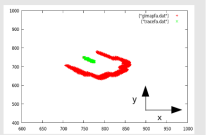
Number of fireflies/particles =40	Execution time/scan	Execution time/scan	Accuracy	Accuracy
Number of iterations	PSO	FA	PSO	FA
5	0.02s	0.09s		
50	0.12s	0.85s		
500	1.1s	8.6s		

Fig. 21: Results of the second case (translation) with varying the number of iterations

fraction of the time TABLE. 1, but in Fig. 9 firefly algorithm outperforms PSO but PSO still faster than FA in all cases . The results demonstrated in Fig. 19, Fig. 21, Fig. 22 and Fig. 25 The PSO and FA results are compared in Fig. 19, Fig. 21, Fig. 22 and Fig. 25. It appears that the PSO is the superior algorithm when it comes to solve SLAM problem. And when time is of the essence, the PSO returns results extremely fast.

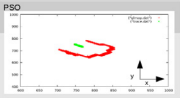
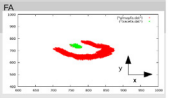
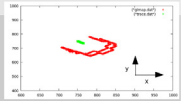
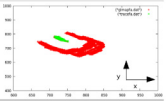
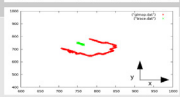
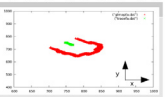
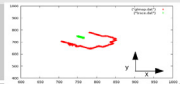
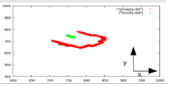
Number of iterations = 500	Execution time/scan	Execution time/scan	Accuracy	Accuracy
Number of particles/ fireflies	PSO	FA	PSO	FA
5	0.03s	0.01s		
10	0.033s	0.04s		
40	0.13s	0.8s		
100	0.25s	1.7s		

Fig. 22: Results of the second case (translation) with varying the number of fireflies/particles

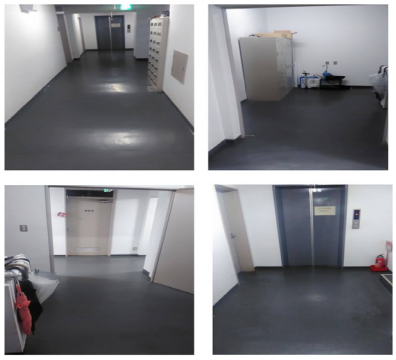


Fig. 23: Example III environment

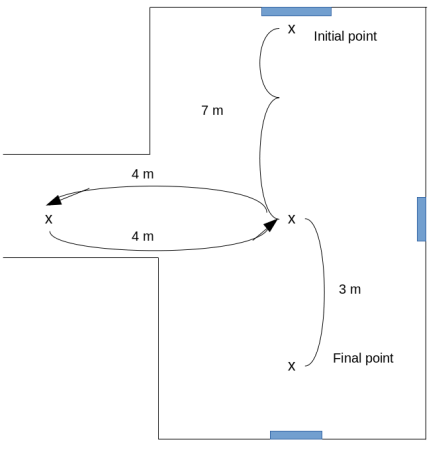


Fig. 24: Trajectory plan in Example III

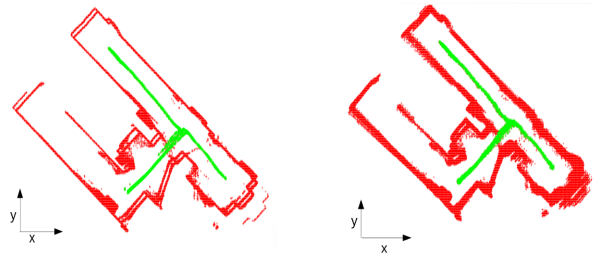


Fig. 25: Results of third case (translation,rotation)

IV. CONCLUSIONS

In this work,we compare between two procedures particle swarm optimization (PSO) algorithm and FireFly algorithm (FA), investigated their resemblances and inequalities. simulation results for determining the global optima of benchmark functions propose that particle swarm usually exceed Firefly algorithm in general cases and for solving Simultaneous Localisation and Mapping problems.

ACKNOWLEDGEMENT

This research was supported by Yasutaka Fujimoto Laboratory, we would like to thank the reviewers for their insights and we are also immensely grateful for their comments on an earlier version of the manuscript.

REFERENCES

[1] Hugh Durrant-Whyte, Fellow, IEEE, and Tim Bailey "1Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms," 2006.

[2] Michael Lohrer, "A comparison between the firefly algorithm and particle swarm optimization," Honors College Theses, Oakland University, pp. 11-16, Diss. 2013.

[3] Yuhui Shi, and Russell Eberhart, "A modified particle swarm optimizer," Proc.IEEE Int. Conf. on Evolutionary Computation, DOI:10.1109/ICEC.1998.699146, 1998.

[4] Jongdae Jung, Seung-Mok Lee, and Hyun Myung, "Indoor Mobile Robot Localization Using Ambient Magnetic Fields and Range Measurements," Robot Intelligence Technology and Applications 2. Springer, Cham, 137-143, 2014.

[5] Yudai Hasegawa, and Yasutaka Fujimoto, "Experimental Verification of Path Planning with SLAM," IEEJ Journal of Industry Applications, vol.5, no.3, pp. 253–260, 2016.

[6] Kazuaki Okada, and Yasutaka Fujimoto, "Grid-based localization and mapping method without odometry information," IEEE Industrial Electronics Society, proc. IEEE Industrial Electronics Society Annual Conference (IECON), PP.159-164, 2011.