Middle-East Journal of Scientific Research 24 (S1): 312-320, 2016
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2016.24.S1.65

Optimal Path Planning for Mobile Robots Using Particle Swarm Optimization and Dijkstra Algorithm with Performance Comparison

[1]M. Malathi, [2]K. Ramar and [3]C. Paramasivam and M. Malathi

[1]Department of Computer Science and Engineering,
Vaigai College of Engineering, Madurai, Tamilnadu, India
[2]Einstein College of Engineering, Tirunelveli
[3]Department of Mechanical Engineering, Thiagarajar College of Engineering, Madurai, India

Abstract: Path planning of mobile robot in a static environment is one of the most important problems in the field of mobile robot. The purpose of path planning is to find collision-free path for mobile robot between start and target position. However, the path planning problem shall be solved the two important issues such as the obstacles must be kept away obstacles and the length of path shall be minimized. So, it is wise to combine two (or) more heuristic techniques to solve these issues. In this paper, the watershed algorithm (WSA) is implanted along with particle swarm optimisation (PSO) for obtain optimal path between start and target position within a working environment using Matlab coding. Initially, the image of working environment is given as input to the watershed algorithm in order to get initial solution and then the output from the watershed algorithm will be given as input for particle swarm optimization technique in order to get optimal path to be followed by mobile robot to complete the specified task. Also, Dijkstra algorithm is proposed here, for the same working environment to obtain shortest path. Subsequently, the output performance through PSO compared with Dijkstra algorithm. This proposed methodology has direct application for mobile robot path planning such as material distribution within a manufacturing environment and pick-and-place task in a plane environment.

Key words: Particle Swarm Optimisation · Path planning · Dijkstra algorithm · Optimal path

INTRODUCTION

Mobile robot is one kind of robots which can able to move sense and react in a given environment [1]. Navigation of mobile robot is the control the movement from a start point to a target point in a given environment with obstacle avoidance capabilities. Based on the volume of information available about working environment, path planning can be classified into off-line and on-line path planning. In off-line or global path planning a robot possesses prior information about the environment and in on-line or local path planning a robot has no prior information about the environment [2].

There have been many algorithms for global path planning, such as artificial potential field, visibility graph, cell decomposition etc. Potential field has been used widely due to its simple structure and easy implementation, yet it still has some shortcoming. The cell decomposition, roadmaps have difficulty in solving RPP with complex environments due to their high cost of computation [3]. It has been used for hazardous target search applications such as landmine detection, fire fighting and military surveillance and effective robotic search problems [4].

Related Works: Nancy Arana Daniel *et al.* [5] have developed smooth path planning for mobile robot using particle swarm optimization with radial basis functions. Here, the MAKLINK graph is built to describe the working space of the mobile robot. Then sub-optimal path is obtained using Dijstra algorithm and the optimal path is obtained using particle swarm optimization with radial basic functions. Here, only the static obstacles are considered. It provides smooth and collision-free path which is to be followed by a mobile robot regardless of geometry of obstacle.

Corresponding Author: M. Malathi, Department of Computer Science and Engineering, Vaigai College of Engineering, Madurai-625122, Tamilnadu, India.

Maryam Yarmohamadi *et al.* [6] have developed a methodology to obtain a path for mobile robot using particle swarm optimization in a dynamic environment with mobile obstacle and target. Here, both Static and dynamic obstacles in circular shape are considered. It provides optimal path between start and goal position with escaping local optimum.

Kun Su *et al.* [7] have proposed a methodology for robot path planning based on random coding particle swarm optimization. Here, a sub-optimal collision-free path is obtained using Dijstra algorithm and the optimal path is obtained using random coding particle swarm optimization technique. It gives optimal path with better convergence speed and dynamic convergence compared to basic PSO.

Yuan-Qing Qin *et al.* [3] have implemented a procedure to find out the shortest path between start and goal position using Dijstra algorithm and the corresponding optimal path is obtained using PSO for mobile robot application. Here, the obstacles considered as polygonal shape and the MAKLINK graph is used to describe the working area for mobile robot. It provides better optimal path compared to basic PSO in terms of success rate and convergence speed.

Summary of Related Works: Here, few literatures are consolidated related with the mobile robot path planning using particle swarm optimization with Dijstra algorithm. It is necessary to obtain an optimal path with obstacle avoidance by considering some parameters such as time, energy and safety. Various working environment such as static and dynamic environment, fixed target, moving target are considered by many researchers. The circular, rectangular and some polygonal shapes are considered here, for their research works. These papers provide a better result in terms of computation time, shortest smooth and collision-free path with safety with higher success rate.

Watershed Algorithm: In watershed algorithm, a digital image is used and the value of each pixel of the image represents the elevation at that point. Here, a barrier is considered as a watershed lines or water rigid line [8]. An assumed simulated working environment is designed as shown in Figure 1 with circular shapes represent the different machines. A Matlab code has been written for implementing watershed algorithm to process the image of the working environment. The corresponding output from the watershed algorithm is shown in Figure 2 which shows the water rigid line.

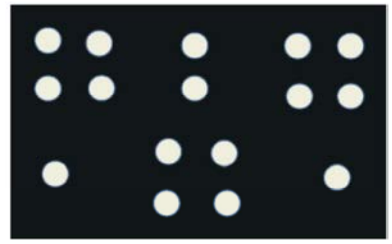

Fig. 1: Assumed simulated working environment

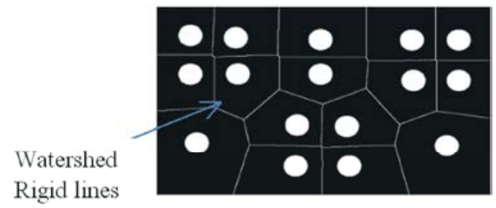

Fig. 2: Output image from watershed algorithm

Extraction of Branch and End Points: The output image of the working environment has many watershed rigid lines shown in Figure 2, which are going to act as path lines for mobile robot movement.

Initially, all these line segments are needed to be extracted, because the image data cannot be used directly. For this purpose, a Matlab code has been written to extract data points from the watershed rigid line. The extracted coordinate points in terms of pixels are given in Table 1 for the sample image considered.

Table 1: Coordinate points of branch and end points in pixels

Row Branch Points (Y)	Column Branch Points (X)	Row End Points (Y)	Column End Points (X)
131	130	131	1
137	135	291	1
281	135	1	135
316	206	525	183
389	220	1	292
132	290	525	397
137	293	1	504
231	293	525	607
392	395	1	672
396	397	147	799
272	398	311	799
227	503	-	-
137	504	-	-
148	509	-	-
390	574	-	-
330	585	-	-
145	672	-	-
147	674	-	-
292	674	-	-

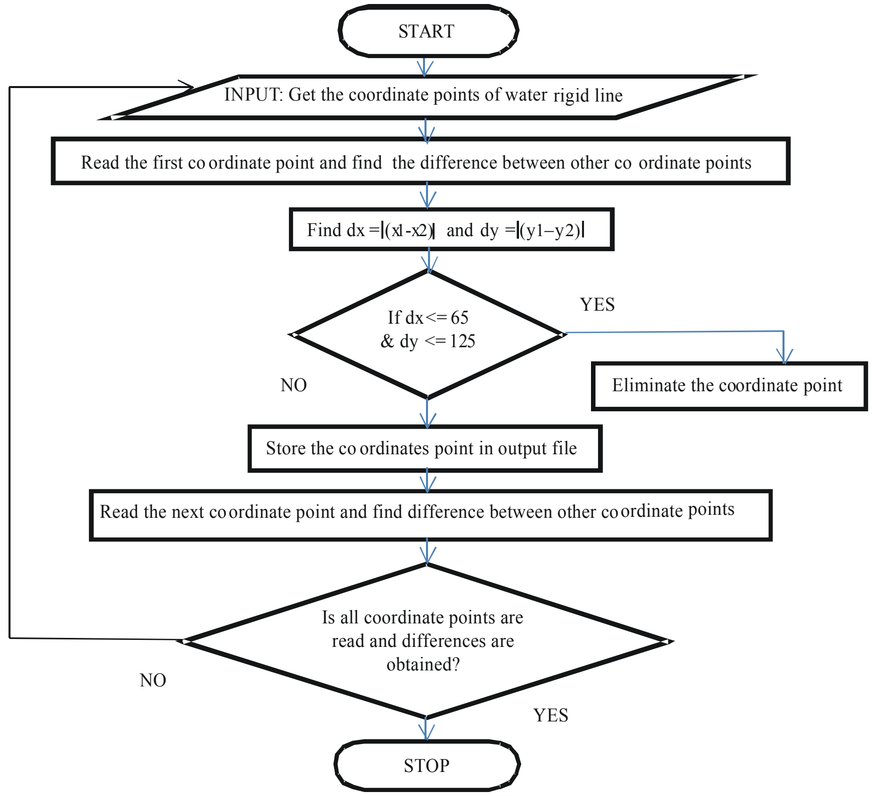

Fig. 3: Flow chart for reducing the coordinate points already extracted

Table 2: Output coordinate point data using Matlab code in pixels

Nodal No.	Row coordinate points	Column coordinate points
1.	131	130
2.	281	135
3.	389	220
4.	132	290
5.	231	293
6.	392	395
7.	227	503
8.	137	504
9.	390	574
10.	145	672
11.	292	674
12.	131	1
13.	291	1
14.	1	135
15.	525	183
16.	1	292
17.	525	397
18.	1	504
19.	525	607
20.	1	672
21.	147	799

Reduction of Coordinate Points Obtained: The branch and end points are extracted from water rigid line through Matlab code gives more number of points. If it is directly used for mobile robot path, defiantly leads to wrong solution and it take relatively more computational time. So, a Matlab code has been written for reducing the coordinate points which are more close to each other based on Euclidian difference along the X and Y directions and corresponding flow chart is shown in Figure 3. The output from the Matlab code with reduced number of coordinate points is given in Table 2.

Obtaining Center and Radius of the Circular Shapes Available in the Sample Image: There are two methods to find out the center and radius of circle such as phase coding method and two-stage technique. Due to the higher sensitivity level, there is a chance for detecting the false circles on the image by phase coding method. Hence, the two-stage technique is used to find out the center and radius of circles which are the representation of machines available within the working environment. The centers and radius of all circles are shown in Table 3. The flow chart for the two-stage Hough transform is shown in Figure 4 for finding centers and radius values in pixels.

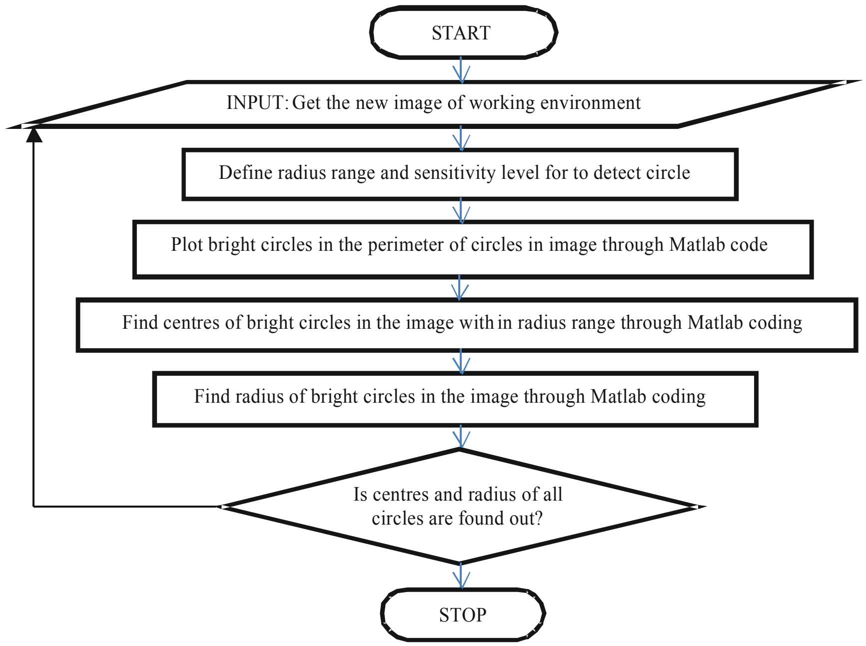

Fig. 4: Flow chart for finding centers and radius of circles using two-stage Hough Transform

Table 3: Centers and radius of all the circles available in sample image

S.No.	Coordinate values of centers in pixels: Column	Row	Radius in pixels
1	341.23	337.02	28
2	398.24	094.28	28
3	097.07	388.49	28
4	397.91	192.48	28
5	081.52	082.15	28
6	468.24	455.86	28
7	733.58	209.63	28
8	623.63	210.02	28
9	191.41	089.53	28
10	081.70	192.56	28
11	704.23	395.73	28
12	733.16	095.80	28
13	197.57	191.22	28
14	619.45	095.85	28
15	461.78	341.41	28
16	337.30	455.68	28

Particle Swarm Optimization: The Particle Swarm Optimization algorithm (PSO) is a population-based stochastic search algorithm and an alternative solution to the complex non-linear optimization problem [9, 10]. The PSO algorithm was first introduced by Dr. Kennedy and Dr. Eberhart in 1995 and its basic idea was originally inspired by simulation of the social behavior of animals such as bird flocking, fish schooling [11]. The formula for calculate velocity of particle is given in Equation 1 and position of particle is given in Equation 2. The process of PSO is initialized with group of random particles (solutions) and then searches for optimal solution by updating many iterations. The formula to calculate fitness function is given in Equation 3. The flow chart for the implementation of PSO is presented in the Figure 5.

$$Vi^{k+1} = Wvi^{k} + C1r1(Pbest - xi^{K}) + C2r2(gbest) - Xi^{K}) \quad (1)$$

The position of each particle is updated using following formula [12]:

$$xi^{K+1} = xi^{K} + vi^{k+1} \quad (2)$$

The fitness value of particle is calculated using following fitness function [13]:

$$F = L * (1 + beta * Violation) \quad (3)$$

where,

$$L = \sqrt{(x_i - x_g)^2 + (y_i - y_g)^2}; i = 1, 2, 3, \ldots \ldots, N$$

Dijkstra Algorithm: Dijkstra algorithm is used for finding the shortest path between nodes in a graph, which may represent working machine environment, road networks etc. It was conceived by computer scientist Edsger W. Dijkstra during 1956. The general flow chart of Dijkstra algorithm is shown in Figure 6.

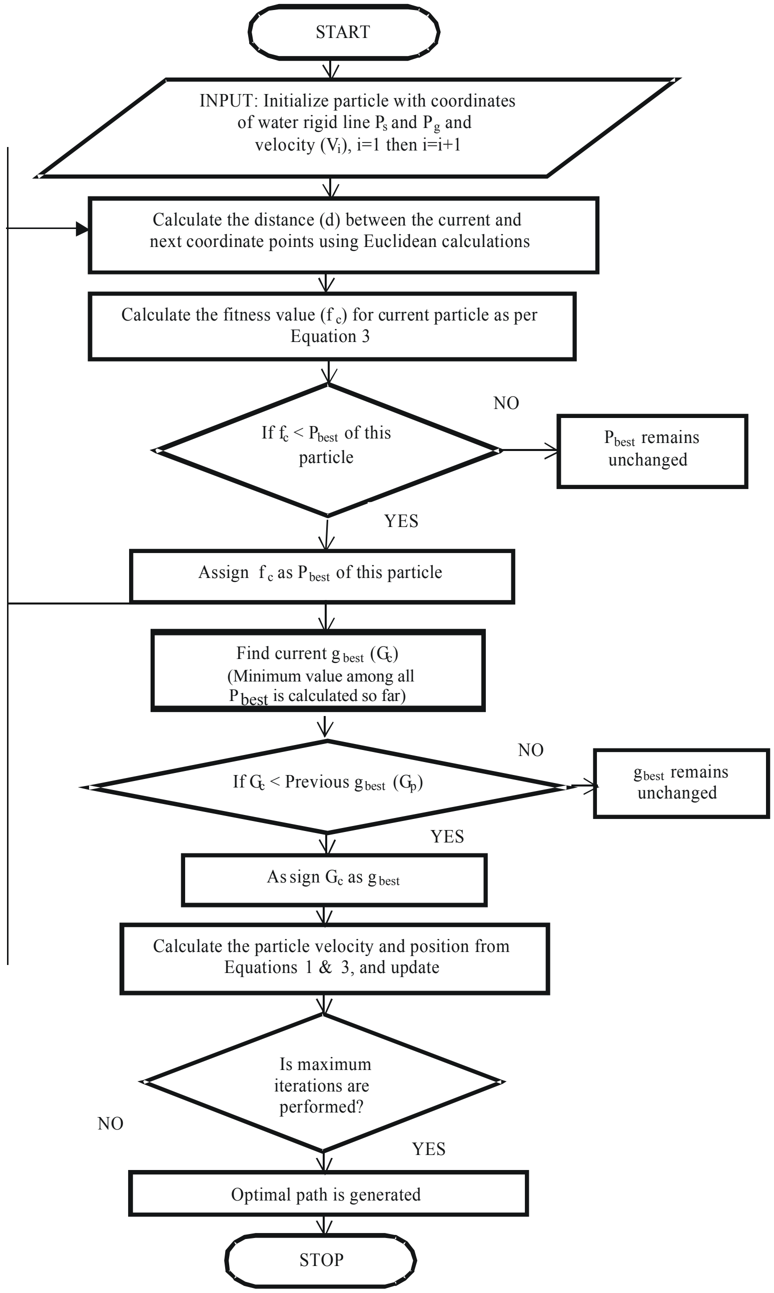

Fig. 5: Flow chart for Particle Swarm Optimization algorithm

The twenty one coordinate points obtained using the sample working environment image shown in Figure 1 from watershed rigid line are given as input to Dijkstra algorithm and the initial and target node is also defined randomly. Firstly, the initial node defined by user will be considered as current node and all other nodes considered as unvisited nodes. Then it will find out the tentative distance between current and neighbour nodes. The shortest distance of neighbour node with current node is identified and then this neighbour node gives shortest distance will be considered as new current node. This procedure will be continued for all other nodes until it reaches the target node. Finally, the path which has shortest distance is obtained from initial and target node through Matlab coding.

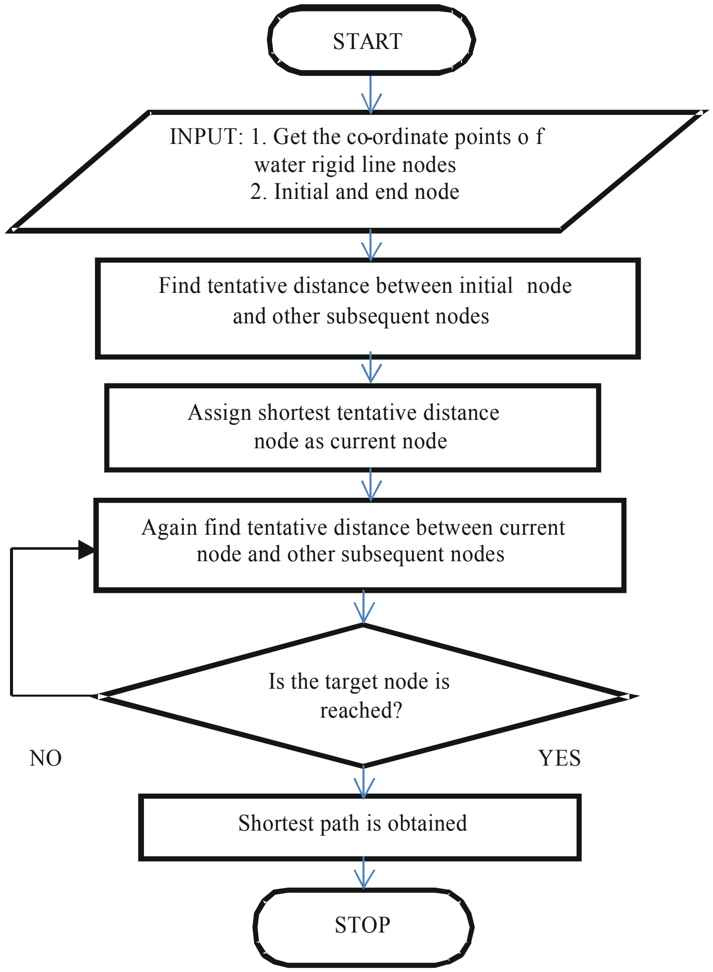

Fig. 6: Flow chart for Dijkstra Algorithm

RESULT AND DISCUSSION

Simulation Results of PSO: The simulation output of PSO algorithm through Matlab code is written for mobile robot path planning application using the sample image as shown in Figure 7. Here, the all the circular shapes are considered as different machines in a working environment. The input to the PSO will be centers and radius of the circles shown in Table 3. Then, the start and target point are fixed randomly by the user within the working environment. While finding the path, all the circular shapes are considered as obstacles. Finally, a collision-free path has been obtained using PSO procedure.

Here, the path length is calculated as 619.29 pixels for the sample image considered using Euclidian distance from start to target position. For obtaining the optimal path, it requires a number of iterations. The iteration will be continued based on the fitness function. The convergence of iteration based on fitness function value gives the optimal solution. The convergence graph between fitness value and the number of iteration is shown in figure 8. Whenever the deviation between fitness function value and iteration is less then, the iteration will be stopped.

Simulation Results of Dijkstra Algorithm: Here, the coordinate points extracted from the watershed rigid line will be given as input to Dijkstra algorithm. In this case, all the coordinate points obtained using watershed rigid line as given in Table 2 has been given as input data to this algorithm. Also, the start (672, 145) and target node coordinate values (183, 525) are fixed randomly within the image of the working environment. Finally, the shortest path is obtained with a distance of 1116 pixels using Dijkstra algorithm is shown in Figure 9.

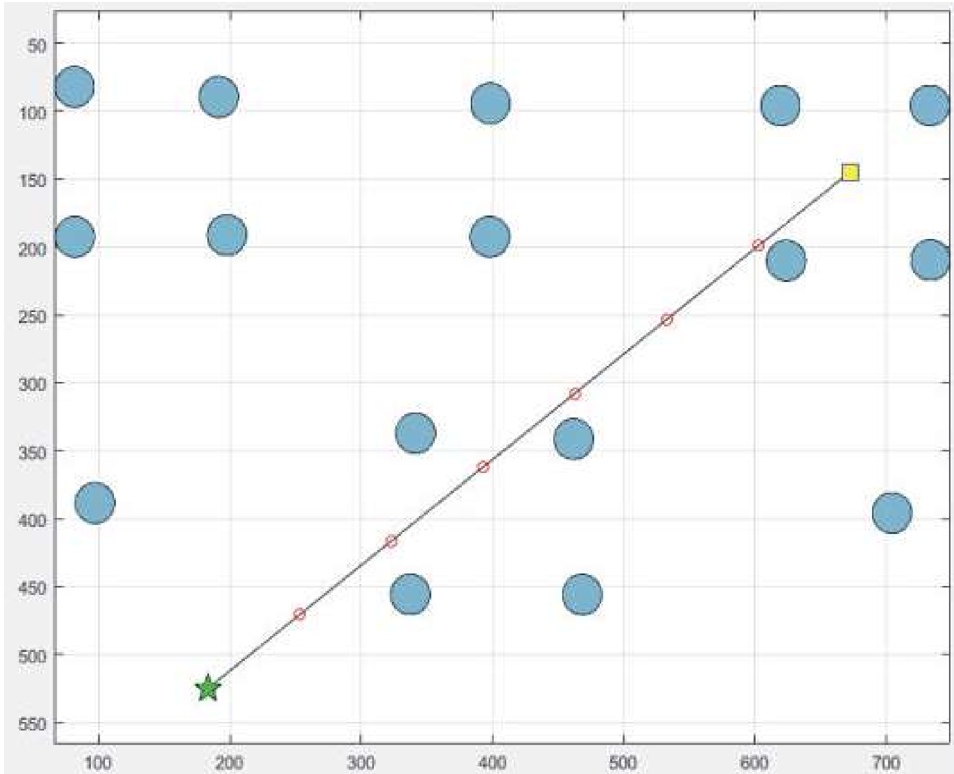

Fig. 7: Output image for PSO

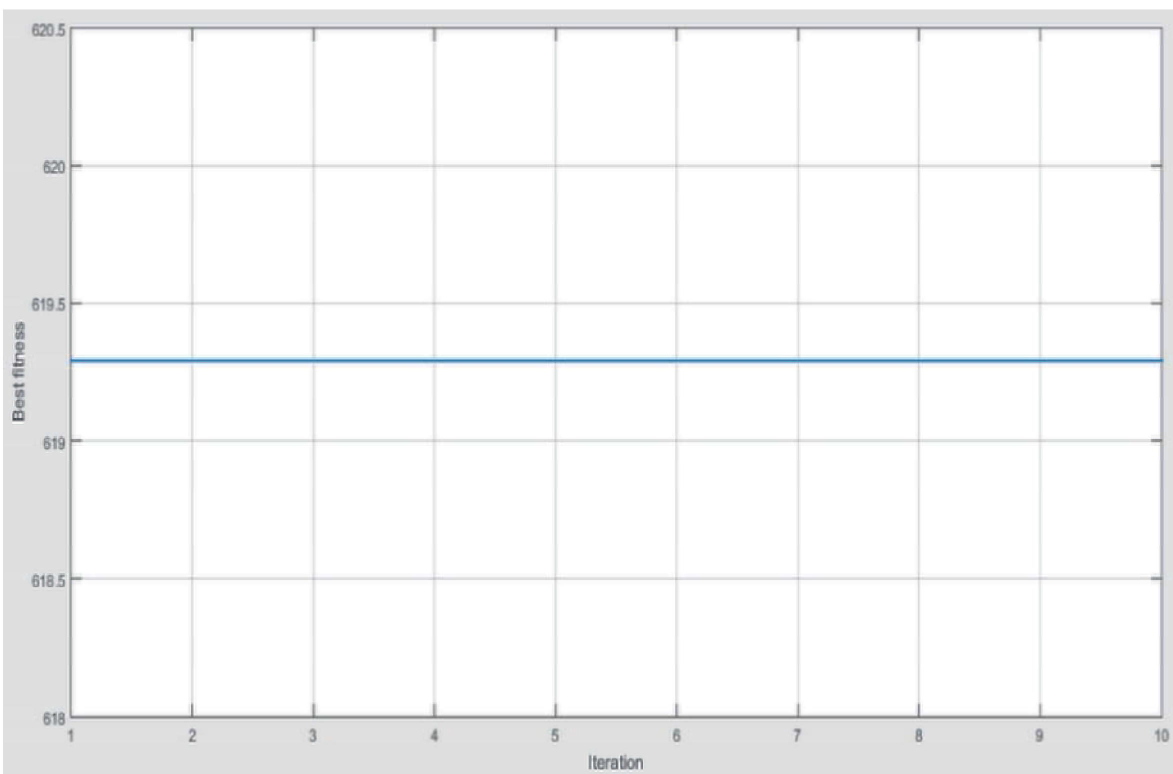

Fig. 8: Convergence graph between fitness and number of iterations

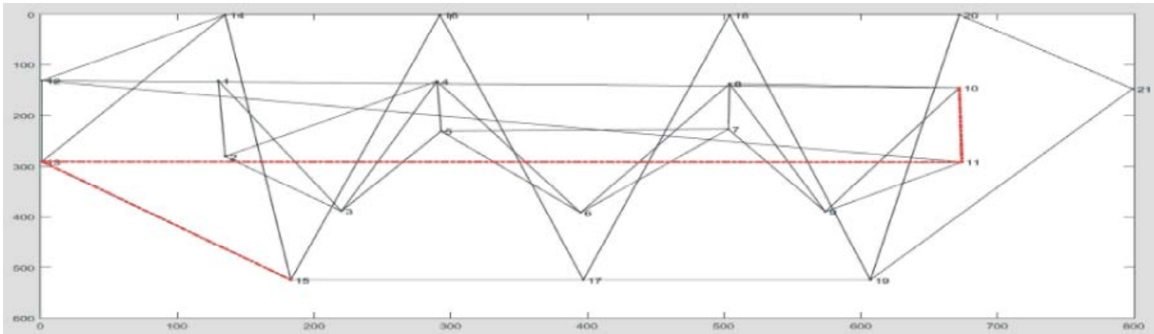

Fig. 9: Output image for Dijkstra Algorithm with shortest path

Table 4: Comparison between PSO and Dijkstra Algorithm

No. of Machines considered	Start and Target node in pixels	Dijkstra Algorithm Total Distance in pixels	Dijkstra Algorithm Time taken in sec.	PSO Total Distance in pixels	PSO Time taken in sec.	Inferences
5	S(178, 162) T(618, 199)	855.570	0.479	947.130	8	If No. of machines are less, the path length in Dijkstra algorithm is minimum and computational time is higher for PSO
9	S(514, 395) T(214, 1)	495.213	0.602	495.213	9	Here, path length taken by both algorithms are same
16	S(672, 145) T(183, 525)	1116.50	1	619.290	10	If No. of machines increased, path length taken by PSO is minimum
26	S(115, 162) T(732, 462)	2003.25	1	1139.20	11	Here, the path length minimum for PSO. As a whole, when the number of machines are more PSO will be more suitable

Performance Comparison: The performance of PSO and Dijkstra algorithm is compared along with inference is presented in Table 4. Here, four different trial simulations with 5, 9, 16 and 16 machines are performed using the corresponding procedures explained. In all cases, the start and target node are randomly fixed by the user for simulation in order to obtain the optimal path. Here, the two parameters are considered for the performance comparison like total distance from start to target node and computation time taken the corresponding algorithm and the comparison is given in Table 4.

CONCLUSION

The PSO along with Dijkstra algorithm is proposed in order to obtain the optimal path for the 2 dimensional image of the working environment. Here, a sample image of an environment with 16 machines is considered. The watershed rigid line values are the input to the PSO as well as Dijkstra algorithm. Initially, watershed algorithm is implemented to obtain the watershed rigid line by writing Matlab code. Subsequently, a Matlab code has been written to obtain the optimal path using the sample image through both algorithms. Finally, the performances of both algorithms with corresponding inferences are also completed.

REFERENCES

1. Mohammed Algabri, Hassan Mathkour, Hedjar Ramdane and Mansour Alsulaiman, 2015. Comparative Study of Soft Computing Techniques for Mobile Robot Navigation in an Unknown Environment. Computers in Human Behaviour, 50: 42-56.
2. Shahab Alam, Usman Rafique and Umer Khan, 2015. Mobile robot path planning in Static Environments using Particle Swarm Optimization. International Journal of Computer Science and Electronics Engineering, 3(3): 253-257.
3. Yuan-Qing Qin, De-Bao Sun, Ning Li and Yi-Gang Cen, 2004. Path planning for mobile robot using the particle swarm optimization with mutation operator. International conference on Machine Laming and Cybernetics, 4: 2473-2478.
4. Rakibul Islam, Tajmiruzzaman, Mahfuzul Haque Muftee and Sanowar Hossain, 2014. Autonomous robot path planning using particle swarm optimization in dynamic environment with mobile obstacles and multiple Target. International Conference on Mechanical, Industrial and Energy Engineering, pp: 1-6.

5. Nancy Arana Daniel, Alberto A. Gallegos, Carlos Lopez-Franco and Alma Y. Alanis, 2014. Smooth path planning for mobile robot using particle swarm optimization and radial basis functions, splines and Bezier curves. IEEE Congress on Evolutionary Computation, pp: 175-182.
6. Maryam Yarmohamadi, H. Haj Seyyed Javadi and Hossein Erfani, 2011. Improvement of robot path planning using particle swarm optimization in dynamic environments with mobile obstacles and Target. Advanced Studies in Biology, 3(1): 43-53.
7. Kun Su, YuJia Wang and XinNan Hu, 2015. Robot path planning based on random coding particle swarm optimization. International Journal of Advanced Computer Science and Applications, 6(4): 58-64.
8. Rafael C. Gonzalez and Richard E. Woods, 2009. Digital image processing. Pearson Education, 3rd Edition, pp: 769-777.
9. Martin Saska, Martin Macas, Libor Preucil and Lenka Lhotska, 2006. Robot Path Planning using Particle Swarm Optimization of Ferguson Splines. IEEE Conference on Emerging Technologies and Factory Automation, pp: 833-839.
10. Satyobroto Talukder, 2011. Mathematical Modelling and Applications of Particle Swarm Optimization: Mathematical Modelling and Simulation, Master Thesis, No: 2010:8, Blekinge Institute of Technology, Karlskrona, Sweden.
11. Kaiyou Lei, Yuhui Qiu and Yi He, 2006. A Novel Path Planning for Mobile Robots Using Modified Particle Swarm Optimizer. International Symposium on Systems and Control in Aerospace and Astronautics, pp: 981-984.
12. Amin Zargar Nasrollahy and Hamid Haj Seyyed Javadi, 2009. Using particle swarm optimization for robot path planning in dynamic environments with moving obstacles and Target. Third UKSim European Modelling Symposium on Computer Modeling and Simulation, pp: 60-65.
13. Mostapha Kalami Heris, S., 2015. Using PSO in Matlab for Mobile robot Path Planning. Khaje Nasir Toosi University of Technology, Iran.