Path Planning in Swarm Robots using Particle Swarm Optimisation on Potential Fields

Sanjay Sarma O V
Department of Mechatronics
School of Mechanical Engineering
SASTRA University, Thanjavur
Tamilnadu-613402

Vishwanath Lohit T
Department of Mechatronics
School of Mechanical Engineering
SASTRA University, Thanjavur
Tamilnadu-613402

Deepak Jayaraj
Department of Mechatronics
School of Mechanical Engineering
SASTRA University, Thanjavur
Tamilnadu-613402

ABSTRACT

This article presents a novelimplementation of Particle Swarm Optimisation(PSO)forfinding the most optimal solution to path planning problem for a swarm of robots. The swarm canvasses through the configuration space having static obstaclesby applying PSO on potential fields generated by the target. The best possible path by the momentary leaders of the group is retraced toget the solution. The designed algorithm was simulated on a specially developed simulator adhering to real time constraints and conditions faced by the mobile robots. The solutions for various configuration spaces are presented to verify the effectiveness of the algorithm.

General Terms

Mobile robots, Path planning, Swarm robots.

Keywords

Path planning, Configuration Space, Particle Swarm Optimisation (PSO), Potential Fields.

1. INTRODUCTION

Swarm Intelligence (SI), the intelligence arising as a result of emergence in global behavior also called collective behavior describes the coordinated behavior of large groups of similar systems and the emergent properties of these groups. Social insects are known to interact with each other to accomplish tasks which areusually beyond the scope of individual ones. Examples of Swarm Intelligence include Ant colonies [1], bird flocking, animal herding, bacterial foraging [2], fish schooling etc.

Swarm Intelligence-based techniques arefinding their place in a number of applications. Apart from their prominent use in optimisation, and their application in tuning PID controllers of both linear and nonlinear systems [3,4], they are also being used to control vehicles, in swarms for self-assembly and interferometry, for planetary mapping etc., A paper by M. Anthony Lewis et al [5] discuss the possibility of using swarm intelligence to control nanobots within the body for the purpose of killing cancer tumors. Artistic applications include using swarm technology as a means of creating complex interactive systems or simulating crowds. The application of Swarm Intelligence in Telecommunication Networks has also been researched, in the form of Ant Based Routing [6]. Also, its application in the field of Robotics is prominent.

By the definition of Swarm Robots by ErolSahin [7], a large number of new strategies in various robotic applications were proposed, especially in path planning. FengWeiXing et al [8], proposed algorithms for coordination of underwater swarm robots to realize aggregation, formation and flocking. Othman et al examined the emergence of aggregations from various movement models of simple agents, from which the idea of generation of artificial potential fields was drawn for this paper.

Particle Swarm Optimisation (PSO) algorithm an evolutionary computation technique developed by James Kennedy et al [9] inspired by social behavior of bird flocking or fish schooling is considered as the prime idea behind the proposed algorithm in path planning. PSO optimizes a problem by maintaining a population of candidate solutions called particles and moving these particles around in the search space according to simple formulae. The movements of the particles are guided by the best found positions in the search-space, which are continuously updated as better positions are found by them. PSO has a wide range of applications which include function optimisation, Control Systems [3,4], Artificial Neural Network, Fuzzy system control, mobile robotics etc.,

Innavigating a mobile robot in a configuration space to a distant target, it should execute the task while avoiding obstacles. A path planning algorithm should consider the configuration space parameters, the presence of obstacle and produce a continuous motion that connects a starting point and an ending point while avoiding collision with obstacles which may be previously mapped.O.Hachour's [10] paper on path planning strategizes the development of occupancy grid matrix which is a driving idea in the development of the simulator. Also, in real world situations, it is more probable to face irregular obstacles by the robot, the back ground of this idea was drawn from Raja et al's work [11].

Apart from traditional path planning techniques, many algorithms applying Swarm Intelligent techniques in particular Particle Swarm Optimisation over path planning problem were proposed.Qiang Zhao et al's [12] approach introduces chaos into the particle swarm optimisationfor path planning so as to intensify the local search ability and improve the solution precision.Kaiyou et al [13] proposed a modified particle swarm optimizer which harmonizes global search abilities and local search abilities to avoid premature problem and gain rapid convergence to the satisfactory solution.

Sedigheh et al [14] work converts the navigation problem into an optimisation one by associating an evaluation function for every particle, the best position is evaluated and the robot moves to the next calculated point to reach the goal which also showed positive results in dynamic environments. Through this work we are proposing an approach which

implements potential field method though PSO in a highly constrained environment with complex static configuration and its efficacy and efficiency was proved in a self-developed simulator

The proposed work's objective is to develop a simulation strategy and apply Particle Swarm Optimisation technique in order to obtain an optimum solution in path planning. It also accommodates all the real time constraints which bolsters the development and extension of other optimisation algorithms like Ants colony optimisation, Simulated Annealing, Genetic algorithm etc., over path planning.

This paper details the development of a simulation strategy which involves the formation of various occupancy grids and potential fields, application of PSO on the generated configuration space and tuning for better results. The assumptions made while developing the simulator are presented in section 2 and an introduction to the graphical design and conversion of the configuration space into obstacle matrixand power gradient graph matrix are briefed in sections 3 and 4 respectively. The application of PSO and leader path generation algorithm is presented in section 5. The results for various configuration spaces are presented in section 6 followed by conclusions.

ending point, while avoiding collision with obstacles in between. The methods of robot path planning are classified into two categories Global Path Planning and Local Path Planning [15]. Global path planning is based on the information of the environment obtained in prior. Local path planning is based on the information of the environment acquired by a robot's sensors in real time, while not colliding with the obstacles in the environment detected by these sensors.

Most of the existing methods of path planning assume that the environment is precisely known and so is offline. But while planning the navigation of the swarm robots in real-time and in a stochastic non deterministic environment, local online path planning is preferred. Hence, considering the real time constraints, the current work frames the problem of path planning in terms of local path planning by converting the designed space into an occupancy grid matrix.

The assumptions made in the work about the 2D configuration space are as shown in the figure 1. The swarm robots start at the green starting zone with a pre alignment, with respect to each other. The swarm should reach the end point marked in red while avoiding the obstacles (colored blue) in between. It must be noted that, any irregular shape of obstacle can be generated with the combination of 3 basic shapes, Triangle, Rectangle and a Circle by superimposing.

The swarm as a group following the Particle Swarm Optimisation technique, searches for the path towards the end point (red colored area). Normally Euclidean distanceisconsidered as the function to be minimized in such scenarios while applying PSO. This characteristic function was replaced with a power distribution function where the leader always tends to move towards the maximizing power direction.

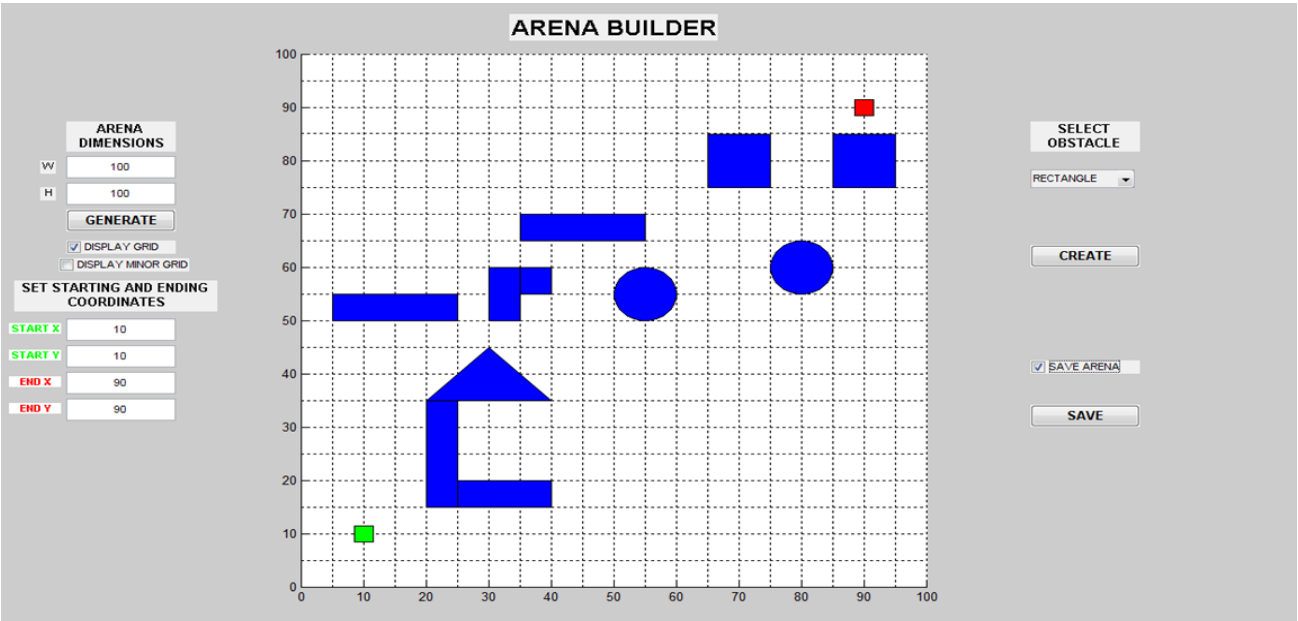

Figure 1: A sample configuration space and its builder

2. ASSUMPTIONS

A basic motion planning problem is to produce a continuous motion that connects a starting point and an

3. CONFIGURATION SPACE DESIGN

The exclusively developed simulator presented in this work was developed in MATLAB software using various basic functions of GUI and plot. MATLAB software gave us the flexibility of programming while using various inbuilt functions. Also, further developments of the simulator are being planned by using 3D simulations.

The configuration space builder also called the arena builder in the work is a special feature of the self-developed simulator which supports the design of a 2D space with three basic shaped obstacles, circle, rectangle and triangle. By superimposing these basic shapes many other irregular shaped obstacles can be obtained.

In real time situations, the robots are required to maintain a certain distance with the obstacles while planning their motion in order to avoid unexpected collisions. Hence, while designing the configuration space care was taken by providing a minimum gap between the obstacles. A screenshot of the arena builder is presented in figure 1.

4. CONVERSION INTO LOCAL PATH PLANNING PROBLEM

4.1 Obstacle matrix

To convert the global path planning approach to local path planning, the configuration space designed is converted into a matrix form whose size is equal to the size of the arena broken into units.

Each of the elements of matrix occupied by an obstacle is assigned a very low (negative) value, starting zone with a lower positive value and the ending zone with a very high positive value. This reduces the complex combination of obstacles to a simple occupancy grid matrix. A few layers around the obstacle are wrapped with a lower value in the matrix than the obstacle for the obstacle vicinity zone marking, where the number of layers surrounding the obstacle is proportional to the range of the sensors. The regenerated map from the matrix with the obstacles is as shown in the figure 2. The extrapolated region around the obstacle is in cyan.

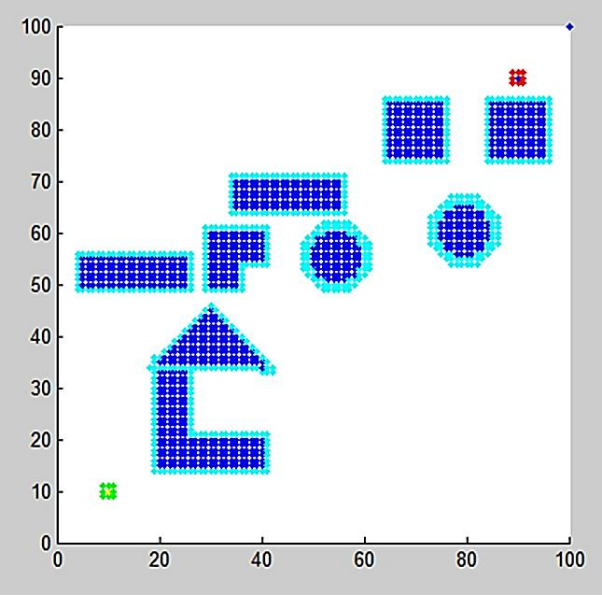

Figure 2: Obstacle Matrix

4.2 Power matrix

This idea of assigning a power source to the end point came up with the research papers published by Othman et al [16], which had a centric power source attracting the swarms towards it. In real time this power source can be a luminant source where the intensity of light decreases with distance.

The obstacle matrix generated previously contains free space (unoccupied space) with zero values. To create affinity of a robot towards the end point, a power gradient is generated with the destination as the power source. This simplifies the solution of path planning by programming the leader robots in moving towards the increasing gradient of power.

Considering the end point as the power source, with end point as the center concentric circle equations are generated and the points falling between two energy circles are given the energy of the smaller circle. The decrement rate of power across the circles is inversely proportional to the square of the distances.Every coordinate is assigned with power value in power matrix which is obtained by adding the power values to the obstacle matrix discussed previously. Hence the '0' values of the free space are replaced by the power values. No change is made to the starting point, ending point, obstacles and the boundaries.

The so developed power matrix is expressed in the form of a graph as shown in the figure. In the figure 3 the color intensity of green was varied proportional to the power at the coordinate.

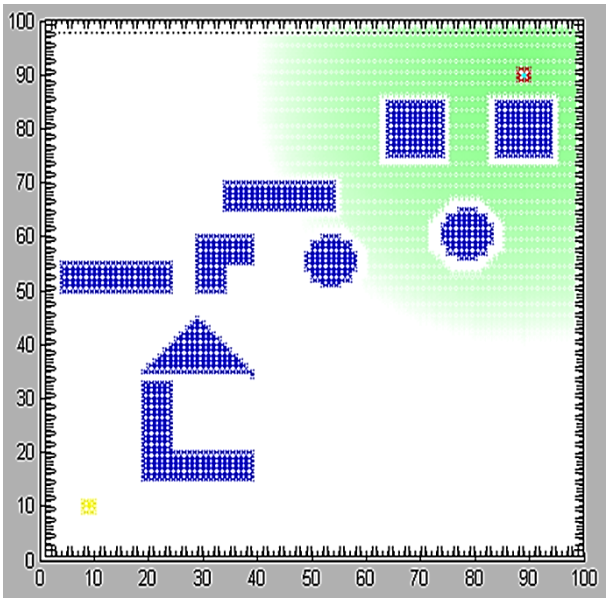

Figure 3: Power Matrix

4.3 Bot parameters

The parameters as shown in the table are set for each individual robot in the swarm.

Table 1.List of bot parameters

Parameter	Value
Shape	Point
Co-ordinate interval	1
Detection co-ordinates	8

5. THE ALGORITHM

5.1 Particle Swarm Optimizer

PSO is a population based optimisation technique. The system initialized with a population of random solutions (particles), searches for optima. PSO optimizes a problem by moving solution particles around the search-space according to simple formulae. The movements of the particles are guided by the best found positions in the search space, which are continually updated as better positions are found by the particles.

In the PSO algorithm, first the system is initialized with a population of random solutions for particles, and each particle is also assigned with a random velocity. PSO relies on the exchange of information between particles of the population in the swarm. In the iterative process, each particle adjusts its trajectory towards its best solution (fitness) that is its trajectory towards the best previous position attained by any member of its group. This value is called 'gbest'. Each particle moves in the search space with an adaptive velocity. The fitness function is used to evaluate the performance of particles to determine whether the best fitting solution is achieved. During the iteration process the fitness of the best individual improves over time and typically tends to stagnate.The process terminates when it coincides with successful discovery of the global optimum.

The governing equations for Particle Swarm Optimizer are as follows.

For each particle, calculate particle velocity according to the equations 5.1 and 5.2

$$Vi = Vi + (C1 \times rndi \times (pbsti - prsnti)) + (C2 \times rndi) \times (gbest - prsnti)) \quad \text{Eq(5.1)}$$

$$prsnti = prsnti + Vi \quad \text{Eq(5.2)}$$

Where, *Vi* is the i[th] particle velocity, *prsnti* is the current particle (solution),*rndi* is a random number between (0,1). *C1, C2* are learning factors. (Ranges between 1 to 4).

The characteristic function for the swarm is a distance function where, with momentary target being the leaders position itself, as presented in the equation 5.3.

$$r = \sqrt{(px - plx)^2 + (py - ply)^2} \quad \text{Eq(5.3)}$$

Where *px*and *py*are particle's x and y coordinates. *plx*and *ply* are the leader's position coordinates.

5.2 Modification of PSO for path planning

The main challenge in the work was in applying the PSO algorithm in the path planning domain. Certain important issues with regard to the general PSO algorithm in implementing over robots are presented further.

In PSO simulations the particles in the Swarm vary their velocities with respect to their distance from the *gbest*particle. Hence particles far away have greater velocities when compared to the ones which are near. But, in the case of swarm robots, the motion is limitedby the *Co-ordinate interval* parameter value along any of the eight directions possible limited by the *Detection co-ordinates value* as presented in table 1.

While applying PSO particles might converge at a point where collisions and occupancy of same coordinate are also possible whereas robots cannot occupy the same coordinate. So the algorithm is slightly modified toeliminate overlapping coordinates.

PSO particles are free to move over the given domain but in case of robots, they are constrained by the position of the obstacles. So, every next position assumed by the robot is checked for any presence of an obstacle with in the sensor range. Also, with a greater Weight Inertia value, particles in PSO oscillate for a certain period till they converge, which is not desired in case of robots, hence an optimum weight inertia value was selected.

Along with these general considerations, the behavior of the *gbest* (called LEADER here after) was different with respect to the general PSO algorithm. The algorithm relating the *gbest*behavior is described in the next sub section.

5.3 Leader path modification algorithm

In the PSO algorithm stated before, for *gbest*particle the next position always turns out to be in the same position or continues to move due to momentum. Now considering the scenario where the whole swarm starts from a single region. And hence the swarm at the starting point converges near the particle acting as*gbest*initially.

Hence for a leader, a special algorithm was designed. This algorithm makes the leader move towards the destination and the swarm following it by artificial potential fields. Logically, when the leader gets stuck between obstacles, due to the oscillatory and dispersive property of the swarm (with proper parameters) another member of the swarm assumes the leader position and its behavior is similar to the previous one. This fetches an alternative path when the swarm gets stuck.

The leader bot checks for the surrounding coordinates for higher power associated, which is obtained from the power matrix generated. The checking of the coordinates is limited to the range of the sensors. Coordinates having the highest power is marked and the increment of the x and y coordinates are made accordingly. This process also eliminates the leader from colliding with the obstacles, as they have a very low power value.

The power fringes formed are mostly circular. There comes a situation where the leader is stuck at a point with a common power value in all the directions or decrement in power in all the directions. Then, the bot is allowed to move randomly towards any of the eight coordinates possible.Pertaining to the rules of swarm intelligence, after defining the individual behavior of the particles in the swarm, emergence can be seen in path planning. The complete algorithm is presented as follows.

STEP 1: Set the checking coordinate (Cx,Cy) to the first surrounding coordinate. Set POWmax to 0 and (Px,Py)=(0,0).

STEP 2: If POW (Cx,Cy) >POWmax go to STEP 3, else go to STEP 4.

STEP 3: Assign POWmax = Power(Cx,Cy) and Px=Cx, Py=Cy.

STEP 4: Change (Cx,Cy) to the new coordinate in the clock wise direction. Repeat STEP 2 to 4 for all the 8 coordinates. Then go to STEP 5.

STEP 5: If POW(Px,Py)= 0, OR POW(Cx,Cy) has not better value, then randomly select a direction out of eight and assign relative coordinate to Px and Py. Go to STEP 6.

STEP 6: Place the G-best at (x+Px,y+Py) where (x, y) is the location.

The flow chart for the algorithm is presented in the figure 4.

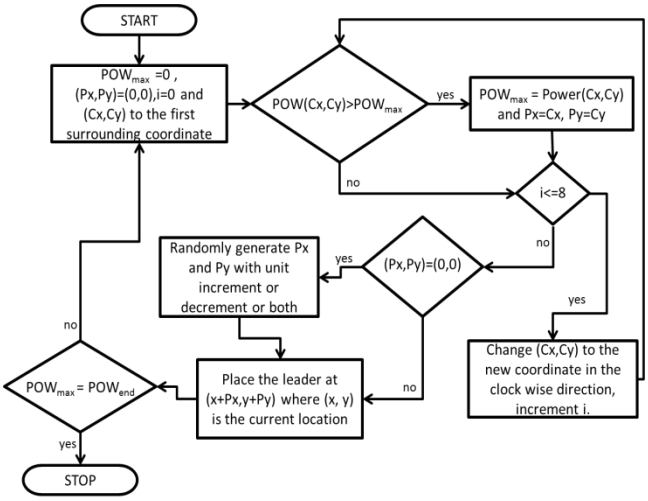

Figure 4: Flow chart for leader path

5.4 The combined path planning algorithm

Holding to the basic algorithm of Particle Swarm Optimisation, with a slight modification of the behavior of the leader, the final algorithm of path planning was arrived.

In summary, a leader of the swarm is the one receiving highest power in the group at its respective coordinates (position).The leader has the tendency to move in the direction of increasing power (in the power matrix linked to the power graph).The particles follow the leader following the PSO algorithm. With proper setting of the initial velocity and momentum parameters, a slight oscillatory behavior for the swarm is given by hit and trial. And when the leader is not facing any increment of power in any direction from its current position, it selects a random position.When the leader gets struck at any point in the arena, due to the slight oscillatory behaviorof the swarm, dispersion takes place around the leader which can give an alternate path, which reorients the direction of the swarm, thus giving a continuous path.

The final algorithm for path planning is as follows.

Step 1: Initialize a population of n (30). P=0.

Step 2: If (P !=n and P!=Pb (gbest particle number)). . If the power value is better than the previous personal best (pbestp), set current value as the new pbestp. If pbestp>gbestp set gbestp = pbestp andPb=P. Increment P.

Step 3: If P = Pb. The particle is the leader. The following steps are followed.

S.1: set the checking coordinate of the leader (Cx,Cy) to the first surrounding coordinate. Set POWmax to 0 and (Px,Py)=(0,0).

S.2: Check if the power value of the coordinate (Cx,Cy) in the power matrix is greater than POWmax. If it is greater, go to STEP S.3, else go to STEP S.4.

S.3: Assign POWmax = Power(Cx,Cy) and Px=Cx, Py=Cy.

S.4: Change (Cx,Cy) to the new coordinate in the clock wise direction. If all the coordinates are not checked, go to STEP S.2. Else go to STEP S.5.

S.5: If (Px,Py)are zeroes, then randomly select a direction out of eight and assign relative coordinate to Px and Py. Go to STEP S.6.

S.6: Set the leader coordinate to (x+Px,y+Py) where (x, y) is the current position in the 2D plane.

Step 4: Calculate particle velocity according to the equations 5.4 and 5.5

$$Vi = Vi + (C1 \times rndi \times (pbsti - prsnti)) + (C2 \times rndi) \times (gbest - prsnti)) \quad \text{Eq(5.4)}$$

$$prsnti = prsnti + Vi \quad \text{Eq(5.5)}$$

Step 5: The following steps are followed to get the direction of heading for the particles.

S.H.1: Set the direction of heading (Hx,Hy) to the first surrounding matrix coordinate. Set POWHmin to 0 and (Hx,Hy)=(0,0).

S.H.2: Subtract power at (Hx,Hy) from nextpi. If pow<POWHmin, goto step S.H.3 else change (Hx,Hy) to new surrounding position and goto S.H.2. Also check if the heading direction coordinate has any obstacle or another robot. If all the surrounding positions are checked goto step 6.

S.H.3: set POWHmin = pow and (xp,yp)=(Hx,Hy). Change (Hx,Hy) to new surrounding position and goto S.H.2.

Step 6: Increment P, if P= n go to step 7.Else, goto Step 2.

Step 7: Plot all the particles in their new positions. Give a small delay. Check if the leader is in the 5% vicinity of the endpoint. If present terminate. Else set P=0 and gbestp=0 and goto step2.

The flow chart for the algorithm is presented in figure 5.

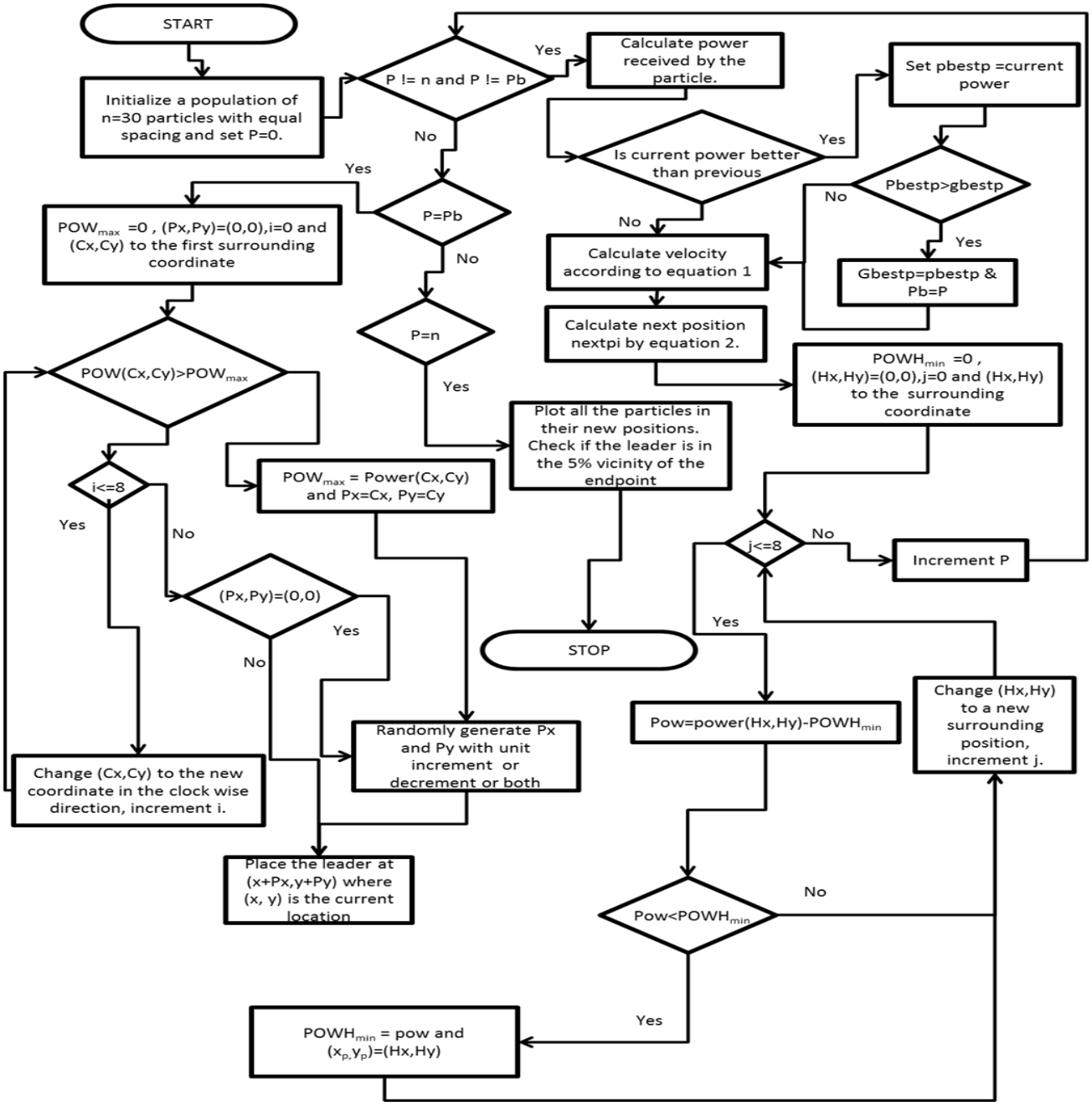

Figure 5: Flow chart for combined path planning algorithm

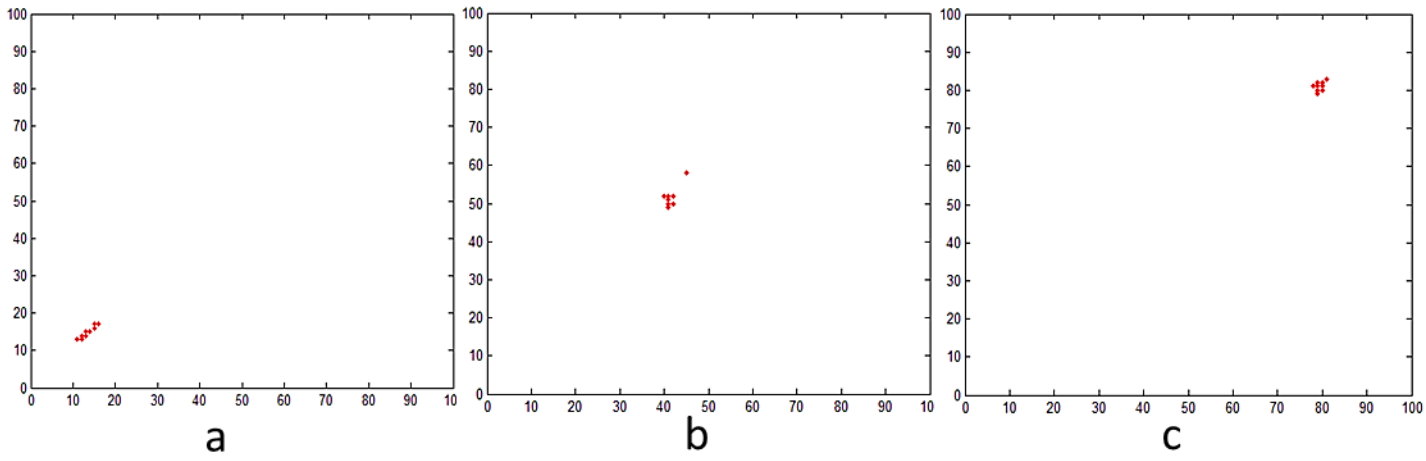

Figure 6: Screen shots of swarm animation finding path in a configuration space. The obstacles are removed in while animating to speed up the iterations.

6. RESULTS

The proposed algorithm was executed on the developed simulator. The simulations of the swarm moving in the configuration space are presented in the figure 6. After a series of iterations were carried out, the final path is generated by connecting the path traversed by the *gbest* of the swarm.The calculated solution for the configuration space or the optimal pathis presented by a tail (tracks) of animation as shown in the figure 7.

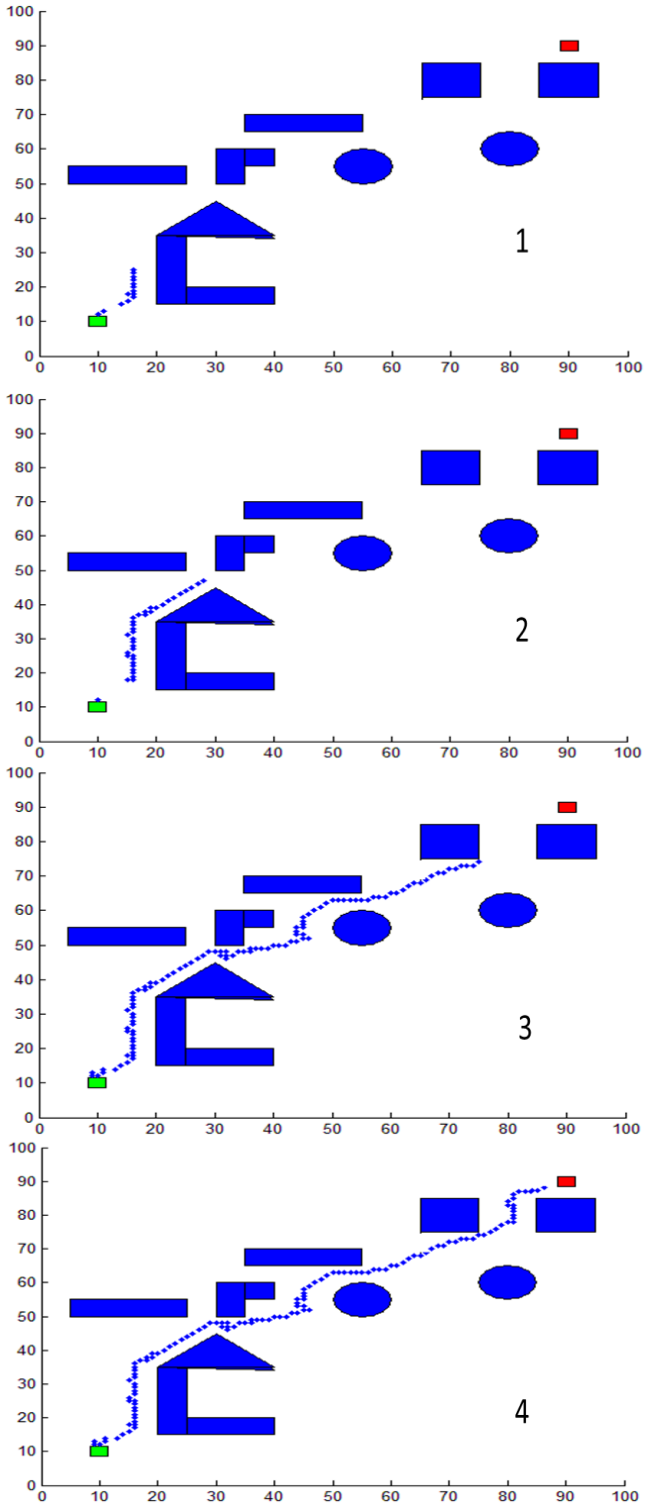

Figure 7: Screenshots showing animation of the final path connecting starting and ending points.

Another sample configuration space simulated with the proposed algorithm is shown in the figures 8(a-d) with configuration space, obstacle matrix, power matrix and the final solution in sequence.

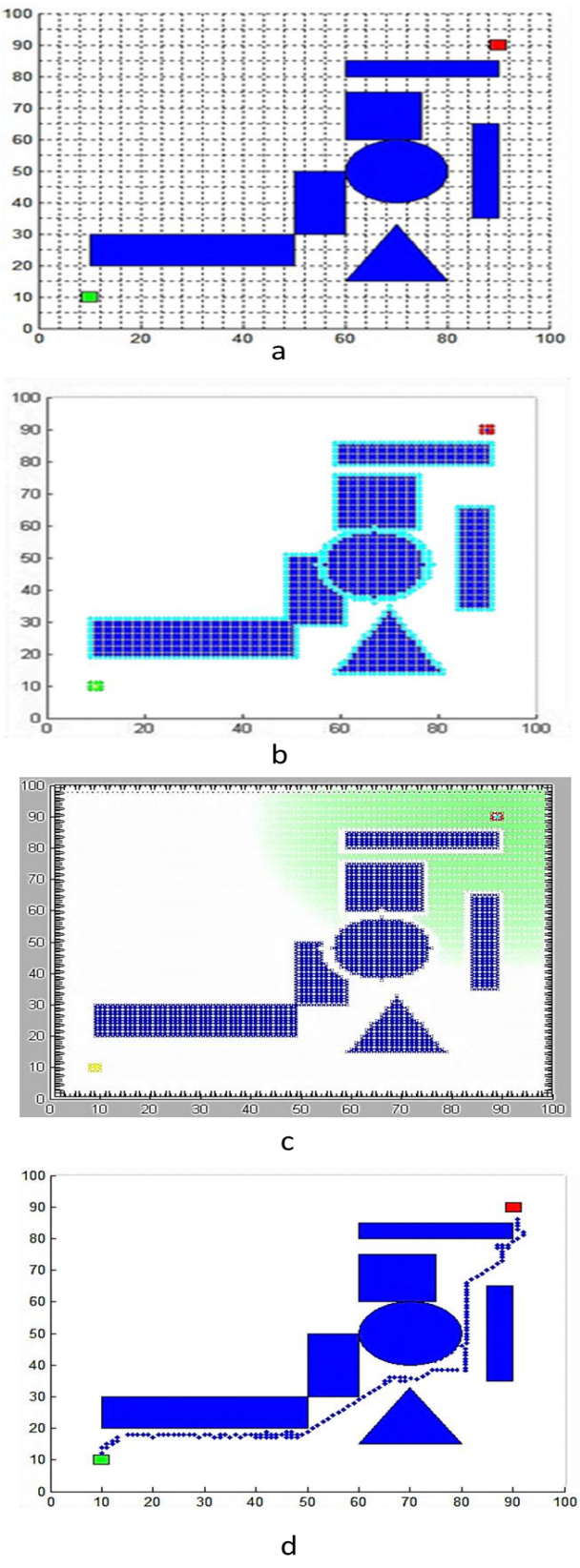

Figure 8: a) Sample configuration space b) Obstacle matrix c) Power matrix d) Path planning solution.

7. CONCLUSIONS

The developed simulator and algorithm provided satisfactory results for various configuration spaces. We were able to achieve relatively better solutions by carefully selecting the PSO parameters and it was also seen that a minimal variation in the parameters did not actually bring about any considerable difference or improvement. The proposed algorithm generated optimal paths for various configuration space and the approach in designing the simulation environments for local path planning are encouraging in development of newer algorithms in path planning.

But, an issuewas encountered during the testing process. It is apt to present it for future analysis and development of the simulator and the algorithm.Considering the power source as light, the light gradient is also dependent on the obstacles, where laws of optics come into picture. This was not completely considered which sometimes gives vague or Np incomplete solution for deep corners formed by the group of obstacles, because the swarm gets struck in the cavity like points.This can be overcome by re programming the power matrix generator following the nature of the gradient of the power source (this might vary for a heat source).

Future works with this simulator are planned in developing algorithms like ants colony optimisation, simulated annealing, genetic algorithm for path planning problem. Further extension to 3D spaces is planned.

8. ACKNOWLEDGMENTS

Our thanks to Dr. S.K.Saha and Dr.B.K.Panigrahi of IIT Delhi for their guidance in developing the simulator and Dr. S.MGiriraj Kumar and Dr. S. Pugazhenthi of SASTRA University for extending their support in analysis of various algorithms related to swarm intelligence and robotics. We also thank SASTRA University for providing the required facilities in carrying out this project.

9. REFERENCES

[1] A. Colorni, M. Dorigo et V. Maniezzo, Distributed Optimisation by Ant Colonies, actes de la première conférenceeuropéennesur la vie artificielle, Paris, France, Elsevier Publishing, 134-142, 1991.

[2] Swagatam Das, ArjitBiswas, SambartaDasgupta, and Ajith Abraham, Bacterial Foraging Optimisation Algorithm: Theoretical Foundations, Analysis and Applications, IEEE Transactions on SMC, Part A, Vol. 39, Is- 3, 670 – 679, 2009.

[3] Sanjay Sarma O V,Dr.S.M.GiriRajkumar, Dr.K.Ramkumar. Real time application of Ants Colony Optimisation.International Journal of Computer Applications 3(8):1–6, June 2010. Foundation of Computer Science.

[4] Dr. S.M. GiriRajkumar, Dr K. Ramkumar, BodlaRakesh, Sanjay Sarma O V and Deepak Jayaraj. Real Time Interfacing of a Transducer with a Non-Linear process using Simulated Annealing. Sensors and Transducers. Vol. 121, Issue 10, October 2010, pp.56-67.Sensors Portal.

[5] M. Anthony Lewis and George A. Bekey (1992), The Behavioral Self-Organization of Nanorobots Using Local Rules. Proc. of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems, Raleigh, vol. II pp. 1333-1338, 1992.

[6] M. Dorigo and R. Groß . Cooperative transport of objects of different shapes and sizes. Ant Colony Optimisation and Swarm Intelligence – Proceedings of ANTS 2004 – Fourth International Workshop, volume 3172.

[7] ErolS¸ahin, Swarm Robotics: From Sources of Inspiration to Domains of Application, KOVAN – Dept. of Computer Eng., Middle East Technical University, Ankara, 06531, Turkey.

[8] FengWeiXing, Wang KeJun, Ye XiuFen and GuoShuXiang, Novel Algorithms for Coordination of Underwater Swarm Robotics- Proceedings of the 2006 IEEE – International Conference on Mechatronics and Automation.

[9] Kennedy, J.; Eberhart, R. (1995). "Particle Swarm Optimisation".Proceedings of IEEE International Conference on Neural Networks. IV. pp. 1942–1948.

[10] O.Hachour. Path Planning of Autonomous Mobile robot. International Journal of Systems Applications, Engineering & Development, pp. 178-190, Issue 4, Volume 2, 2008.

[11] P. Raja and S. Pugazhenthi, Path planning for a mobile robot in dynamic environments, International Journal of the Physical Sciences Vol. 6(20), pp. 4721-4731, 23 September, 2011.

[12] Qiang Zhao and Shaoze Yan,' Collision-Free Path Planning for Mobile Robots Using Chaotic Particle Swarm Optimisation', ADVANCES IN NATURAL COMPUTATION,Lecture Notes in Computer Science, 2005, Volume 3612/2005, 632-635, DOI: 10.1007/11539902_77.

[13] Kaiyou Lei; YuhuiQiu;Yi He, A novel path planning for mobile robots using modified particle swarm optimizer'. Systems and Control in Aerospace and Astronautics, 2006.

[14] SedighehAhmadzadeh and Mehdi Ghanavati, Navigation of mobile robot using the PSO particle swarm optimisation, Journal of Academic and Applied Studies (JAAS), Vol. 2(1) Jan 2012, pp. 32-38.

[15] Robot Motion Planning, Jean-Claude Latombe, 1991, Kluwer Academic Publishers.

[16] Othman W.A.F.W., Amavasai,B.P., McKibbin, S.P., Caparrelli, F. and Travis, J.R. An Analysis of Collective Movement Models for Robotic Swarms. In Proceedings of The International Conference on Computer as a Tool (EUROCON 2007), Poland, 2007, pp. 2373-2380