INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 5, Issue 3)
Available online at: www.ijariit.com

Single robot path planning using nature inspired algorithm in a static environment

Agrim Mehra
b416005@iiit-bh.ac.in
International Institute of Information Technology, Bhubaneswar, Odisha

Prakash Kumar Singh
b416033@iiit-bh.ac.in
International Institute of Information Technology, Bhubaneswar, Odisha

Ankur Kumar
b316033@iiit-bh.ac.in
International Institute of Information Technology, Bhubaneswar, Odisha

Deepak Kumar Rastogi
b416059@iiit-bh.ac.in
International Institute of Information Technology, Bhubaneswar, Odisha

ABSTRACT

This paper proposes a solution to the problem of Path Planning for a robot in a static environment consisting of obstacles using Nature inspired Algorithm. Here, in this case, we have applied a very efficient search Algorithm known as Particle Swarm Optimization Algorithm. The main focus is to minimize the length of the path taken by a robot from the initial position to target position and at the same time avoiding any obstacles that occur in the path of the robot. Particle Swarm Optimization uses different techniques of searching for the most Cost efficient path by the use of random simulations of different possible paths and comparing their costs. Simulations are achieved in MATLAB and are further demonstrated in this paper.

Keywords— *Robot Path Planning, PSO, Path planning using PSO*

1. INTRODUCTION

Robot path planning is a very complex topic in the case of an unknown environment as well as Known environment. Methods like Potential Fields, Cell Decomposition, Roadmaps and Mathematical Programming are most used techniques in path planning. However, they are not feasible for a large environment setting. So to overcome the problems occurring with these classical techniques. We use Heuristic and Meta-Heuristic approaches such as Genetic Algorithms, Gravitational Search Algorithm and other nature-inspired Algorithms.

In this Paper, Particle Swarm Optimization Algorithm is implemented. It is inspired by the search method of a Swarm of birds when looking for food or any other particular Object.

Here, the Particle Swarm Optimization technique is used in order to derive the most Low-cost path for the robot between two points in a static environment. Simulations are made for a robot which is capable of avoiding obstacles and following the intended path. To achieve the intended results new functions are added to the basic Particle Swarm algorithm to make it applicable to path planning.

2. PARTICLE SWARM OPTIMIZATION ALGORITHM

Particle Swarm Optimization technique is a very simple yet Powerful technique used in order to achieve the desired results for any Objective Function. It draws inspiration by the techniques used by Swarm of Birds in order to find the optimal Solution. It divides the solution into a number of particles and then uses their personal best values in order to achieve the desired results.
The mathematical formula for Particle Swarm Algorithm is given below:

$$\vec{X}^t{}_i + 1 = \vec{X}^t{}_i + \vec{V}^t{}_i \quad (1)$$

$$\vec{V}^t{}_i + 1 = w\vec{V}^t{}_i + c_1 r_1(\vec{P}^t{}_i - \vec{X}^t{}_i) + c_2 r_2(\vec{G}^t - \vec{X}^t{}_i) \quad (2)$$

Where, i = Particle Number, t= Iteration counter, w= Inertial weight, c_1 and c_2 are Acceleration coefficients known as cognitive and social parameters, $r_1$and r_2= Random values in [0,1], $P^t{}_i$ = Best position of particle for iteration t, G^t = Global best position for all the particles.

Here First equation updates the position for the next search Iteration.

The velocity of a particle at the next iteration=$V^t{}_i$ + 1, Position of para ticle at the present iteration =$X^t{}_i$, Position of the particle at the next iteration = $X^t{}_i$ + 1.

So Position for the current Iteration in addition to the Velocity for the next Iteration gives the value of Position for the start of Next Iteration.

Here the $Inertial\ Component = (w^* V^t{}_i)$ gives the Current Direction to the particle. $Cognitive\ Component = c_1 r_1 (P^t{}_i - X^t{}_i)$ is known as Cognitive Component. It prompts the particles to move towards the Personal Best Solution.

$Social\ Component = c_2 r_2 (G^t - X^t{}_i)$ is known as Social Component. It prompts movement of particle towards Global Best Solution.

Using the above components, we execute Algorithm in any environment using the basic PSO algorithm.

Algorithm 1 Pseudo Code of PSO

Initialize the controlling parameters (N, c_1, c_2, W_{min},W_{max}, V_{max}, and MaxIter)
Initialize the population of N particles.
while the end condition is not satisfied do for each particle
  for each particle
    Calculate the objective of the particle
    Update PBEST if required
    Update GBEST if required
  end for
  Update the inertia weight
  for each particle
    Update the Velocity (V)
    Update the Position (X)
  end for
end

First, we initialize the controlling parameters such as,

(a) Number of Particles (Population of the Swarm)
(b) Value of Acceleration Coefficients.
(c) Upper and lower bound of Inertial Weight.
(d) Value for the Upper and Lower bound of the Velocity Vector.
(e) Number of Iterations required to achieve the Optimal Solution.

For each particle, we first initialize the Value of Personal best Solution and Global Best solution as Infinity. Then we calculate the Objective value for each particle from the Objective Solution. Hence we update the value of P.Best (Personal Best) and G.Best (Global Best) if the Objective value of the function is less than the Value of Current Value of P.Best and G.Best. At the End of all the Iterations, the Final G.Best value is concluded which denotes the optimal solution.

3. PARTICLE SWARM OPTIMIZATION IN ROBOT PATH PLANNING

In Robot Path Planning PSO is applied in order to find the next position for the robot. Here the search arena corresponds to a Search Landscape for the robot. Each position of the Particle (Robot in this case) is calculated using the three Components: Inertial factor, Cognitive Component and Social Component.

Following steps are required to make an effective Robot Path Planning Environment using PSO.

3.1 Environment Modelling

The first task for the Robot Path Planning is to create a virtual Path in correspondence to any real path in which we wish to practically demonstrate Robot Path Planning. This can be achieved in MATLAB. Steps to create a Virtual Arena for Robot Path Planning.

(a) First, the Initial Starting point of the robot is initialized along with the final Destination point.
(b) Obstacles are then formulated. In this case, we have initialized Round Obstacles in between the Initial and Final Point.
(c) Minimum and Maximum extent of the Arena is given in order to Limit the scope of Robot Violation from Path.
(d) Values are then stored in variables for further use in other Main PSO Function.

4. OBJECTIVE FUNCTION

The Main Objective is to make the robot travel in a straight line from the given Starting Point to the Final Point and also avoiding the obstacles in the path. Here we use the Euclidean Distance between the Robot's Initial Point and the Destination.
Suppose the two Points are given by:
Initial Point: $\{A_1, \boldsymbol{B_1}\}$ and Final Point: $\{A_2, \boldsymbol{B_2}\}$

$$Euclidean\ Distance\ = \sqrt{(A_2 - A_1)^2\ +\ (B_2 - B_1)^2}$$

For n points this Euclidean Distance can be summarized as:

$$Distance\ = \sqrt{\sum_{i=1}^{n}(A_i\ -\ B_i)}$$

Where: A denotes set of points of Initial Position, B: denotes set of points for the Final Position.
Here the concept of waypoints is used in order to cover the distance from the Initial Position to Final Position.
A number of Perpendicular Lines needed in XY grid calculated by using these Waypoints.
Let, n= Number of waypoints.
D= Distance between Initial Point and Final Point.
Number of Perpendicular Lines= D/n+1.

Then the PSO algorithm performs random sampling on these grid lines generated between the start and goal positions using the Waypoint in between. Each point is compared with the goal point and the point having minimum distance with the goal point is selected as the feasible waypoint. Points lying inside obstacles are considered as invalid points which are unfit for being deemed as Waypoint.

Now the path is divided into several Feasible Waypoints in between.
Path of the Robot is given as:

$$P\ =\ \{SP, W_1, W_2, W_3, \ldots W_N, FP\}$$

Where SP is the Starting Point, FP is the Final Point, $W_1, W_2, W_3, \ldots W_N$ are the Number of waypoints.

5. DEALING WITH VIOLATION OF ARENA/VELOCITY

It is quite important to make sure that the Robot stays within the bounds of the given arena. So the concept of Violation is being introduced to make the robot stay within Position bounds.

If Velocity of a particle goes beyond the allowed Velocity value then the Velocity of the robot at that position is replaced by the maximum value of Velocity possible at that point. Velocity Vector is divided into X and Y grid. PSO is applied separately to both the X and Y grid.

Violation in the Position in both X and Y grid is calculated and indicated in the Final result.

Violation with respect to obstacles is also calculated.

Here new term β is introduced in order to define Violation.

Here, β comes in the play when calculating the total cost of Path taken by Robot.

$$T_c\ =\ E_d\ +\ \beta^{*}V$$

Where T_c is Total Cost (Distance in this case), Ed is Euclidean distance, V is Violation.

Violation is calculated by maximizing the distance between the Robot and Centre points of the Obstacles.
Let Centre of the Obstacle be (X1; Y1) with radius = R1.
Let D be any variable, D= Euclidean distance between Robot and centre points.

Now, A variable E = 1 - d/R.
If E>0 then this is added into the Violation for that Robot.

Similarly, values for the Violation is added for all the robots.

6. PATH SIMULATION AND DISTANCE COST

In the Plot (Figure 1: Path Simulation) given below the starting point is taken as (0,0) and the final points for the robot are taken as (8,6) while four circular obstacles are in the way of the Robot. Here, three waypoints denoted by Red points are taken into consideration. The four obstacles are having their centres at points:(3,4),(2,1),(4.5,7.5) and (6.5,3.5).

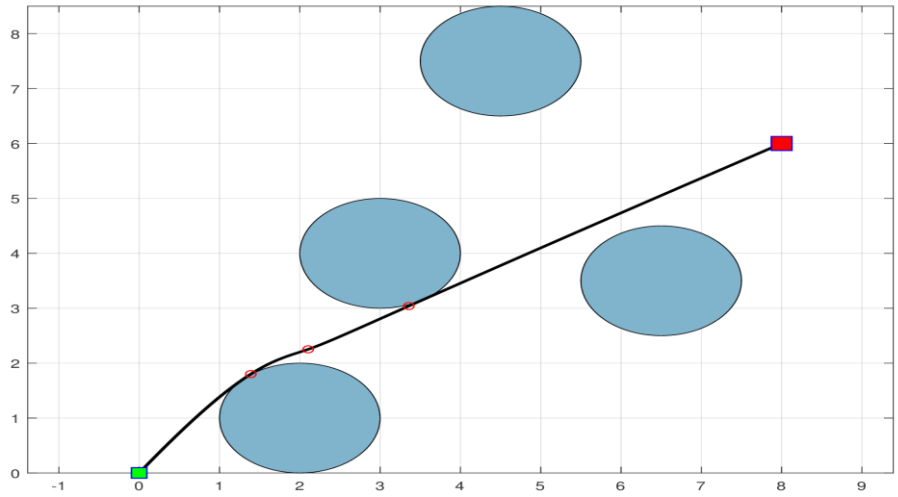

Fig. 1: Path Simulation

Here, the value of Iteration(t) is taken as 500. In (**Figure 2**: Calculated Distance Cost) given below we can observe that the value of Minimum Distance approaching 10 at the End of Iterations which also is approximately equal to the Euclidean distance between (0,0) and (8,6). Thus, proving the accuracy of our Robot Path Planning Model based on PSO.

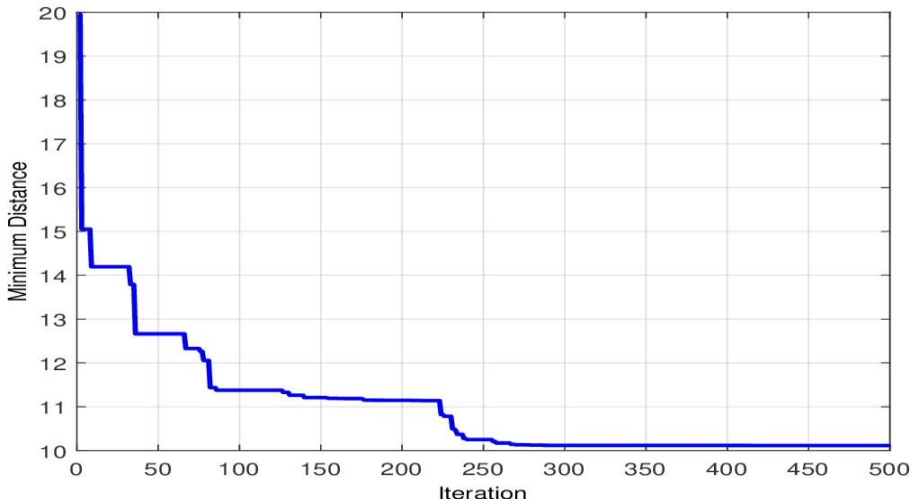

Fig. 2: Calculated Distance Cost

7. CONCLUSION

This paper introduces a way of robot path planning consisting of obstacles in a static environment using nature-inspired algorithm. Here Particle Swarm Optimization Algorithm is used to minimize the length of the path taken by a robot from an initial position to the final position without any collision or going out of the arena. The cost function is the combination of the Euclidean distance and Violation. Particle Swarm Optimization Algorithm minimizes the cost and finds the most cost-efficient path for the robot.

8. REFERENCES

[1] M. Shahab Alam, M. Usman Rafique, and M. Umer Khan, International Journal of Computer Science and Electronics Engineering (IJCSEE) Volume 3, Issue 3 (2015) ISSN 2320–4028 (Online).

[2] Amin Zargar Nasrollahy, Hamid Haj Seyyed Javadi, Using Particle Swarm Optimization for Robot Path Planning in Dynamic Environments with Moving Obstacles and Target,2009 Third UKSim European Symposium on Computer Modeling and Simulation.

[3] Mihir Kanta Rath, B.B.V.L. Deepak, PSO based system architecture for Path Planning of Mobile Robot in Dynamic Environment, Proceedings of 2015 Global Conference on Communication Technologies(GCCT 2015).

[4] Masoud Dadgar, Shahram Jafari, Shiraz University, Shiraz, Iran, A PSO-based multi-robot cooperation method for target searching in unknown environments.

