International Journal of Advanced Mechanical Engineering.
ISSN 2250-3234 Volume 8, Number 1 (2018), pp. 275-284

http://www.ripublication.com

An Algorithm for Online Path Planning of Mobile-Robot based on Displacement Vector Method

Sourav Kumar Sahu[1*], Sumanta Panda[2] and Ajit Kumar Pattanaik[3]

[1,3] Government College of Engineering Kalahandi, Bhawanipatna, India.
[2]Veer Surendra Sai University of Technology Burla, India.
E-mail: sourav7917@gmail.com, sumanta.panda@gmail.com, ajitpuce@gmail.com

Abstract

While navigating towards the target, a reactive mobile robot plans autonomously its local path in real time based on the immediate surrounding information sensed. In the present work a new local path planning algorithm is proposed based on the displacement vector method. Required displacement vector is defined connecting current position and target of robot. Then N number of displacement vector are defined within the sensing range of the robot. Mobile robot selects that displacement vector for the movement which is having highest value of component along required displacement vector. This algorithm was tested for various environments consisting of convex obstacles and the robot generated satisfactory path. The result of the present algorithm is compared with previous works and it was found that path length produced by this algorithm is less. The path length produced and time taken by the algorithm for various value of N and sensing range was recorded and it was found that optimum path is generated for N=36 and sensing range=30 .

Keywords: Path planning algorithm, Displacement vector, Convex obstacle, Sensing range

I. Introduction

Nowadays robot are being widely used in various fields like manufacturing industries, security purpose, medical surgery, military application, space research, nuclear power plant etc. So locomotion of robot is an important issue in field of developing fully autonomous robot. For this the robot should have the capability of defining and following their own path of movement as per their requirement. The path planning problem consist of a mobile robot which needs to move from its start position to goal position in a work space by avoiding collision with obstacle in such a way that length of path followed by the robot and time taken to reach at goal is minimum. Since last three decades development of autonomous mobile robot has been a challenging task for the researchers. Hence many researchers have shown interest on the development

of autonomous mobile robot. BBVL Deepak, Dayal R. Parhi [1] et.al. have proposed a PSO based path planning algorithm for the mobile robot navigation task. In this paper they have modelled a new fitness function which satisfy the obstacle avoidance and optimal path traversal conditions. From the obtained fitness value of each particle in the swarm, the robot moves towards the particle which is having optimal fitness value. Rahul Kala, AnupamShukla, RituTiwari [2]et. al. have presented an online path planning algorithm called multi neuron heuristic search(MNHS) algorithm which is a modified A^* algorithm. They have implemented the algorithm in a hierarchical manner, where each generation of the algorithm gives a more detailed path that has a higher reaching probability. Pratap Kumar Panigrahi, Saradindu Ghosh, Dayal R Parhi[3] et.al. have proposed a path planning and intelligent control of mobile robot in unknown environment with static/dynamic obstacles and fixed target. A radial basis function (RBF) network approach is proposed in this work for obtaining optimized path to reach the goal without collision. The competency of the proposed approach is verified by means of simulation results using MATLAB where robot moves in a variety of environments with obstacles of different shapes and sizes.

II. Modeling

Modeling of robot path planning problem includes mathematical formulation of work space, configuration space formulation, sensing obstacle, navigating robot in workspace.

Mathematical formulation of work space:

It involves both defining size of the work space and position of the obstacle in the workspace .In Euclidian space size of workspace can be specified by giving limit to axes of the world frame F_W. An obstacle in work space may be of two types those are Convex Obstacle and Non-convex Obstacle

Convex Obstacle:

A subset $B \subset R^N$ is called a convex obstacle if and only if for any pair of points in B all points along the line segment that connects them are contained in B.

If $X_1, X_2 \in B$ then for $\lambda \in (0,1) \lambda X_1 + (1-\lambda) X_2 \epsilon B$

Non-convex Obstacle:

A polygon or region which fails to satisfy convex criteria is called Non-convex Obstacle.

Configuration space formulation:

In configuration space formulation data about configuration space is calculated by knowing data about workspace. The shape of the C-obstacle changes with change in orientation of the robot. Consider the case when Robot Axis is Parallel to the world axis. The corresponding C-obstacle is the locus of all the point passed by the origin of robot frame as the robot move around the obstacles in same orientation with some of its part just touching the obstacle shown in figures 1 given below. Now Consider the

case where Robot axis is Inclined at certain angle to the world axis. The C-obstacle for this orientation of the robot is shown in figure 2 given below.

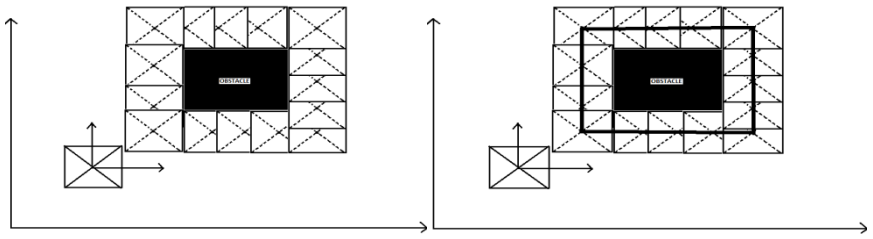

Figure 1: C-obstacle for the orientation when robot axes is parallel to world axes

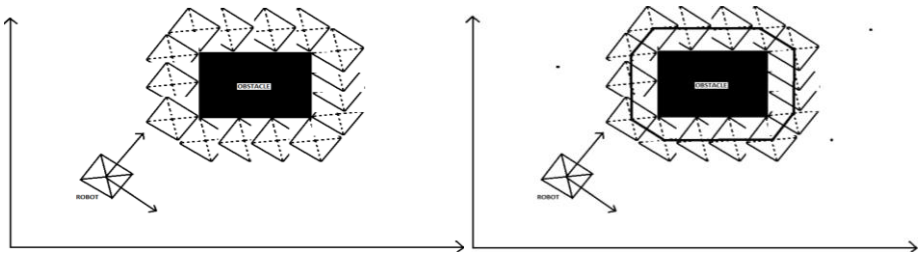

Figure 2: C-obstacle for the orientation when robot axes is inclined to the world axes

So to ensure collision free path it will be safe to increase dimension of obstacle in all direction by the amount equal to distance between origin of robot frame and the farthest point from origin of robot frame to find C-obstacle. However this will ignore those path where path is possible for some particular orientation of robot only. The figure 3 given below depicts the conversion of work space into configuration space.

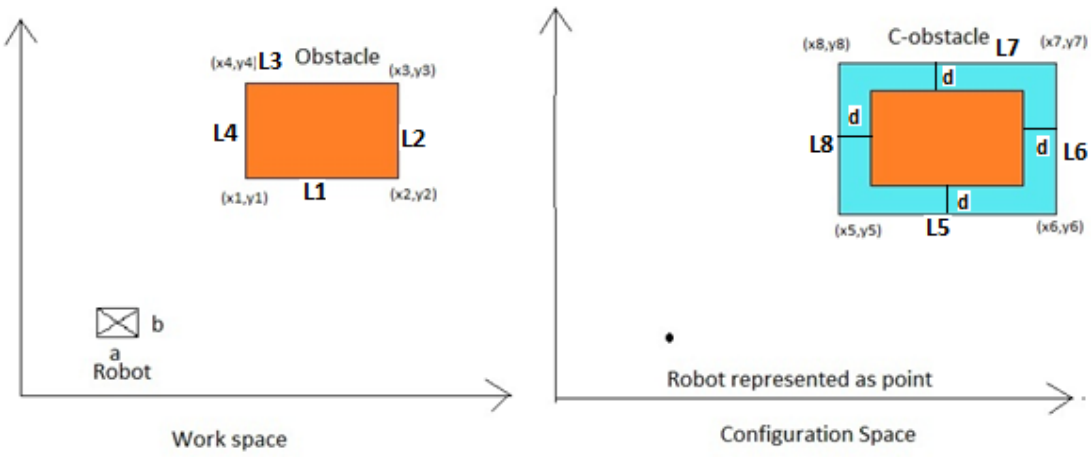

Figure 3: Conversion of work space into configuration space

Sensing obstacle:

Mathematically sensing obstacle means to find out whether a particular point is falling upon obstacle or not. This can be done by dividing the obstacle into number of primitive and checking for whether the point is satisfying each primitive or not.

Sensing obstacle is done by defining a logical predicate that serve as collision detector.

A predicate is a Boolean value function .

Let $\varphi: w \rightarrow \{ true(1), False(0)\}$

Φ returns true for a point (x,y) in W that lies in O and false otherwise. Where O represents obstacle region. Suppose there are 'n' number of obstacles in world then Φ can be represented as the result of Boolean operations as follows.

$\Phi(x,y) = \alpha_1(x,y) \vee \alpha_2(x,y) \vee \alpha_3(x,y) \vee \dots\dots\dots\dots\dots\dots \vee \alpha_n(x,y)$

Where $\alpha_1(x,y)$=1 when point (x,y) falls upon obstacle 1 otherwise $\alpha_1(x,y)$=0.

Similarly $\alpha_n(x,y)$=1 when point (x,y) falls upon obstacle n otherwise $\alpha_n(x,y)$=0.

Suppose there are 'm' number of primitive to constitute obstacle 'n'. So $\alpha_n(x,y)$ can be represented as result of Boolean operation as follows.

$\alpha_n(x,y) = \alpha_{n1}(x,y) \wedge \alpha_{n2}(x,y) \wedge \alpha_{n3}(x,y) \wedge \dots\dots\dots\dots\dots\dots\dots\dots \wedge \alpha_{nm}(x,y)$

Where $\alpha_{n1}(x,y)$=1 when point (x,y) satisfy primitive 1 of obstacle n otherwise $\alpha_{n1}(x,y)$=0.

Similarly $\alpha_{nm}(x,y) = 1$ when point (x,y) satisfy primitive m of obstacle n otherwise $\alpha_{nm}(x,y) = 0$.

Robot Navigation algorithm: For navigating robot we give start position (Ps), goal position (Pg),number of search direction (n) and sensing range (r) as input. Initially we take start position as current position. Then we define a require displacement vector as the displacement vector from current position to goal position. Then we define a circle having radius equal to sensing range and center at current position. Then we locate 'n' number of points on the circumference of the circle dividing the circumference into 'n' parts. Here each point on the circle represent a possible

direction of movement. Now for each point on the circle a displacement vector connecting current position to the point, is defined. Now we have a list of 'n' number of possible displacement vector. Now we calculate component of each displacement vector along the direction of require displacement vector. Those displacement vectors whose final point falls on obstacle or any point on the vector falls on obstacle is deleted from the list of possible displacement vector. Now we select that displacement vector as the movement of robot which is having highest value of component along require displacement vector. This procedure is continued until the robot reaches goal position. The figure 4 given below depicts the entire process.

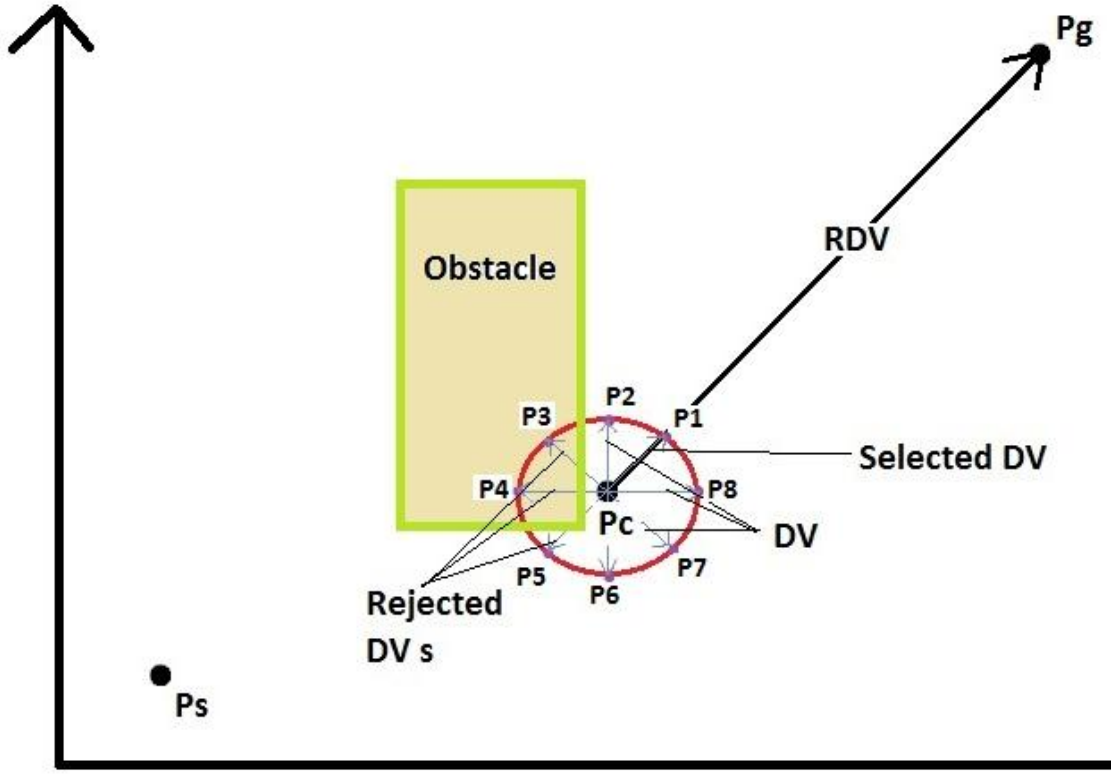

Figure 4: Selection of displacement vector for navigation.

$x_1 = x_g - x_c \qquad x_2 = x_k - x_c$

$y_1 = y_g - y_c \qquad y_2 = y_k - y_c$ (where K= 1,2,3,4,5,6,7,8)

Component of DV along RDV = $\frac{x_1x_2+y_1y_2}{\sqrt{(x_1{}^2+y_1{}^2)}}$

III. Results and Analysis

The proposed algorithm is simulated by using MATLAB in Intel® Core™ i5-2400 CPU @3.10 GHz processor with 2 GB RAM for different environments and the path length produced by the algorithm is found to be better than previously proposed algorithms. It can be noticed that the path length produced for start at (30,20) and goal at (320,320) is 447.34 which is very less as compared to path generated by PSO based path planning algorithms.

Table 1. Tabulation for the result obtained by present algorithm for the environment used by Dayal.R.Parhi et.al.[1]

Sl	Start position	Goal	Shortest	Path length	Time
1	(30,20)	(320,320)	417.25	454.96	36.89
2	(30,20)	(320,320)	417.25	447.34	22.35
3	(36.69,269.66)	(315.72,55.77)	351.57	407.86	33.89
4	(157.66,43.49)	(206.04,69.07)	54.73	68.34	5.14
5	(19.75,123.31)	(322.98,269.66)	336.69	370.09	30.71
6	(278.62,314.80)	(168.95,137.64)	212.63	244.03	18.76

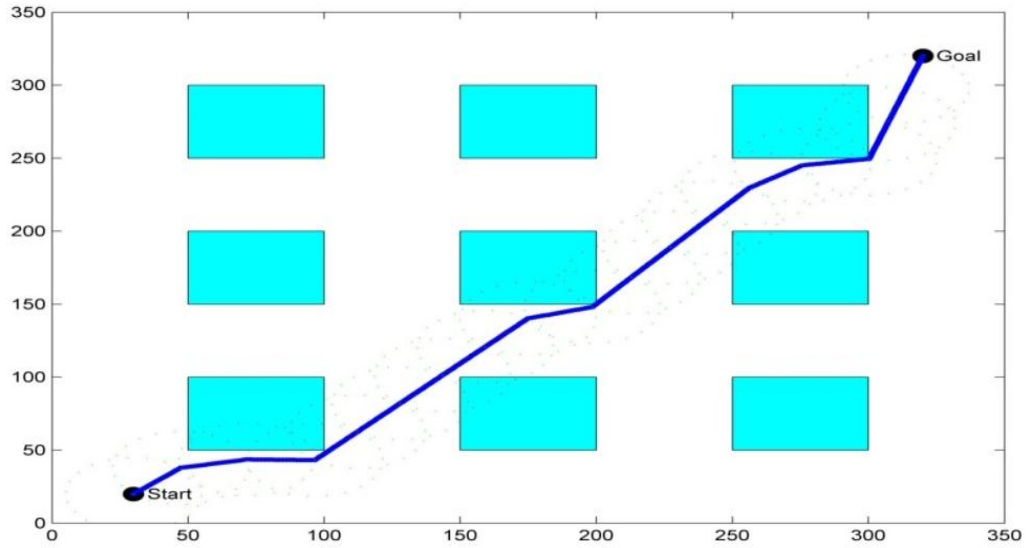

Figure 5: Figure showing Optimum Path generated

For testing the algorithm a new environment was designed and the proposed algorithm was run for 70 numbers of trials. From these trials the shortest distance obtained was 683.8738 m at N=36 and R=30. The shortest time taken was 143.0625 at N=12 and R=30.

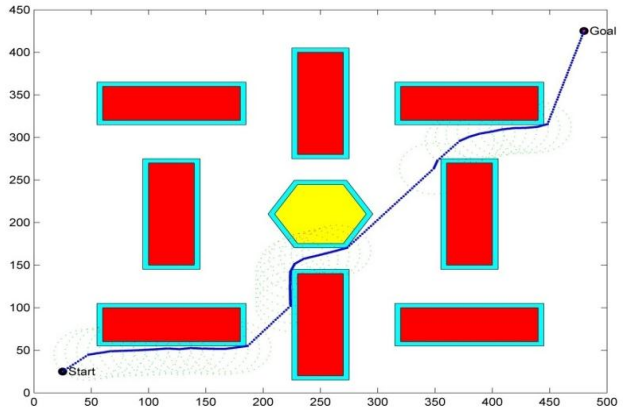

Figure7:Path generated for the case of minimum time taken

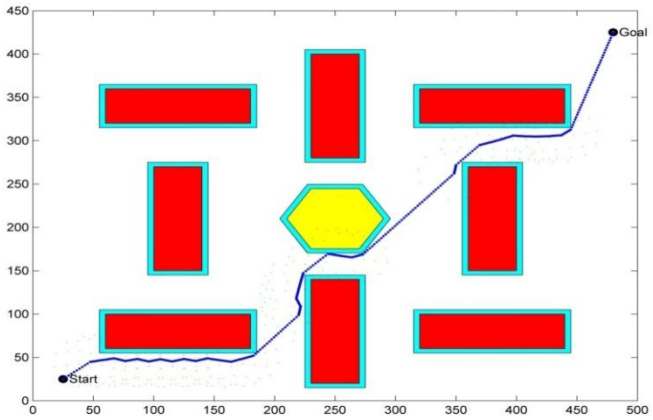

Figure 6:Path generated for the case of minimum path length

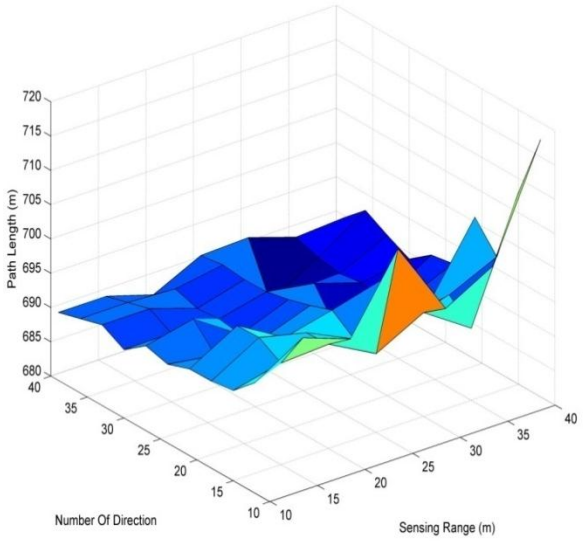

Figure 8: Plot showing variation of path length with N and R

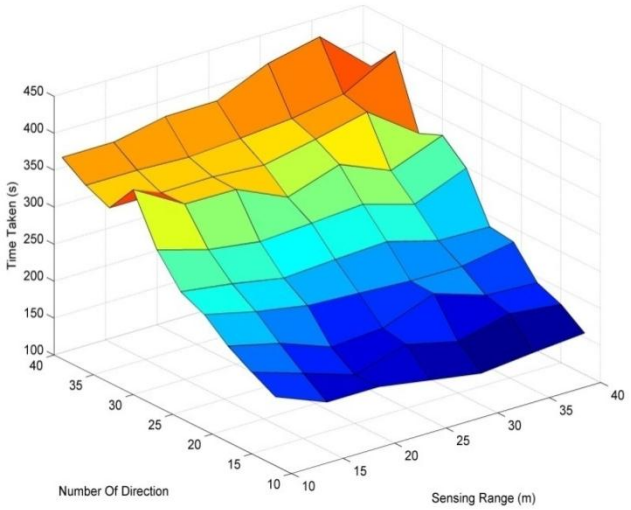

Figure 9: Plot showing variation of time taken with N and R

From the surface plot drawn for path length it can be observed that greater path length is produced at less value of number of direction of motion and high value of sensing range represented by peaks on the surface plot. Furthermore it can be observed that for a particular value of sensing range, path length decreases with increase in number of direction. Further it can be observed that for a low value of number of direction of motion path length produced varies greatly with change in sensing range but for high value of number of direction path length produced does not varies much with change with number of direction.

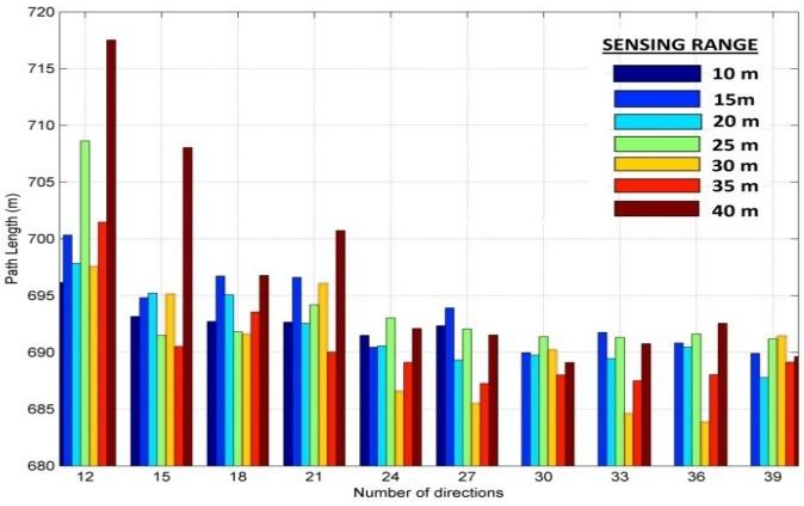

Figure 10: Path length variation with N for different R

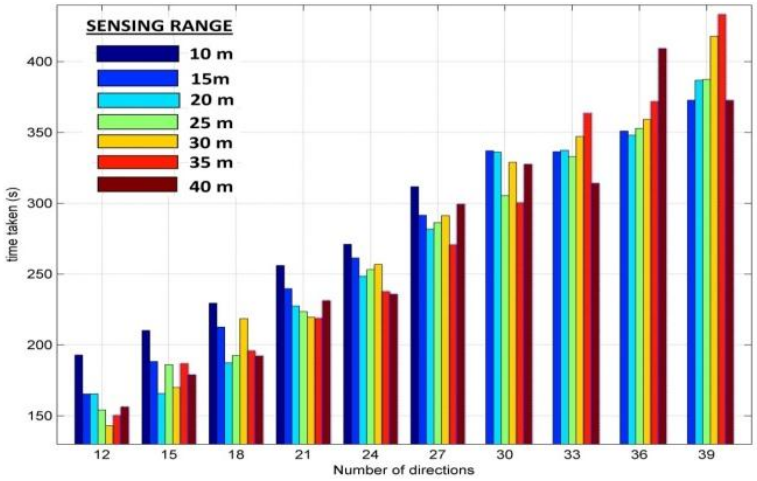

Figure 11: Time variation with N for different R

From the bar graph plotted for path length produced it can be observed that when number of direction of motion is small path length increases with increase in sensing range and when number of direction of motion is more path length remains almost same with increase in sensing range. Furthermore path length decreases with increase in number direction of motion. From the bar graph plotted for tame taken it can be observed that when number of direction of motion is less time taken decreases with the increase in sensing range and when number of direction of motion is more time taken increases with increase in sensing range.

IV. Conclusion:

In this work a new algorithm for navigating robot in unknown static environment where sensing range of robot is much smaller than size of the obstacle is proposed. The proposed algorithm is tested in various environments and was able to find out satisfactory path having no collision with obstacles and minimum path length. The path length produced by proposed algorithm is compared with number of previously proposed algorithms and it was found to be less than previously proposed algorithms.

References

1) BBVL Deepak, Dayal R. Parhi (2012)-PSO Based path planner of an autonomous mobile robot-Central European journal of computer science 2(2).2012.152-168 – DOI:10.2478/s 13537-012-0009-5

2) Rahul Kala, AnupamShukla, Ritutiwari (2011)- Robotic path planning in static environment using hierarchical multi-neuron heuristic search and probability based fitness-Neurocomputing 74 (2011) 2314–2335-DOI: 10.1016/j.neucom.2011.03.006

3) Pratap Kumar Panigrahi, SaradinduGhosh, Dayal R Parhi (2014)- A Novel Intelligent Mobile Robot Navigation Technique for Avoiding Obstacles using RBF Neural Network- 2014 International Conference on Control, Instrumentation, Energy & Communication(CIEC) 978 - 1-4799 – 2044 - 0/14/$31.00©2014IEEE

4) Qingbao Zhu, Jun Hu, wenbinCai, Larry Henschen (2011)-A new robot navigation algorithm for dynamic unknown environment based on dynamic path re-computation and an improved scout ant algorithm-Applied soft computing 11 (2011) 4667-4676- DOI:10.1016/j.asoc.2011.07.016

5) MarijaDakulović, Ivan Petrović (2011)- Two-way D^* algorithm for path planning and replanning- Robotics and Autonomous Systems 59 (2011) 329–342-DOI:10.1016/j.robot.2011.02.007

6) TAN Guan-Zheng, HE Huan, SLOMAN Aaron (2007)- Ant Colony System Algorithm for Real-Time Globally Optimal Path Planning of Mobile Robots-ActaAutomaticaSinica Vol-33 No.3,March-2007-DOI:10.1360/aas-007-0279

7) VahidAzimirad, HamedShorakaei (2013)- Dual hierarchical genetic-optimal control: A new global optimal path planning method for robots-Journal of Manufacturing Systems 33 (2014) 139– 148-DOI:10.2016/j.jmsy.2013.09.006

8) Usman Ahmed Syed, FarazKunwar, MazharIqbal (2013)- Guided Auto-wave Pulse Coupled Neural Network (GAPCNN) based real time path planning and an obstacle avoidance scheme for mobile robots-Robotics and Autonomous Systems 62 (2014) 474–486-DOI: 10.1016/j.robot.2013.12.004

9) Md. Arafat Hossain, IsratFerdous (2014)- Autonomous robot path planning in dynamic environment using a new optimization technique inspired by bacterial foraging technique. Robotics and Autonomous Systems 64 (2015) 137–141-DOI: 10.1016/j.robot.2014.07.002

10) Hongwei Mo and LifangXu (2014)- Research of biogeography particle swarm optimization for robot path planning- Neurocomputing 148 (2015) 91–99-DOI: 10.1016/j.neucom.2012.07.060