Cellular Automata Based Real Time Path Planning for Mobile Robots

Syed Usman Ahmed, Arsalan Akhter and Faraz Kunwar
Department of Mechatronics Engineering
College of Electrical and Mechanical Engineering
National University of Sciences and Technology (NUST)
Islamabad, Pakistan
Email: s.usman@ceme.nust.edu.pk

***Abstract*—Intelligent mobile robotic agents demand optimal motion planners with minimum query time. Most contemporary algorithms lack one of these two required aspects. This paper proposes a cellular automata (CA) based efficient path planning scheme that generates optimal paths in minimum time. A Cellular automata is evolved over the entire environment and subsequently used for shortest path determination. This approach generates a parent-child relationship for each cell in order to minimize the search time. Analysis and simulation results have proven it to be a robust, complete path planning scheme and time efficient both in static and dynamic environments.**

I. INTRODUCTION

Path planning is an essential constituent of the design of an intelligent mobile robot. The field of motion planning has been explored in detail by researchers in the past two decades with the aim to develop an algorithm that shows convergence to optimal path in minimum possible time. This work is aimed at developing an efficient path planner that provides shortest path from robot to goal location.

Visibility graphs[1,2] and Voronoi diagrams[3] are among one of the earlier techniques explored for optimal path search.These algorithms gave promising results but they were mainly developed for static environments.Furthermore, visibility graph is a computationally expensive algorithm and hence can't be used in real time systems. Also, Voronoi diagrams fail to provide optimal solution. Another class of algorithms developed for this purpose are cell decomposition methods[4]. However computational efficiency of cell decomposition techniques is highly dependent upon the size of the cells, thereby making it an inefficient algorithm for real time usage. Another approach is the use of Potential Field [5] methods which provides assuring results in most cases but fails in some specific situations where attractive and repulsive fields tend to cancel out each other i.e. local minima[6].

In order to overcome the problems posed to above mentioned algorithms, probabilistic approaches namely PRM's[7] and RRT's[8,9] were developed. These algorithms are capable of finding paths in complex environments but due to their probabilistic nature they do not provide optimal solutions. Furthermore their sampling strategies tend to become complex in case of narrow passages.It has been proven that RRT does not approach optimality whereas RRT*[10] requires an infinite number of iterations to converge to optimality.

More recently machine learning paradigms are being utilized for global obstacle free path search.One of the major advances in this paradigm are use of genetic algorithms[11,12] and fuzzy based[13,14] approaches for extraction of global collision free paths. Another promising learning paradigm for this problem is the use of neural networks[15]. The problem with these approaches is that they require large database to learn and generalize. It is mostly difficult and sometimes impossible to provide such kind of data. However an exception to the rule is the use of Modified Pulse Coupled Neural Network (MPCNN)[16]. It is capable of path planning in arbitrarily complex environments and is proven to provide a global shortest paths. The problem with this technique is that it is computationally inefficient. A comparison with this technique is presented in the results section.

Finite automata is a class of algorithms with discrete input and outputs. Cellular automata[17] are a special class of finite automata which constitutes n-dimensional array of cells wherein each cell can take a set of possible values. Path planning using cellular automata has been previously addressed and a straight moving path planner has been proposed in [19] which derives its strategy from a multi agent path planning architecture using cellular automata [18]. However, this approach does not provide an optimal solution since it requires an elaborate search of the environment which degrades the efficiency of the method. Another similar sub-optimal approach that utilizes cellular automata is given by Behring et. al [20]. A case in which a diamond shaped robot is considered for navigation in a 2D environment using cellular automata is presented in [21]. In this strategy cellular automata is used to determine cells that are equidistant from obstacles which later determines the path of robot. Since the path is required to be equidistant from all obstacles therefore it results in a non-optimal solution.

This paper presents a cellular automata based approach to compute the shortest path in a 2D configuration space. Rule based exploration of the environment is coupled with parent-child relationships for each cell to simplify the search process. Simulations have verified it to be an efficient method both in the presence of static as well as dynamic obstacles.

Rest of the paper is organized as follows. Section II discusses the environmental setup and a preprocessing for the proposed method. Section III explains the proposed algorithm in detail.Pseudo code of the given method is presented in section IV. Next section V presents the results and sectionVI concludes this paper.

II. Configuration Space

The space of all possible configurations of a robot is called the configuration space[24]. Consider a robot R navigating in a 2-D Euclidean space, where the set of all possible configurations of the robot are represented by the set $Q = \{q_1, q_2, ..., q_n\}$ and the set of obstacles is represented by $O = \{O_1, O_2, ..., O_n\}$. Then the configuration space can be modeled as a continuous mapping represented by $\tau : [0,1] \rightarrow Q$, where $\tau(0) = q_{init}$ and $\tau(1) = q_{goal}$. The path planning problem is to find a path in the configuration space such that no configuration of the robot collides with the obstacles. In other words, the problem is to find a set of configurations of the robot from q_{init} to q_{goal} in the free configuration space where

$$Q_{free} = \{q \in Q \,|\, R(q) \cap (\bigcup_{i=1}^{n} O_i) = \phi\} \tag{1}$$

Since the Robot can be of any arbitrary shape in the workspace, the profile of the robot needs also to be considered in the configuration space. This is done by taking the Minkowski Sum [22] of the profile of the robot R with every obstacle O_i. Hence every obstacle in the configuration space is remodeled such that

$$R \oplus O_i = \{x + y \,|\, x \in R, y \in O_i\} \tag{2}$$

Since the shape of the robot has been catered, the robot can now be considered as a point robot in the configuration space.

III. Cellular Automata Model

Cellular automata are a four tuple, decentralized, discrete space-time systems defined over a cellular space[18]. Cellular automata consist of a cellular space consisting of a large number of locally connected identical entities, where each entity is updated based on a set of transition rules.

Cellular Automata are formally defined as quadruples (d, q, N, f), where

d	Dimension of the cellular automata.
q	Set of possible states of cellular automaton.
N	Set of relative positions of neighboring cells.
f	Local function defining the local transition rule.

In the proposed algorithm, the CA architecture consists of a 2-D lattice of cells. Each cell constitutes of a six element tuple, which are used in the evaluation of the local transition function. These are

$$(S_{state}, S_{cf}, S_{pf}, t_{on}, P_{(i,j)}, \phi_s)$$

where

S_{state}	Current state of the cell. 0 for off, 1 for on.
S_{cf}	Child Flag. High if the cell is a child.
S_{pf}	Parent Flag. High if the cell is a Parent.
t_{on}	Time at which the cell transitioned to active state.
$P_{(i,j)}$	Address of the parent cell.
ϕ_s	Maintains the accumulative cost from Goal to the current cell.

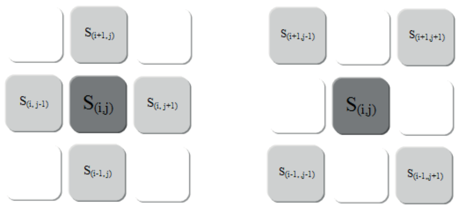

Fig. 1: The Moore Neighborhood

Cells are interconnected with the local neighborhood within the constraints of the 8-nn (nearest neighbors), which is also called type-II neighborhood or the Moore Neighborhood[17] which, for example, consists of nine sites for d = 2, as shown in Fig. 1

The activation of each automaton is governed by the following transition rules:

Rule: 1 - The current cell will only become active if and only if one of the neighboring cell is in active state and the t_{on} of the neighboring cell causing it to activate must be less than t (where t is the time of the current iteration). Otherwise, it will remain quiescent.

$$S^t_{state}(i,j) = \begin{cases} 1 & iff \quad \exists S^t_{state}(i+m, j+n) = 1 \\ & \quad\& \; t_{on}\ (i+m, j+n) < t \\ & \quad m, n \in \{1, -1\} \\ 0 & otherwise \end{cases} \tag{3}$$

Rule: 2 - A cell will only become a child of another cell if the other cell is already in active state with a $t_{on} < t$.

$$S^t_{cf}(i,j) = \begin{cases} 1 & iff \quad \exists S^t_{state}(i+m, j+n) = 1 \\ & \quad\& \; t_{on}\ (i+m, j+n) < t \\ & \quad m, n \in \{1, -1\} \\ 0 & otherwise \end{cases} \tag{4}$$

Rule: 3 - A cell will only become a parent cell iff it is active, it is already a child and its $t_{on} < t$.

$$S^t_{pf}(i,j) = \begin{cases} 1 & iff \quad \exists S^t_{state}(i,j) = 1 \\ & \quad\& \; S^t_{cf}(i,j) = 1 \\ & \quad\& \; t_{on}\ (i,j) < t \\ 0 & otherwise \end{cases} \tag{5}$$

Rule: 4 - A cell will only become a child of a neighboring cell with lowest accumulative cost function.

$$\phi_s(i,j) = \begin{cases} min(\phi_s(i+m, j+n) + \delta) & iff \\ \quad \exists S^t_{state}(i+m, j+n) = 1 \\ \quad \& \; S^t_{pf}(i+m, j+n) = 0 \\ \quad m, n \in \{1, -1\} \\ 0 \quad otherwise \end{cases} \tag{6}$$

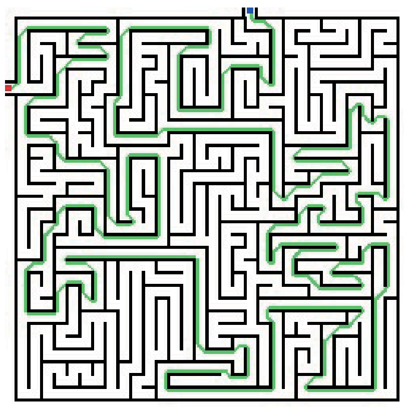

Fig. 2: Solution of a complex maze using CA.

where δ is the cost of connecting current cell $S(i,j)$ with the neighboring cell. The costs are

$$\delta = \begin{cases} |1| & \forall S(i+m,j), S(i,j+n) \\ & m,n \in \{-1,1\} \\ |\sqrt{2}| & \forall S(i+m,j+n) \\ & m,n \in \{-1,1\} \end{cases}$$

Rule: 5 - A cell will remember the address of its parent cell as calculated in Rule (4).

$$P_{(i,j)} = \{(r,s) | \phi_s(i,j) = min(\phi_s(r,s) + \delta)\} \tag{7}$$

A. Completeness of the Algorithm

The proposed algorithm is a complete algorithm i.e. if a solution exists, it will find it, otherwise it'll assert that no solution exists. At every iteration, the algorithm keeps a count of the no. of cells which were activated during that iteration. If, for an iteration, none of the cell is activated, this means that there are no more cells which can be activated, and hence no path exists. On the other hand, as soon as the algorithm finds a path from the goal to the robot, it is the optimal path which is declared as output.

To prove it, we suppose that the contrary is true, i.e. the algorithm is not complete. Therefore, there exists a path from start to goal, but the algorithm couldn't find it. For this statement to be true, either the algorithm never terminates, or it terminates incorrectly.

Suppose it never terminates. But at any instant, if a parent child relationship could be built, a new cell will be activated. According to Rule 4, a situation can arise where there is no other neighbor left with an accumulative cost lower than the current parent. Also there might be no cell in the neighborhood which is in quiescent state. Hence if there is no possible parent child relationship left, no cell will be activated. If no cell is activated, the algorithm will declare that no path exists from start to goal, and will terminate.

Suppose it terminates incorrectly. According to Rule 1, a cell can only become active if its neighboring cell was active. Also the obstacle cells are configured to be permanently off. Hence an obstacle cell will never become a child, and according to Rule 3, it will never become a parent as well. Since an obstacle cell will never be activated, the resultant path will never end inside an obstacle. Also, since no cell can become active if its neighbors are not active according to Rule 1, only successive parent child relationships are created, which only terminate when the start point and goal point are connected, and hence produce a continuous path from start to goal.

Since both the above statements have been proved false, the opposite is true, i.e. the algorithm is complete.

IV. The Algorithm

To plan a path using the proposed technique, all the cells are initialized according to their state of occupancy. Obstacles are initialized as NaN and free space as state zero. Algorithm proceeds with the switching of states in outward fashion from goal location un till robot cell is reached. In the next step path is extracted from robot to target using parent child relationships.

Computation steps of the algorithm are as under:

```
1) Initialization:
   $\forall S \in q_{obstacle}$=NaN, $\forall S \in q_{free}$=0, $S_{target} = 1, S_{cf} = true$,
   $\phi_{target} = 0, t_{on}^{target} = 0$ and $P(i,j) = (i,j)$
2) Cellular Automata Iterations:
   Repeat
   Initialize firing count to zero.
   For each cell
     a) if($S^t_{state}$ == 0)
        result=Execute rule 1 using (3).
        if(result==true)
            Set $t_{on}$ for current cell.
            Set child flag using rule 2 as given by (4).
            Update cost of current cell using (6).
            Configure parent of current cell using (7).
            Increment firing count.
            else
            $continue$
        else
        if ($S_{pf}$== false)
            if ($t_{on} < t$)
            Configure $S_{pf}$ using (5)
   if ($S_{robot}$==1)
        return path
   else if (firing count == 0 )
        return failure
```

V. Results

In order to gauge the performance of the proposed algorithm we tested it in both static as well as dynamic environments.In our experiments it has demonstrated efficient in terms of time and provides optimal paths. Some of these results are discussed here. In all the subsequent figure red represents a robot, blue shows the target configuration, black shows obstacles while green shows the path extracted by the planner.The algorithm

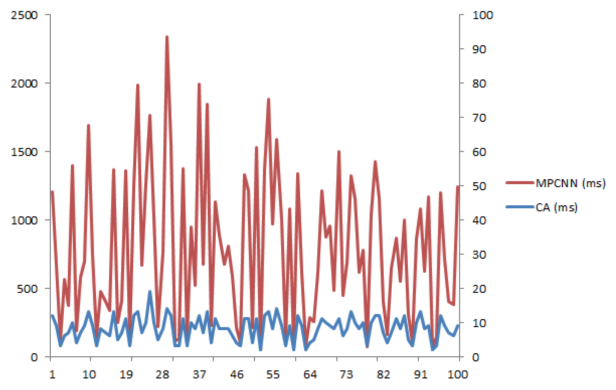

Fig. 3: Time comparison of CA with MPCNN.

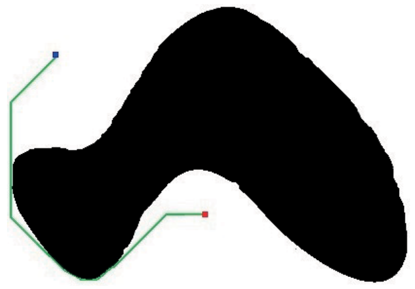

Fig. 4: Path planned by CA in the presence of a local minima

was tested in a challenging maze as shown in Fig. 2, in which it successfully provides an optimal solution.

A. Performance Comparison

Cellular automata based proposed method was tested against a recently proposed Modified Pulse Coupled Neural Network (MPCNN). MPCNN has been shown to be a robust method and determines optimal path. Fig. 3 shows a comparison between CA and MPCNN on basis of computation time where time axes for both algorithms are shown separately (right axis for CA while left for MPCNN). Hundred random cases were tested, Fig. 3 clearly demonstrates that CA outperforms MPCNN while path determined by CA was same as that of MPCNN. Statistically average time taken by CA was 8.19ms whereas that of MPCNN was 749.94ms.

B. Local Minima Evasion

In order to verify the completeness of our algorithm, the algorithm was tested to plan a path in the presence of a local minima. Fig. 4 shows the resultant path in the presence of a local minima. It is clearly evident that proposed method evades a local minima efficiently.

C. Static environment

Real world robots often makes use of a SLAM based mapping of the environments. Algorithm was tested in real world SLAM environments as shown in Fig. 5.

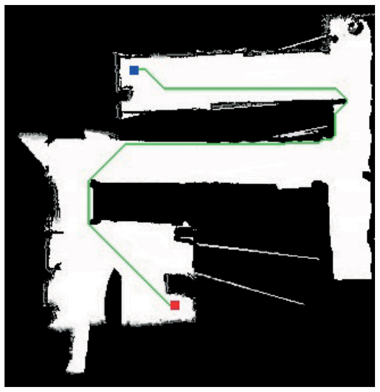

Fig. 5: Path planned by CA in 2D SLAM environment.

D. Dynamic environment

To ensure that the proposed scheme works in real time, it is mandatory for the algorithm to work in the presence of dynamic obstacles. To ascertain this we simulated the real world scenarios on PlayerStage[23] simulator. Fig. 6 to Fig. 8 presents results for dynamic obstacle avoidance where top row represents the path covered by each robot from its starting location to current time step. In these figures there are up to a maximum of four robots, dynamic obstacle ‘A’ is shown with a blue trajectory, ‘B’ is shown with green trajectory while obstacle ‘C’ with a yellow trajectory while intelligent path planner robot ‘R’is shown with a red trajectory. Here ‘T’ represents the target cell for the robot.

Fig. 6 represents a test scenario where robot ‘R’ has to intelligently traverse through a L shaped hallway to reach its destination. Initially robot plans a path and starts navigation as shown in Fig. 6(a), while it is about to have a head on collision with obstacle ‘A’. Fig. 6(b) shows that it successfully avoids dynamic obstacle by rerouting its path. Here planner presents a solution that makes ‘R’ to traverse in-front of obstacle ‘B’ but this decision is changed in the next time step. While traversing towards goal, ‘R’ is intercepted by obstacle ‘B’ and in this case planner finds its optimal path by adopting a path to the right of obstacle as shown in Fig. 6(c). Fig. 6(d and e) shows that it successfully evades obstacle ‘B’.

Fig. 7 (a) shows a case where a robot has to reach a goal location avoiding three closely moving dynamic obstacles in a narrow hallway. Fig. 7 (b) presents a how CA based path planner avoids obstacle ‘A’. After avoiding obstacle ‘A’, robot ‘R’ is encountered by obstacle ‘B’ which is about to have a head on collision with robot ‘R’ as shown in Fig. 7(c). It is successfully avoided by moving to right as shown in Fig. 7(d) whereas on the other hand it is immoderately encountered by obstacle ‘C’ which it avoids as shown in Fig. 7(e).

Another real world scenario is where the target is non static. Fig. 8 shows a case in which target is moving in a sinusoidal manner. Fig. 8 (a and b) shows how initially pursuer ‘R’ tries to catch target robot ‘T’. As long as target is moving towards right pursuer also traverses to right as shown as Fig. 8(c and d). As soon as target changes direction robot changes it too

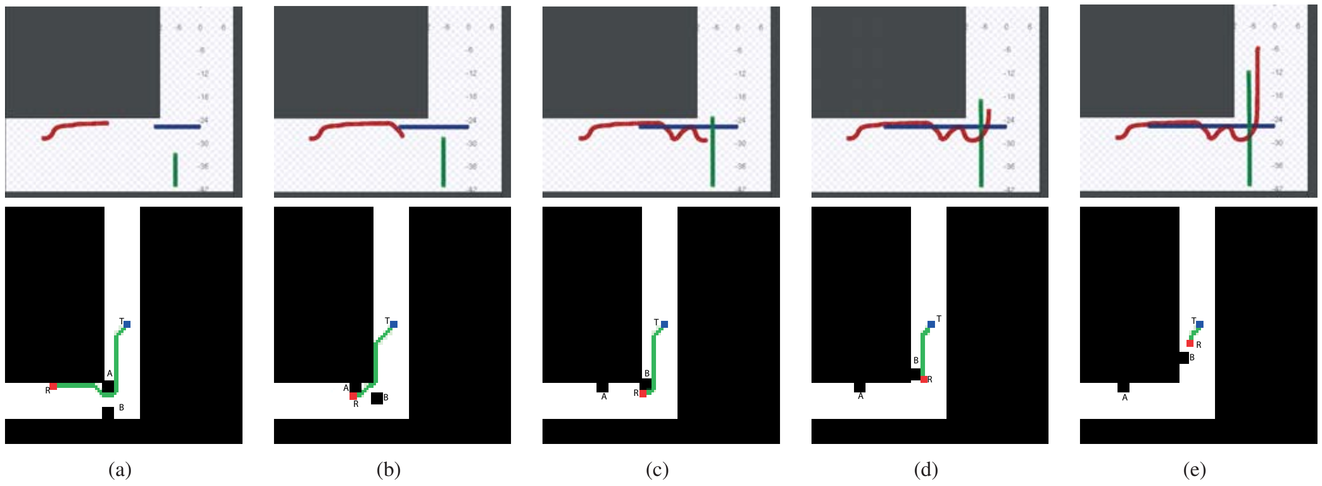

Fig. 6: Dynamic environment with two moving obstacles.

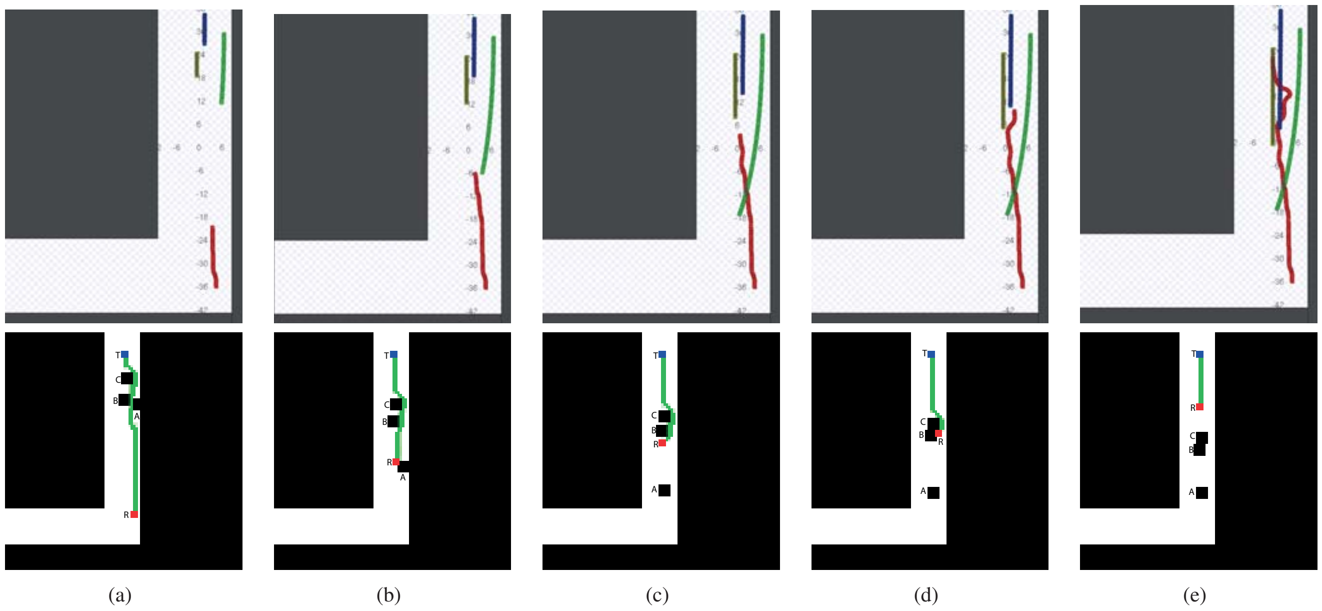

Fig. 7: Dynamic environment with three dynamic obstacles.

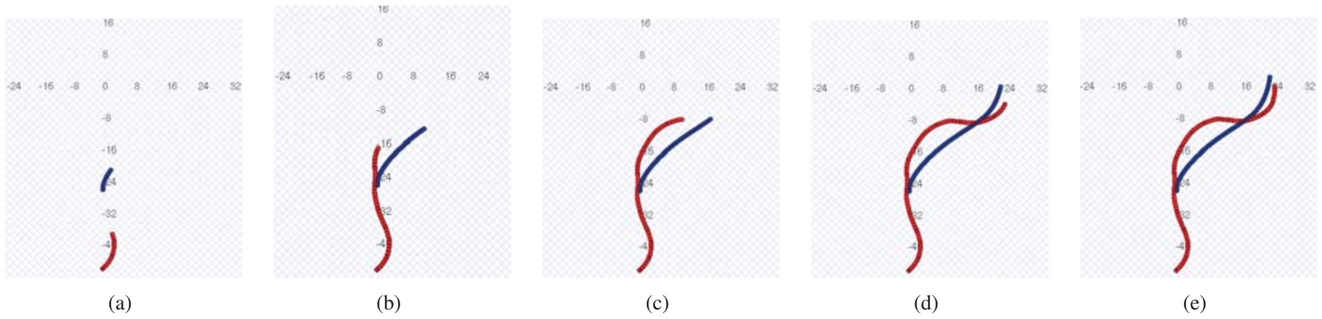

Fig. 8: Pursuer robot with a target moving in a sinusoidal manner .

and ultimately catches it as shown in Fig. 8(e).

VI. Conclusion

The paper presents a cellular automata based real time path planner that always results in an optimal path. Cellular automata is coupled with a parent-child relationship for each cell to achieve improved and real time performance. PlayerStage simulations are conducted to validate the real time behavior of the proposed scheme. The results prove that it outperforms previous path planning algorithms in the light of optimality and time efficiency as shown in comparison with MPCNN.

References

[1] Kito, T.; Ota, J.; Katsuki, R.; Mizuta, T.; Arai, T.; Ueyama, T.; Nishiyama, T.; , "Smooth path planning by using visibility graph-like method,"in Proc. of IEEE International Conference on Robotics and Automation (ICRA), 2003, vol.3, no., pp. 3770- 3775 vol.3, 14-19 Sept. 2003.

[2] Han-Pang Huang; Shu-Yun Chung; , "Dynamic visibility graph for path planning," in Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS), vol.3, no., pp. 2813- 2818 vol.3, 28 Sept.-2 Oct. 2004.

[3] Takahashi, O.; Schilling, R.J.; , "Motion planning in a plane using generalized Voronoi diagrams," in IEEE Transactions on Robotics and Automation , vol.5, no.2, pp.143-150, Apr 1989.

[4] Lingelbach, F.; ,"Path planning using probabilistic cell decomposition," in Proc. of IEEE International Conference on Robotics and Automation (ICRA) 2004 , vol.1, no., pp. 467- 472 Vol.1, 26 April-1 May 2004.

[5] Khatib, O.; , "Real-time obstacle avoidance for manipulators and mobile robots," in Proc. of IEEE International Conference on Robotics and Automation (ICRA), vol.2, no., pp. 500- 505, Mar 1985.

[6] Y. Koren, "Borenstein,Potential Field methods and their inherent limitations for mobile robot navigation," in Proc. of IEEE International Conference on Robotics and Automation (ICRA), pp 1398-1404, 1991.

[7] Kavraki, L.E.; Svestka, P.; Latombe, J.-C.; Overmars, M.H.; , "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," in IEEE Transactions on Robotics and Automation, vol.12, no.4, pp.566-580, Aug 1996.

[8] LaValle,S. M. ; Kuffner,J. J. ; "Rapidly-exploring random trees: Progress and prospects." in proc. of Workshop on the Algorithmic Foundations of Robotics, 2000.

[9] Kuffner, J.J., Jr.; LaValle, S.M.; , "RRT-connect: An efficient approach to single-query path planning , " in Proc. of . IEEE International Conference on Robotics and Automation, 2008 (ICRA), vol.2, no., pp.995-1001, 2000.

[10] Karaman, S.; Frazzoli, E.; , "Sampling-based Algorithms for Optimal Motion Planning," in International Journal of Robotic Research, vol., no., pp., 2011.

[11] Jianping Tu; Yang, S.X.; , "Genetic algorithm based path planning for a mobile robot," in Proc. IEEE International Conference on Robotics and Automation (ICRA), 2003. vol.1, no., pp. 1221- 1226 vol.1, 14-19 Sept. 2003.

[12] Woong-Gie Han; Seung-Min Baek; Tae-Yong Kuc; , "Genetic algorithm based path planning and dynamic obstacle avoidance of mobile robots," in Proc. of IEEE International Conference on Systems, Man, and Cybernetics, Computational Cybernetics and Simulation, 1997 , vol.3, no., pp.2747-2751 vol.3, 12-15 Oct 1997.

[13] Xiaoyu Yang; Moallem, M.; Patel, R.V.; , "A layered goal-oriented fuzzy motion planning strategy for mobile robot navigation," in IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics. vol.35, no.6, pp.1214-1224, Dec. 2005.

[14] Yanrong Hu; Yang, S.X.; , "A knowledge based genetic algorithm for path planning of a mobile robot," in Proc. of IEEE International Conference on Robotics and Automation(ICRA). vol.5, no., pp. 4350-4355 Vol.5, 26 April-1 May 2004.

[15] Yongmin Zhong; Shirinzadeh, B.; Yanling Tian; , "A new neural network for robot path planning," in Proc. of IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM) 2008, vol., no., pp.1361-1366, 2-5 July 2008.

[16] Hong Qu; Yang, S.X.; Willms, A.R.; Zhang Yi; , "Real-Time Robot Path Planning Based on a Modified Pulse-Coupled Neural Network Model," in IEEE Transactions on Neural Networks, vol.20, no.11, pp.1724-1739, Nov. 2009.

[17] Wolfram, S. "Statistical Mechanics of Cellular Automata." Rev. Mod. Phys. 55, 601-644, 1983.

[18] Y. Tavakoli; H. Haj Seyyed Javadi and S. Adabi; A Cellular Automata Based Algorithm for path planning in Multi-Agent Systems with a common Goal, International Jornal of Computer Science and Network Security (IJCSNS), July 2008.

[19] Soofiyani, F.R.; Rahmani, A.M.; Mohsenzadeh, M.; ,"A Straight Moving Path Planner for Mobile Robots in Static Environments Using Cellular Automata," in Proc. of International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN), 2010, vol., no., pp.67-71, 28-30 July 2010.

[20] C. Behring; M. Bracho; M. Castro; J. A. Moreno; , " An Algorithm for Robot Path Planning with Cellular Automata," in Proc. of International Conference on Cellular Automata for Research and Industry (ACRI) 2000, pp. 11-16, Springer, 2000.

[21] Tzionas, P.G.; Thanailakis, A.; Tsalides, P.G.; , "Collision-free path planning for a diamond-shaped robot using two-dimensional cellular automata," in IEEE Transactions on Robotics and Automation, vol.13, no.2, pp.237-250, Apr 1997.

[22] Oks; Eduard; Sharir; Micha; "Minkowski Sums of Monotone and General Simple Polygons," in Discrete and Computational Geometry, Vol. 35, Issue 2, pp. 223-240, 2006.

[23] Brian Gerkey; Richard T. Vaughan and Andrew Howard. "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems." in proc. of the 11th International Conference on Advanced Robotics (ICAR 2003), pages 317-323, Coimbra, Portugal, June 2003.

[24] H. Choset; K.M. Lynch; S. Hutchinson; G. Kantor; W. Burgard; L.E. Kavraki; S. Thrun; "Principles of Robot Motion: Theory, Algorithms, and Implementations,", MIT Press 2005.