A Guided Autowave PCNN for Improved Real Time Path Planning

Syed Usman Ahmed,Usman Ali Malik, Mazhar Iqbal and Faraz Kunwar

Abstract— **Real time path planning for mobile robots requires fast convergence to optimal paths. Most rapid collision free path finding algorithms do not guarantee the optimality of the path. In this paper we present a Guided Autowave Pulse Coupled Neural Network (GAPCNN) approach for mobile robot path planning. The proposed model is a novel approach that improves upon the recently presented Modified PCNN by introducing directional autowave control and accelerated firing of neurons based on a dynamic thresholding technique. Simulation and experimental evaluation in both static and dynamic environments confirm GAPCNN to be a robust and time efficient path planning scheme for finding optimal paths.**

I. INTRODUCTION

The development of an Intelligent Transportation System (ITS), which can drive autonomously to a desired location, requires solving such problems as Localization, Path Planning and Autonomous Obstacle Collision Avoidance. One of the earliest and most well-known problems for such a system specifically in the indoor domain, is the generation of collision free global path for the robot to move to a given point in a dynamic environment. Conventional path planning approaches employed for such motion planning can be categorized into four types including Visibility graphs [1], [2], Cell Decomposition [3], Voronoi Diagrams [4] and the Potential Field methods [5]. Most of these algorithms suffer from time inefficiency in their computation and are not meant for use in real-time path planning. Also, Potential Field methods are known to suffer unwanted local minima in which the robot gets stuck in a U-shaped obstacle [6].

Sampling-based algorithms such as Probabilistic Road Maps [7] improve on the limitations of the previously mentioned algorithms but because of their iterative nature, they are not well suited to real-time implementations in highly cluttered environments. Techniques that use the Monte Carlo based growing data structures like Rapidly Growing Random Trees [8], [9], can find paths in complex environments but they are proven not to approach optimality. RRT* [10], however, does approach optimality but requires infinite iterations to do so. Search algorithms (e.g A* [11]) have also been employed for global free path searching. In case of weighted A*, selection of suitable weights is itself a problem and inaccuracy in weights results into suboptimal paths. More recently genetic algorithms [12], [13] and fuzzy approaches [14], [15] have been proposed for collision free path planning. Optimality of path returned by fuzzy planners is dependent on the quality of rules and often end result is a suboptimal paths.

Neural network approaches have also been employed for efficient planning of the shortest path. Glasius et al. [16] devised a method based on Hopfield network for motion trajectory generation while avoiding obstacles. More recently cellular neural network [17] and reinforcement learning [18] based algorithm have also been presented. Another promising approach is the use of a Pulse Coupled Neural Network (PCNN) which is a biologically inspired algorithm capable of emulating the behavior of a cat's visual cortices [19]. After the earlier work on PCNN done by Johnson [20]- [21], the algorithm has been subject to extensive research in various fields such as image processing and pattern recognition [22]. By employing the autowave approach in the PCNN, Caulfield and Kinser [23] proposed a method to solve maze problems. Their approach finds the shortest path where computational complexity is related to the path length. However this approach needs a large number of neurons to find paths in large mazes thereby making it computationally expensive. Improving on the limitations of the aforementioned PCNN approach, a modified PCNN (MPCNN) model was recently introduced by Qu et al. [24] which has been proven to be more efficient than most of the algorithms. The modified model requires fewer neurons than the Caulfield and Kinser model, guaranteeing the shortest path and a solution that is independent of the complexity of the search space. However, since the model employs an unconstrained autowave, it searches the whole space irrespective of where the target position is located, hence unconstrained search leads to time inefficiency.

In order to facilitate an informed search, we present a novel path planning approach that employs a directionally constrained Guided Autowave Pulse Coupled Neural Network (GAPCNN) model. The model, which works on a 2D configuration space, uses the position of target neuron to focus a controlled search in its direction. Also, it employs a variable threshold unique to each neuron. These modifications enable the proposed scheme to significantly improve the optimal path query times, in both static and dynamic settings, as compared to the MPCNN.

Rest of the paper is organized in the following manner. Section II covers some related research on MPCNN based path planning. The proposed GAPCNN model is described in Section III. Section IV deals with the comparison of MPCNN and GAPCNN. In Section V, simulation and experimental

This work was supported in part by the Department of Mechatronics Engineering, College of Electrical and Mechanical Engineering (CEME), National University of Sciences and Technology (NUST), Islamabad, Pakistan. All authors are affiliated with Mechatronics Engineering Department, NUST.

Syed Usman Ahmed (s.usman@ceme.nust.edu.pk)
Usman Ali Malik (usman.malik@smme.nust.edu.pk)
Mazhar Iqbal (drmazhar@ceme.nust.edu.pk)
Faraz Kunwar (k.faraz@ceme.nust.edu.pk)

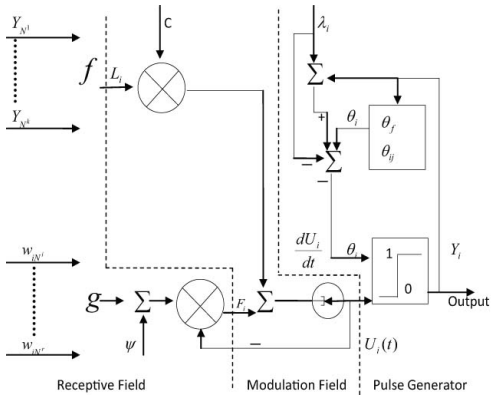


Fig. 1: The GAPCNN Model.

results are discussed. Finally conclusions are given in Section VI.

II. MPCNN Based Path Planning

Here we briefly review some fundamental concepts of MPCNN algorithm for mobile robot path planning. MPCNN considers the environment map as a 2D grid of neurons. Then current robot position and target location are identified. To find the shortest path, search proceeds through the use of a circular autowave. The autowave propagates through coupling of the neighboring neurons. When the internal activity of a particular neuron exceeds the threshold level, it fires. Initially the target neuron's internal activity is set greater than the threshold level i.e. a necessary condition to initiate the firing process. The firing of a neuron is followed by coupling process in which neighboring neurons are bonded by a parent-child relationship. Subsequently the internal activity of the neighboring neurons increases with time. This firing pattern propagates outwards in the form of a wave until the robot neuron is reached. In this process, internal activity of the obstacle neurons is kept zero, hence avoiding their firing. Path is traced backwards from robot to target through the parental sequence neurons.

An analysis of the algorithm reveals that it suffers in two important aspects. Firstly it utilizes an uninformed search behavior. To find the shortest path it propagates a wave in all directions, not using the available information which could be utilized to perform an intelligent search. Secondly MPCNN assigns the same threshold level to all the neurons in the provided map. As we demonstrate in this paper, gradual decrease in the threshold in the direction of the possible solution can be utilized to find the shortest path with better time efficiency. We term the new model as Guided Autowave PCNN (GAPCNN). GAPCNN utilizes informed search and threshold constraints by employing mathematical expressions of least computational complexities; thereby leading to a computationally efficient solution to the presented problem. Moreover, firing rate based wavefront expansion ensures its completeness. GAPCNN outperforms the previous method with the same path planning results, but with much more time efficiency.

III. GAPCNN Model

The GAPCNN model for a single neuron i is shown in Fig.1. The model is based on the MPCNN in [24] which has a laterally connected grid structure of neurons over the entire search space. Each neuron in the space is connected only to its immediate neighbors. The neighbors of a particular neuron are said to be in its straight connected set N^s if their Euclidean distances from the neuron are 1. Likewise the neighbors are said to be in the slantwise connected set N^d of the neuron if their Euclidean distances are $\sqrt{2}$. The neurons need not be actually spaced a unit distance (or $\sqrt{2}$ in slantwise direction) apart from each other. The neurons can be placed at greater distances for less cluttered environments, provided the adherence to straight and slantwise connection rules is ensured i.e. slantwise distance is always greater than straight and in proportion with the above mentioned distances . A neuron is said to fire at time T if $Y_i(t) = 1$ at $t = T$ and $Y_i(t) = 0$ for all $t \neq T$. It is important to note here that a neuron only fires once in its lifetime. A neuron i fires only if some neuron in its neighborhood N_i fires and sets it up to fire at some later time. The neuron is said to be its temporary parent N_i^{P*} and its firing time is denoted as $t^{N_i^{P*}}$. If another neuron in the same neighborhood fires and if it can make neuron i fire earlier, then it will become the new temporary parent of i. When a neuron fires at some time t_{fire}^i it records its current temporary parent as its actual parent with its parent firing time $t^{N_i^P}$. The model differs from the MPCNN model in two ways. First the threshold function of the proposed model is a function of time t and distance λ from the target while the MPCNN threshold is a function of time only. The modified threshold has distinct values over the whole space and it decreases towards the target neuron. The threshold for i^{th} neuron can be written as:

$$\theta_i(t,\lambda) = \begin{cases} \theta_{init} & \text{for} \quad t < t^{N_i^{P*}} \\ \theta_{ij} - \lambda_i & \text{for} \quad t^{N_i^{P*}} \leq t < t_{fire}^i \\ \theta_f & \text{for} \quad t > t_{fire}^i \end{cases} \tag{1}$$

where θ_{init} and θ_f are constants. θ_f is set to be a very large value which ensures that the neuron may not fire more than once and θ_{ij} is given as:

$$\theta_{ij} = \begin{cases} \theta_s & \text{if} \quad j \in N^s \\ \theta_d & \text{if} \quad j \in N^d \end{cases} \tag{2}$$

where j is the temporary parent of neuron i; θ_s and θ_d are constants and λ_i is the normalized distance of the i^{th} neuron from the target neuron given by:

$$\lambda_i = A\left(1 - \frac{D_{target}}{D_{max}}\right) \tag{3}$$

Here D_{target} is the Euclidean distance from the target neuron, D_{max} is taken as the distance between the two farthest neurons in the network, and A is a constant.

Second major difference of the proposed model from the MPCNN model is the actual directional constraining of the autowave. This is achieved by employing an angle modification in the original differential equation (as proposed in

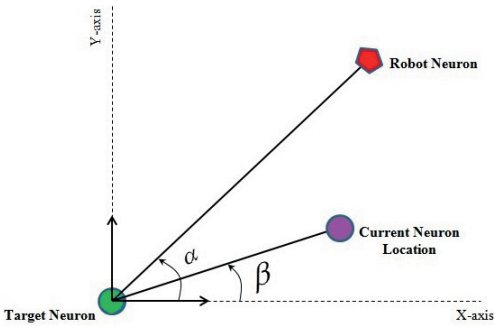


Fig. 2: Illustration of the direction constraint.

[24]) of the internal activity. The proposed internal activity value $U_i(t)$ is given by:

$$\begin{cases} \frac{dU_i(t)}{dt} = F_i + CL_i & t \geq t^{N_i^P} \\ U(t^{N_i^P}) = 0 \end{cases} \tag{4}$$

Where C is a constant $F_i(t)$ and $L_i(t)$ are the linking and feeding fields given by:

$$L_i(t) = f(Y_{N^1}, \ldots, Y_{N^k}, t) = \begin{cases} 0 & \text{if} \quad t < t^{N_i^{P*}} \\ 1 & \text{otherwise} \end{cases} \tag{5}$$

$$F_i(t) = -g(w_{iN^1}, \ldots, w_{iN^k}, \psi, t) U_i(t) \tag{6}$$

Here g is a function of time t , connection weights among the k neighbors and the directional constraint ψ and is given by:

$$g(w_{iN^1}, \ldots, w_{iN^k}, \psi, t) = \begin{cases} 0 & \text{if} \quad t < t^{N_i^{P*}} \\ \mu(w_{ij}) + \psi_i & \text{if} \quad t > t^{N_i^{P*}} \end{cases} \tag{7}$$

where $\mu(w_{ij})$ is given by:

$$\mu(w_{ij}) = \frac{B}{w_{ij}} = \begin{cases} B & \text{if} \quad j \in N^s \\ \frac{B}{\sqrt{2}} & \text{if} \quad j \in N^d \end{cases} \tag{8}$$

The directional constraint is given by:

$$\psi_i = K(\alpha - \beta_i) \tag{9}$$

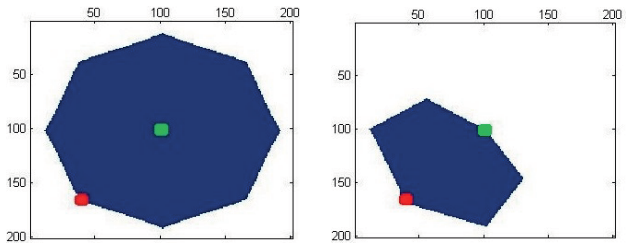
where α is the principle angle of the robot from the target which is calculated by assuming a coordinate system with the origin at the target. β_i is the angle of the i^{th} neuron from the target. The angle calculations are illustrated in Fig. 2 and the orientation of this axis is predefined and all angle computations shall be carried out with reference to this orientation. K is given as:

$$K = p\delta \tag{10}$$

Here p is the proportionality constant, δ is the difference in the firing rate of neurons at time step t and $t+1$. A decrease in firing rate indicates obstruction in front of wavefront, which would consequently decrease K thereby opening the wavefront.

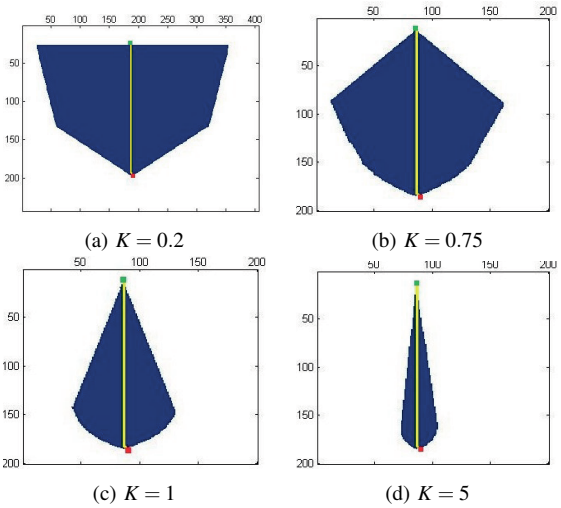
The output $Y_i(t)$ of a neuron is given by:

$$Y_i(t) = Step(U_i(t) - \theta_i(t, \lambda)) = \begin{cases} 1 & \text{if} \quad U_i(t) \geq \theta_i \\ 0 & \text{otherwise} \end{cases} \tag{11}$$



(a) MPCNN (b) GAPCNN

Fig. 3: Search space.



(a) $K = 0.2$ (b) $K = 0.75$

(c) $K = 1$ (d) $K = 5$

Fig. 4: Closing of autowave face with different values of K.

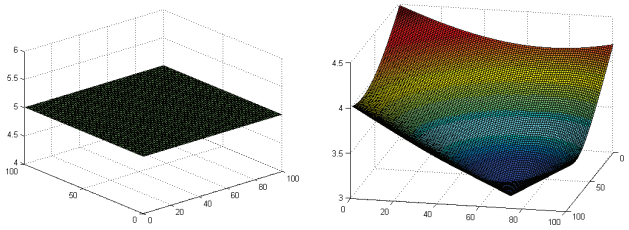
IV. MPCNN vs. GAPCNN

GAPCNN is designed to avoid unnecessary propagation of the autowave by imparting a directional behavior and also to boost up the directed neuron's energy in order to accelerate neuron firing.

A. Wave Propagation Control

The search space of the original MPCNN and the GAPCNN is depicted in Fig. 3. In all the subsequent figures the blue region represents the search space of the algorithms, green shows the target neuron, red is the robot neuron while yellow represents the path planned by the scheme. The MPCNN wave propagates equally in all directions regardless of the location of the robot while the GAPCNN wavefront only propagates in a small region around the target reducing the number of neurons that have fired (blue area). Two parameters K and A are introduced in the modified PCNN model. K controls the opening and closing of the face of guided autowave. Fig. 4 demonstrates the influence of parameter K on the wave behavior. The face of the autowave opens with lower values of K and closes as K increases.

The magnitude of K is controlled based on the difference in the firing rate of neurons at each time step i.e. upon a decrease in the firing rate in successive iterations, the value of K increases, causing the wavefront to expand.



(a) MPCNN (b) GAPCNN

Fig. 5: Threshold for grid of MPCNN and GAPCNN.

Thus, whenever the directed wave gets stuck against an obstacle, the wavefront expands, allowing the autowave to go around the obstacle. This guarantees superior efficiency of GAPCNN over MPCNN even in scenarios where no relatively straight collision free path exists. Also when the rate of firing diminishes to zero, this will indicate that there is no route between the target and the robot thereby ensuring completeness of the approach.

B. Dynamic Threshold Control

In GAPCNN each neuron of the grid possesses a unique threshold in accordance to its location whereas in MPCNN each neuron was associated with the same threshold level. It can be seen that while the MPCNN Fig. 5(a) has a constant threshold for all the neurons, GAPCNN threshold is dynamic and it significantly decreases as we approach the neurons in the vicinity of the robot neuron Fig. 5(b).

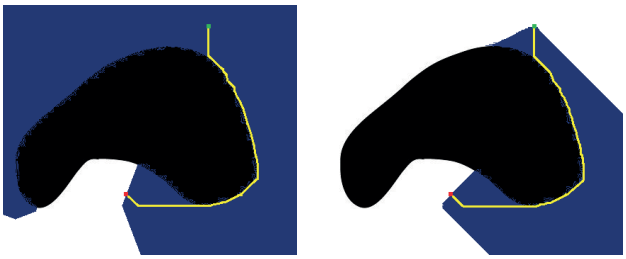
Dynamic thresholding improves the computational efficiency by allowing the neurons to fire earlier as compared to MPCNN. On the other hand it foils the possibility of false firing by providing suitable threshold level to all the neurons. In proposed model parameter A impacts the propagation speed of the autowave. Larger values in A decrease the wave propagation time as given in Table. I.

TABLE I: Execution time vs. parameter A.

Value of Parameter A	Execution time [s]
0.01	1.0
0.5	0.85
1.0	0.71
3.0	0.59
6.0	0.28

C. Performance Comparison

In order to gauge the performance improvements of GAPCNN in comparison with MPCNN, path query was performed in various situations. For the case of local minima (Fig. 6(a) and (b)), the proposed scheme successfully finds the path while still achieving a faster execution than the MPCNN. Note that the search space is almost halved. Graph shown in Fig. 7 presents the query times of the two algorithms for 200 test environments with randomly generated obstacle placements. The mean execution time of MPCNN for the results given in Fig. 7 is 11.82 milliseconds whereas



(a) MPCNN (b) GAPCNN

Fig. 6: Path planning in the presence of a local minima.

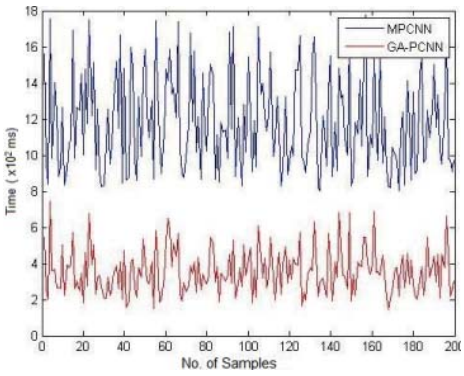


Fig. 7: Query times of MPCNN and GAPCNN.

that of GAPCNN is 3.57 milliseconds. The mean time improvement here is 8.25 milliseconds which corresponds to a 70% time improvement. From Fig. 7 it is clear that in all of the path queries, GAPCNN performs significantly better than MPCNN in terms of time taken to determine the optimal path.

V. Results

Extensive testing of the algorithm has demonstrated its suitability for path planning in various real world scenarios. Here we present some of the results showing path planning using GAPCNN in both static as well as dynamic environments.

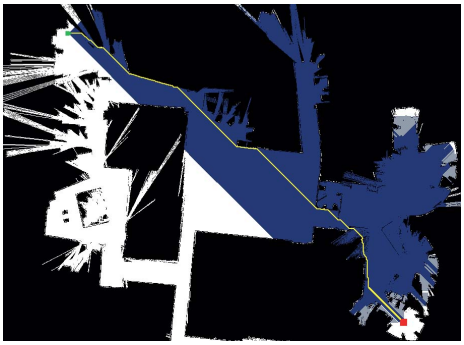


Fig. 8: A path planned in a 2D SLAM environment.

A. Simulation Results

2D SLAM maps generated in different real world environments were used to test the performance of proposed

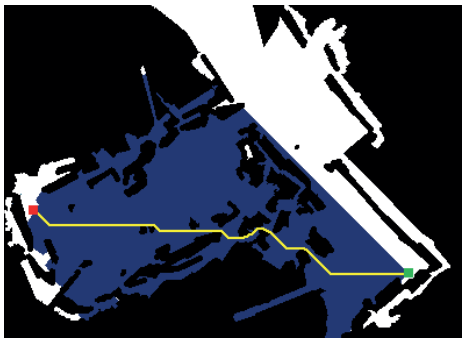


Fig. 9: Another path planned in a 2D SLAM environment.

algorithm. Figs. 8 and 9 present the paths planned in two different environments by GAPCNN. It can be observed in these figures that GAPCNN does not search the complete environment but follows a directed search pattern.

To ascertain the performance of the proposed method in dynamic world environments, simulations were carried out using Player/Stage [25]. In Figs. 10 to 12 the bottom figures show the path planned for the robot by GAPCNN at the present time step given the current obstacle proximity and is drawn in green color. Black shows the obstacles whereas the robot is drawn in red. The corresponding figures at the top present the trajectories actually followed by the dynamic obstacles and the robot in the Player/Stage simulation uptil that time step. This includes replanned robot trajectory using GAPCNN in order to accommodate the dynamically altering obstacle arrangement. In Player/Stage figures there are two dynamic obstacles, one with a blue trajectory (obstacle 'A') and the other with a green trajectory (obstacle 'B') while red shows the robot and its trajectory as determined using the GAPCNN-based intelligent path planning scheme.

Fig. 10 presents a test case in which obstacle 'A' is on a head-on collision course with the path planning robot. A straight path exists from robot to target but its obstructed by this obstacle. Fig. 10(a) shows initial path planned to avoid head on collision with obstacle 'A'. Curvilinear path taken by robot 'R' in Fig. 10 (b and c)shows the avoidance behavior of the proposed method. Next the robot is intercepted by obstacle 'B' and it evades collision by rerouting the path to the left as shown in Fig. 10 (d). Fig. 10 (e) the complete trajectories of robot and obstacles is presented.

Fig. 11 simulates a similar situation with an additional static obstacle. Fig. 11 (a and b) shows avoidance of possible collision with obstacle 'A'. Fig. 11 (c) presents how collision with static obstacle is avoided. In Fig. 11 (d), the robot determines that the optimal path is in front of obstacle 'B', and by moving in a curvilinear pattern it avoids obstacle 'B'

Fig. 12 shows another generally encounter path planning problem in which the target is dynamic. Here target is moving in a sinusoidal manner. Robot moves towards right as long as the target is moving right as shown in Fig. 12 (a, b and c) and drastically readjusts its direction towards left as target start moving left(Fig. 12 (d)). Fig. 12 (e) shows the robot following the target until it intercepts it.

B. Experimental Results

In order to validate the modifications carried out in the existing model of MPCNN, experimental evaluation was carried out using a P3AT mobile robot with vision based environment perception system. Microsoft Kinect was used to provide life feed of the robot workspace. Each object in the environment is identified by the color of its marker(color circles possessed by each object).

In the first stage experiments were carried out in a static environment setting where three obstacle are placed in the environment. The robot follows the planned path initially but soon it deviates due its dynamic constraints. The path is continuously replanned until the robot reaches its goal as shown in Fig. 13. This experiment was repeated thrice in identical setting and compared with the simulation result of this exact scenario as shown in Fig.15(a).The results obtained in the experiment are close to simulation whereas random variations in the robot dynamics slightly deviates experimental curves from the simulation curve.

Likewise an experiment was conducted in a dynamic environment, with three static and one dynamic obstacle as shown in Fig. 14. As given in Fig. 14 the path planner continuously reroute its path in accordance with the position of the dynamic obstacle in order to avoid it. Repeatedly similar robot trajectories were achieved as shown in Fig. 15(b) with comparably small variation from the simulation trajectory. Deviation from the simulation trajectory is partly because of low update frequency of robot control parameters for experimental cases as compared to that of the simulations and partly because of the stochastic dynamics of the real robot as compared to the simulation robot.

It is important to note that the parameters used in the GAPCNN model are independent of the configuration space and a property of an individual neuron. Same parameters shall work irrespective of the environment in which path is being planned.

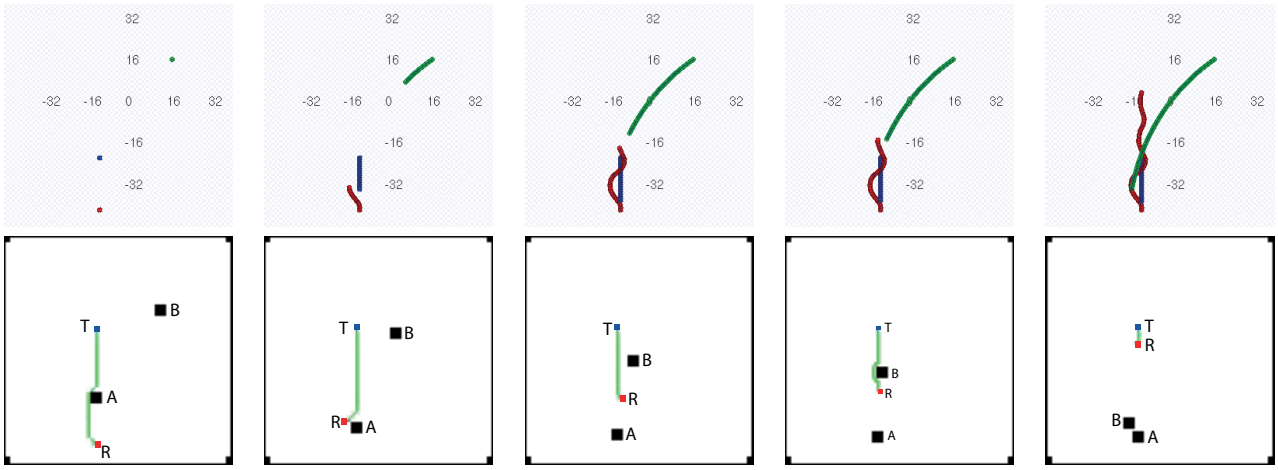
VI. Conclusion

A novel approach based on a modified PCNN is presented for real time path planning and obstacle avoidance of mobile robots. Guided and adaptive behavior of the wave has resulted in significant improvement in path query time. Time efficiency of the algorithm proves that it can be used for path planning of static as well as dynamic environments. Simulation and experimental results have shown that the GAPCNN has a significant time improvement from that of the MPCNN while retaining all of its capabilities.

Research is being carried out to extend the algorithm to include environments with uncertain robot, target and barrier locations. In addition to this dynamics constraints shall be introduced to the detected paths, thereby facilitating the motion of mobile machine.

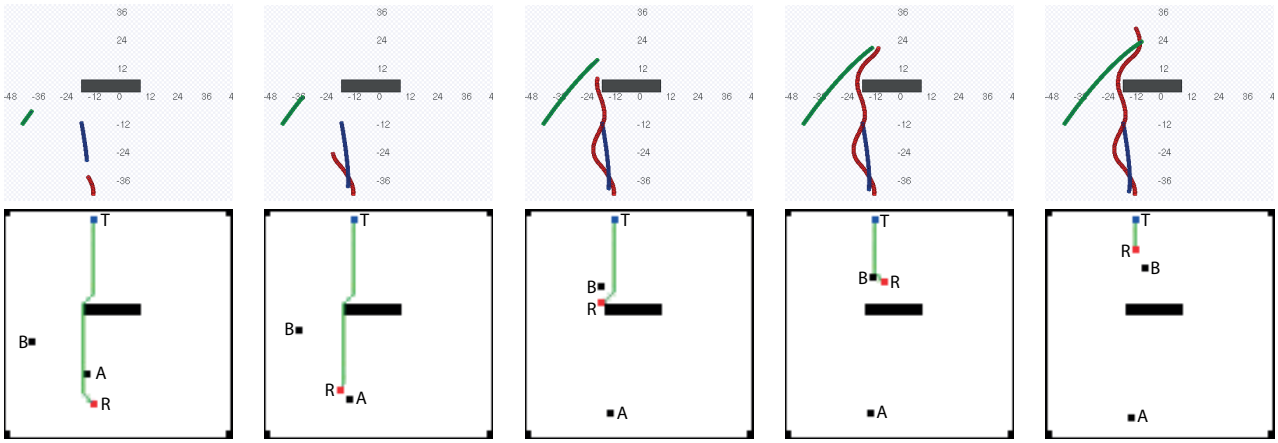
References

[1] J. Janet, R. Luo, and M. Kay, "The essential visibility graph: an approach to global motion planning for autonomous mobile robots," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 2, pp. 1958 –1963 vol.2, may 1995.



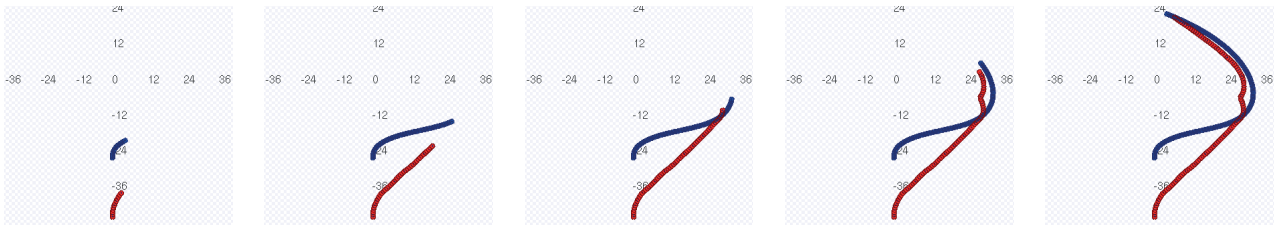
(a) (b) (c) (d) (e)

Fig. 10: Dynamic environment with two straight moving obstacles.



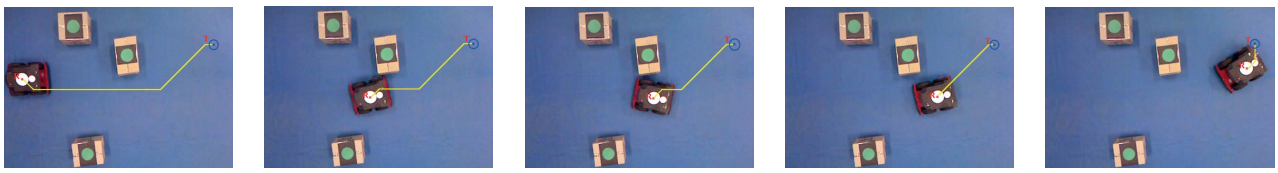
(a) (b) (c) (d) (e)

Fig. 11: Dynamic environment with two straight moving and one static obstacles.



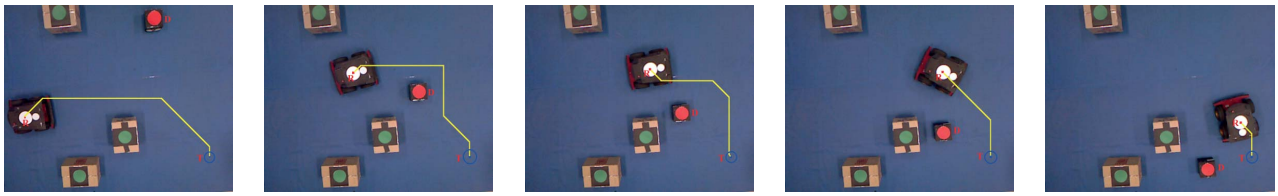
(a) (b) (c) (d) (e)

Fig. 12: Target following robot with target moving in a sinusoidal manner .



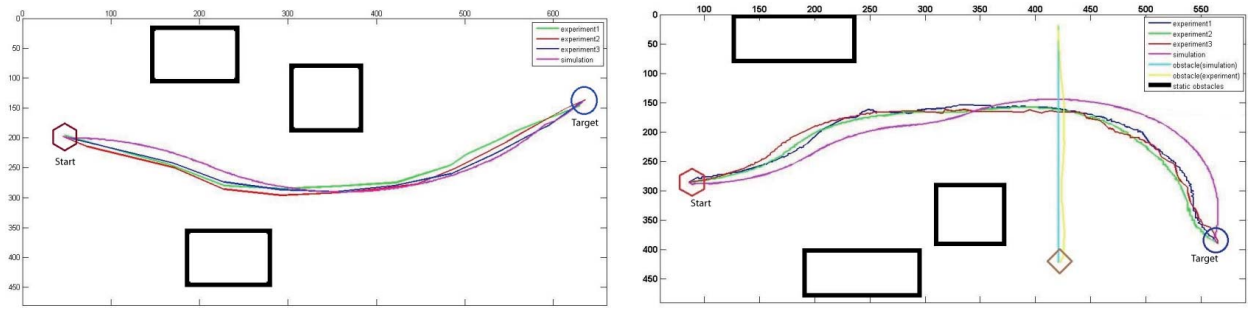
(a) (b) (c) (d) (e)

Fig. 13: Experimental results of GAPCNN path planning in a static environment.



(a) (b) (c) (d) (e)

Fig. 14: Experimental results of GAPCNN path planning in dynamic environment.



(a) (b)

Fig. 15: Comparison of the robot trajectories

[2] H.-P. Huang and S.-Y. Chung, "Dynamic visibility graph for path planning," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 3, pp. 2813 – 2818 vol.3, sept.-2 oct. 2004.

[3] F. Lingelbach, "Path planning for mobile manipulation using probabilistic cell decomposition," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 3, pp. 2807 – 2812 vol.3, sept.-2 oct. 2004.

[4] O. Takahashi and R. Schilling, "Motion planning in a plane using generalized voronoi diagrams," *Robotics and Automation, IEEE Transactions on*, vol. 5, pp. 143 –150, apr 1989.

[5] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, vol. 2, pp. 500 – 505, mar 1985.

[6] J. Andrews, *Impedance Control as a Framework for Implementing Obstacle Avoidance in a Manipulator*. Massachusetts Institute of Technology, Department of Mechanical Engineering, 1983.

[7] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on*, vol. 12, pp. 566 – 580, aug 1996.

[8] M. Kalisiak and M. van de Panne, "Rrt-blossom: Rrt with a local flood-fill behavior," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 1237 –1242, may 2006.

[9] J. Kuffner, J.J. and S. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 2, pp. 995 –1001 vol.2, 2000.

[10] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[11] J. Yao, C. Lin, X. Xie, A. Wang, and C.-C. Hung, "Path planning for virtual human motion using improved a* star algorithm," in *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on*, pp. 1154 –1158, april 2010.

[12] Y. Hu and S. Yang, "A knowledge based genetic algorithm for path planning of a mobile robot," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 5, pp. 4350 – 4355 Vol.5, april-1 may 2004.

[13] S. C. Yun, V. Ganapathy, and L. O. Chong, "Improved genetic algorithms based optimum path planning for mobile robot," in *Control Automation Robotics Vision (ICARCV), 2010 11th International Conference on*, pp. 1565 –1570, dec. 2010.

[14] X. Yang, M. Moallem, and R. Patel, "A layered goal-oriented fuzzy motion planning strategy for mobile robot navigation," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 35, pp. 1214 –1224, dec. 2005.

[15] P. Ro and B. Lee, "Neural-fuzzy hybrid system for mobile robot path-planning in a partially known environment," in *American Control Conference, 1995. Proceedings of the*, vol. 1, pp. 673 –677 vol.1, jun 1995.

[16] R. Glasius, A. Komoda, and S. Gielen, "Population coding in a neural net for trajectory formation," *Network, Computation and Neural Systems*, vol. 5, pp. 549–563, 1994.

[17] I. Gavrilut, A. Gacsadi, C. Grava, and V. Tiponut, "Vision based algorithm for path planning of a mobile robot by using cellular neural networks," in *Automation, Quality and Testing, Robotics, 2006 IEEE International Conference on*, vol. 2, pp. 306 –311, may 2006.

[18] F. Jian, F. MinRui, and M. ShiWei, "Rl-art2 neural network based mobile robot path planning," in *Intelligent Systems Design and Applications, 2006. ISDA '06. Sixth International Conference on*, vol. 2, pp. 581 –585, oct. 2006.

[19] R. Eckhorn, H. J. Reitboeck, M. Arndt, and P. Dicke, "Feature linking via synchronization among distributed assemblies: Simulations of results from cat visual cortex," *Neural Comput.*, vol. 2, pp. 293–307, Sept. 1990.

[20] J. L. Johnson and D. Ritter, "Observation of periodic waves in a pulse-coupled neural network," *Opt. Lett.*, vol. 18, pp. 1253–1255, Aug 1993.

[21] J. L. Johnson, "Pulse-coupled neural nets: translation, rotation, scale, distortion, and intensity signal invariance for images," *Appl. Opt.*, vol. 33, pp. 6239–6253, Sep 1994.

[22] T. Lindblad and J. Kinser, *Image Processing Using Pulse-Coupled Neural Networks*. Perspectives in neural computing, Springer, 2005.

[23] H. Caulfield and J. Kinser, "Finding the shortest path in the shortest time using pcnn's," *Neural Networks, IEEE Transactions on*, vol. 10, pp. 604 –606, may 1999.

[24] H. Qu, S. Yang, A. Willms, and Z. Yi, "Real-time robot path planning based on a modified pulse-coupled neural network model," *Neural Networks, IEEE Transactions on*, vol. 20, pp. 1724 –1739, nov. 2009.

[25] B. P. Gerkey, R. T. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in *In Proceedings of the 11th International Conference on Advanced Robotics*, pp. 317–323, 2003.