Navigating Mobile Robots to Target in Near Shortest Time using Reinforcement Learning with Spiking Neural Networks

Amarnath Mahadevuni
Department of Electrical and Computer Engineering
Texas A&M University
College Station, Texas 77843, USA
Email: amarnathmhn@tamu.edu

Peng Li
Department of Electrical and Computer Engineering
Texas A&M University
College Station, Texas 77843, USA
Email: pli@tamu.edu

***Abstract*—The autonomous navigation of mobile robots in unknown environments is of great interest in mobile robotics. This article discusses a new strategy to navigate to a known target location in an unknown environment using a combination of the "go-to-goal" approach and reinforcement learning with biologically realistic spiking neural networks. While the "go-to-goal" approach itself might lead to a solution for most environments, the added neural reinforcement learning in this work results in a strategy that takes the robot from a starting position to a target location in a near shortest possible time. To achieve the goal, we propose a reinforcement learning approach based on spiking neural networks. The presented biologically motivated delayed reward mechanism using eligibility traces results in a greedy approach that leads the robot to the target in a close to shortest possible time.**

I. Introduction

The research about autonomous navigation of unmanned vehicles in known and unknown environments is growing rapidly due to the advent in self-driving cars, extra terrestrial exploration, rescue missions and similar areas. In this paper, the problem autonomous navigation of mobile robot towards a given goal location in environment containing obstacles is considered. An example instance of the problem is shown in Figure 1.

In Figure 1 the mobile robot is the circular object "Robot". The lines emanating out of the robot's center are ultrasonic sensor beams to detect any obstacles. The ultrasonic sensors have a limited range of detection. The rectangular objects are wall-like obstacles that can be sensed by the ultrasonic sensors. The point labeled "Start" is the initial position of the robot. The square shaped object in the figure labeled "Goal" is the global target to which the robot has to navigate. It is assumed that the robot has a way to detect the position of target with respect to itself. The shape and location of the obstacles is unknown to the robot which makes this a challenging problem.

A. Previous Work

The existing solutions to similar problems have been proposed using various approaches. [1] presents the Artificial Potential Field Approach which essentially treats the obstacles as repelling the robot, while the target attracts the robot. [2] and [3] provide strategies for the robot to escape the problem of the robot getting stuck in local minima in [1]. These approaches assume that the robot has knowledge of the shapes and locations of obstacles. Our version of the problem assumes no such information.

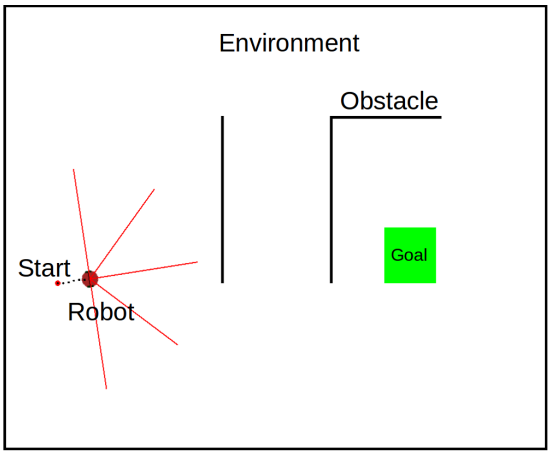


Fig. 1: Example instance of the problem

Reinforcement learning has also been applied in variety of ways to the mobile robot navigation problem. For example In [7], Q-learning [8], a form of Temporal Difference Reinforcement learning algorithm is used to learn a collision free path in an unknown environment without any goal. Other works include [9], [10] which use some form of Q-learning to learn to avoid the obstacles. In all of these works, the aim was to learn to navigate through obstacles by avoiding collision. There is no target in the picture.

B. Overview of the Proposed Solution

We first discuss a navigation strategy presented in [11] and then discuss our reinforcement learning algorithm that improves this strategy.

In the "go-to-goal" approach [11], the mobile robot navigates towards the goal while there are no obstacles around.

When faced with an obstacle, it turns (left or right) in place and follows the obstacle using a wall following controller. The robot leaves the wall following mode upon satisfying a certain condition discussed in upcoming sections. Then it moves towards goal. Figure 2. illustrates this solution for a simple environment.

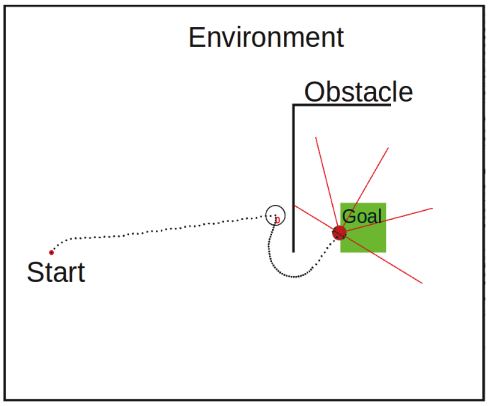


Fig. 2: Example instance of the solution proposed in [11]

To navigate to the target in the shortest time under this strategy, the robot must make intelligent turning decisions when faced with an obstacle. For instance, in Figure 2, if the robot had taken a left turn at the circled location, it would have taken a longer time to reach the target.

To facilitate this decision making process, we propose a biologically motivated reinforcement learning strategy using a spiking neural network to make the turn decisions. The set of states for the proposed reinforcement learning algorithm are the locations where the robot makes a turning decision. The actions for the robot at each state are: 1) turn left and follow the wall on its right , and 2) turn right and follow wall on its left. Our algorithm utilizes state-action values similar to Q-values in Q-learning, to select the action in each state.

The learning algorithm is implemented using a spiking neural network [12]. The states of the robot are represented with an input layer using the state encoding approach used in [18] and discussed in detail in Section III. The output layer has two neurons for the two robot turning actions. The firing rates of the output neurons at each state are considered as the state-action values.

At each state, the robot selects an action by simulating our neural network. The synaptic connections of the network generate a decaying eligibility trace for each of the state-action pair. The robot is given a reward upon reaching the goal which is multiplied with the eligibility trace and added to the synaptic weights which eventually ensures high state-action value for all actions that lead the robot to the target in near shortest possible time.

The details of our neural learning mechanism are discussed in Section II and Section III.

II. Background

In reinforcement learning, an agent learns by discovering actions that maximize the total expected reward it receives. In the reinforcement learning algorithm Q-learning, the agent's state-action value function i.e. the Q value is updated after each state transition. So upon reaching a state, only the Q value for the previous state is updated. However as it happens in many problems including the navigation problem, the agent often takes actions whose rewards are delayed. This problem is known as the "credit assignment problem" or the "delayed reward problem" [4].

To solve the "delayed reward problem", Izhikevich [13], proposed a mechanism called Dopamine modulated Hebbian Spike Timing Dependent Plasticity (STDP) with eligibility trace. This is explained briefly in the following subsections.

A. Hebbian STDP

Consider a simple spiking neural network with a single synapse connecting a presynaptic neuron i to a postsynaptic neuron j as shown in Figure 3A. Let the synaptic strength be denoted by W_{ij} . If t_{post} is any spike time of postsynaptic neuron, t_{pre} is any spike time of presynaptic neuron, then the synaptic weight change ΔW_{ij} happens according to Hebbian STDP [14] as shown below :

$$\Delta W_{ij} = A_+ exp(-\frac{\Delta t}{\tau_+}) \text{ for } \Delta t > 0 \quad (1)$$

$$\Delta W_{ij} = A_- exp(-\frac{\Delta t}{\tau_-}) \text{ for } \Delta t <= 0, \quad (2)$$

where $\Delta t = t_{post} - t_{pre}$. A_+, A_-, τ_+, τ_- are design parameters. A graphical representation of the STDP is shown in Fig 3B.

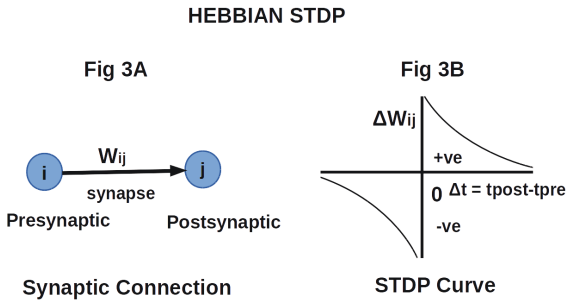


Fig. 3: Synaptic Connection and STDP curve (Redrawn from [13])

In any period of time, there are multiple presynaptic and postsynaptic spikes for synapse. So it is not computationally efficient to consider all possible pairs of presynaptic and postsynaptic spike times to calculate synaptic weight changes. Therefore they use the nearest neighbour STDP [13] where only one presynaptic spike time is considered for every post-synaptic spike time.

B. Dopamine Modulated STDP

Reinforcement learning in spiking neural networks is thought to happen via a mechanism called Dopamine Modulated STDP [13]. Synaptic weight changes are enhanced in the presence of Dopamine neuromodulator which is associated with reward in reinforcement learning. We treat the reward as taking both positive and negative values for reward and punishment respectively as is done in [15]. The synaptic strength is changed by the amount defined by the STDP change decayed as the eligibility trace multiplied(modulated) with the scalar reward [13]. Figure 4A. shows this idea for a presynaptic and postsynaptic spike pair. We can see that the eligibility trace in Fig 4B. is updated on a pre-before-post spike pattern and is decayed as time passes. When a reward is applied, it gets multiplied to the eligibility trace and the weight is updated. This is shown in equation below:

$$\Delta W_{ij} = \Delta W_{stdp}.reward \tag{3}$$

At every time step, the STDP weight change is decayed.

$$\Delta W_{stdp} = \Delta W_{stdp}.\beta \tag{4}$$

where ΔW_{stdp} is calculated using (1) and (2) after each simulation for 1000 ms of our network, β is the eligibility trace decay parameter, and $reward$ is the numerical reward applied.

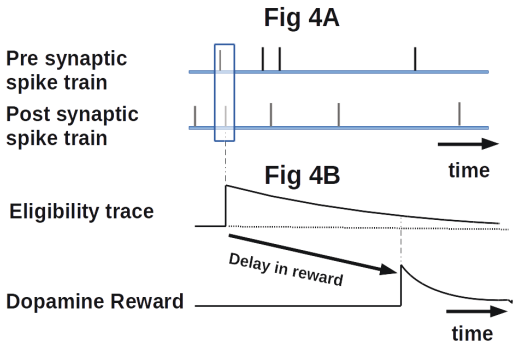


Fig. 4: Illustration of Eligibility Trace. (Redrawn from [13])

We use this dopamine modulated STDP in our spiking neural network to make the mobile robot turn decisions previously discussed in section 1.B.

III. Reinforcement learning using Spiking Neural Network

The solution to the navigation problem proposed in this work involves a learning part where the robot makes a decision of either 1) follow the obstacle with obstacle on its left side by taking a right turn or 2) follow the obstacle with obstacle on its right side by taking a left turn when faced directly with an obstacle. This decision making scenario is shown in Figure 5. The robot makes the wall following decision at the areas marked with circles (labeled '0', '1', and '2' in Figure 5).

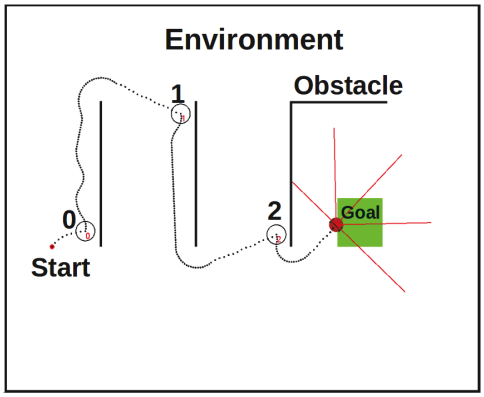


Fig. 5: Navigation of the robot showing the points (circled) on the path where spiking neural network is run to decide which side of the wall to follow. The dotted path is the trail of the trajectory taken by the robot.

The spiking neural network designed is shown in Figure 6. It is a two layered network: one input layer of 88 excitatory Poisson spiking neurons and one output layer with 2 excitatory spiking neurons. The input layer is connected to output layer in a fully connected fashion. The input neurons are numbered from 0 through 87 and output neurons are labeled LEFT and RIGHT. The state for the reinforcement learning problem is represented by the input layer. This is accomplished by calculating the input firing rates in such a way that for each state, a distinct population of input neurons fire. This encoding of the states has been done previously in [18]. For each input neuron i, a frequency f_i is used to generate the Poisson spike train. f_i is calculated as follows:

$$f_i = f_0 exp\left(-\frac{(d-d_i)^2}{2\sigma_1^2} - \frac{(\theta-\theta_i)^2}{2\sigma_2^2}\right), \tag{5}$$

where θ is the orientation of the robot heading direction with respect to a fixed axis measured in degrees and d is the distance of the center of robot to the target location (center of the square shaped target) measured in pixels. Calculaton of these two parameters is shown in Figure 7. For θ, a fixed axis pointing upwards in the plane of simulation environment is used as reference and is measured in the clockwise direction. θ can take values from 0° through 360° .

The parameters d_i and θ_i are such that $(d_i, \theta_i) \in D \times \Theta$, where

$$D = \{100, 200, 300, 400, 500, 600, 700, 800\} \tag{6}$$

$$\Theta = \{30^0, 60^0, ..., 300^0, 330^0\} \tag{7}$$

In other words, D is the set of discretized distances from the robot agent to the target, and Θ is the set of discretized orientations of the robot heading direction. The total number of possible pairs (d_i, θ_i) are therefore $n(D)n(\Theta) = (8)(11) = 88$, which determines the number of neurons in the input layer of network in Figure 6. Here $n(S)$ denotes the cardinality of a set S. σ_1 and σ_2 are design parameters that determine the spread of the curve f_i v/s (d, θ) .

Each state in the reinforcement learning problem is thus represented by the ordered pair (d, θ) and the input rates f_i calculated are such that distinct set of input neurons are excited for distinct ordered pairs (d, θ). For example in Figure 5, the circles labeled '0', '1', and '2' are locations where the d and θ are measured and f_i is calculated for each of the 88 input neurons. The population of neurons fired are different for the points '0', '1', and '2'. To visualize this firing behavior, a plot of firing rates vs input neuron number is shown in Figure 8 for each of the states '0', '1', and '2' of Figure 5.

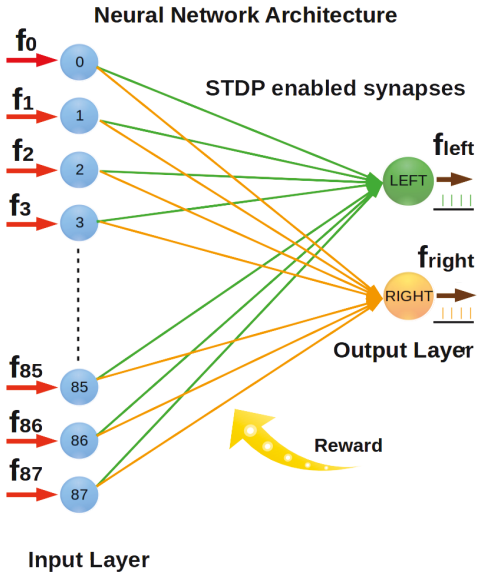


Fig. 6: The spiking neural network architecture used for reinforcement learning part of the solution.

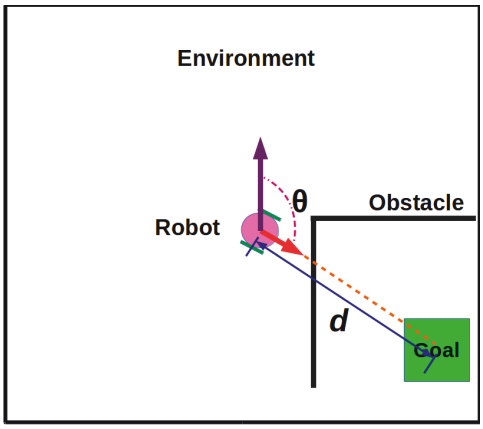


Fig. 7: Calculations for distance to the target d and robot orientation θ . The upward facing arrow is a fixed axis and the shorter arrow is the robot heading direction

When the neural network is run, its output layer produces two output firing rates: f_{left} and f_{right} corresponding to output neurons labeled 'LEFT' and 'RIGHT' in Figure 6. When making the wall follow decision, if f_{left} is greater than f_{right}, then the robot decides to follow the obstacle on its LEFT by making a right turn. Otherwise the robot decides to follow the obstacle on its RIGHT by making a left turn.

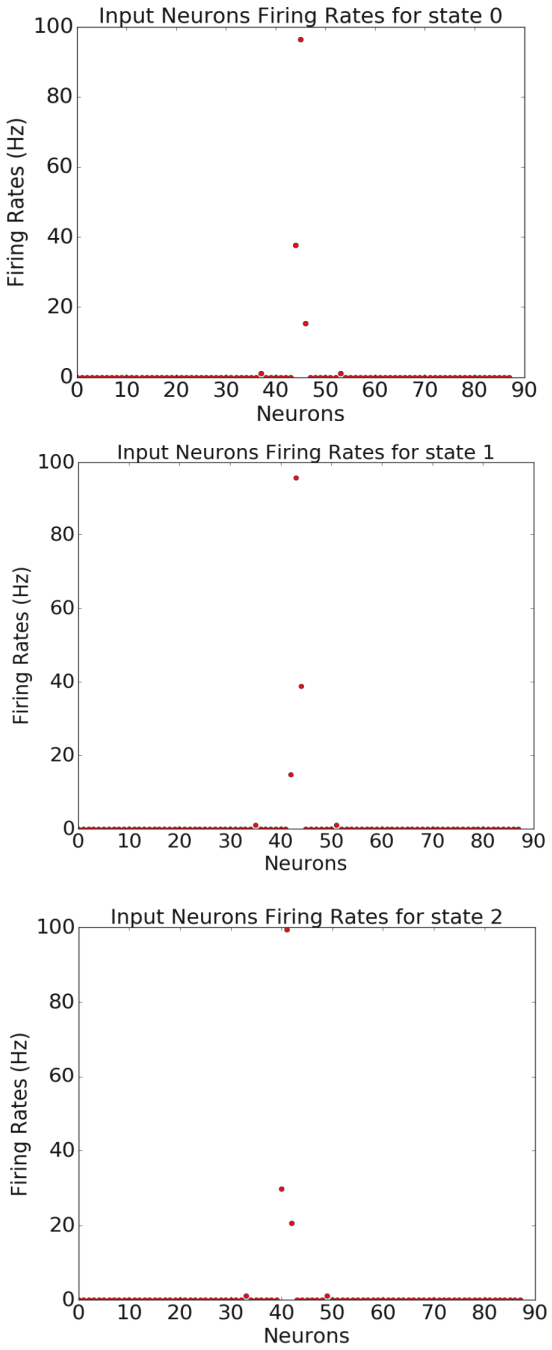


Fig. 8: Firing Rate v/s Input layer neuron number for state 0, 1, and 2 of Figure 5.

IV. The Complete Proposed Solution

In this section we will combine the "go-to-goal" approach [11] and the reinforcement learning with spiking neural network to make the wall following decisions. The "go-to-goal" like solution is also used in work presented in [19].

Figure 9. shows the robot abstraction used for the solution. It is a simple differential drive robot equipped with 5 ultrasonic sensors with $45°$ angle between them. They are designated as US_LEFT, US_LEFT45, US_FRONT, US_RIGHT45, US_RIGHT as shown in Figure 9. The robot is also assumed to have a sensor that determines its orientation with respect to a fixed axis and also assumed that it can sense its orientation with respect to and distance to the target.

The proposed navigation algorithm is designed as a state machine with 5 states which we referred to as "Navigation Modes" namely *PRE-GO-TO-GOAL, GO-TO-GOAL, PRE-WALL-FOLLOW, WALL-FOLLOW* and *TARGET-REACHED*. The robot behavior in these states is discussed in the following detailed algorithm.

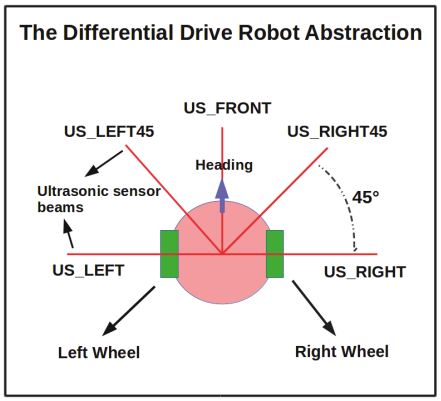


Fig. 9: The differential drive robot model used in the proposed solution.

A. Detailed Algorithm

The robot is in *PRE-GO-TO-GOAL* mode at the start of each trial.

1) *PRE-GO-TO-GOAL* :
   a) The robot makes turns so that its orientation is towards the target.
   b) Once the robot is done orienting itself towards target, it changes its navigation mode to *GO-TO-GOAL*.
2) *GO-TO-GOAL* :
   a) The robot moves 10 pixels every time step towards the target.
   b) If the sensor readings from US_LEFT45 or US_FRONT or US_RIGHT45 indicate a proximity of less than 40 pixels, then change the mode to *PRE-WALL-FOLLOW*.
   c) If the robot reaches the target, it transitions to *TARGET-REACHED* mode.
3) *PRE-WALL-FOLLOW*:
   In this mode, the robot runs the spiking neural network to decide which side of the wall to follow on.
   a) The robot calculates the distance to the target d and its orientation θ as shown in Figure 7. The firing rates f_i are calculated for each input Poisson spiking neuron.
   b) The network is simulated for 1000 ms and the output firing rates f_{left} and f_{right} are calculated. Running the neural network updates eligibility traces on the synapses from input layer to LEFT and RIGHT output neurons as discussed in section II B . For example, if a subset N_j of neurons are excited in a particular state $S_j = (d_j, \theta_j)$, then eligibility traces for synaptic connections from N_j to LEFT output and N_j to RIGHT output neuron are updated as shown in Figure 4.
   c) Action exploration is done with a probability of 50% (0.5 greedy). During exploitation, if $f_{left} \geq f_{right}$, the robot decides to follow the obstacle in a wall following manner with the obstacle on its left. If $f_{right} > f_{left}$, the robot decides to follow the obstacle in a wall following manner with the obstacle on its right . The two wall follow modes can be denoted by *WALL-FOLLOW-LEFT* and *WALL-FOLLOW-RIGHT*. During exploration, one of the two actions is selected randomly with a probability of 50%.
   d) Eligibility trace is recorded only for the synapses from input layer to the output neuron that has highest firing rate. That is we do not record eligibility traces for actions that are not taken. If the condition

$$\|(f_{left} - f_{right})\| \geq f_{threshold} \tag{8}$$

   is satisfied, then no eligibility trace is stored. If this condition is satisfied for all the times the robot enters PRE-WALL-FOLLOW mode in a trial, the learning algorithm is assumed to be converged and weights of the neural network are fixed at their current values.
   e) The eligibility traces for all the synapses are decayed every time step throughout the trial.
   f) After deciding which side to follow the obstacle on, the navigation mode is changed to WALL-FOLLOW.
4) *WALL-FOLLOW*:
   a) The robot moves in a wall following fashion. It follows the wall on its left or on its right side depending on the wall following decision made in *PRE-WALL-FOLLOW* mode.
   b) The robot leaves the mode if it decides that it has a clear way towards the goal. The calculations to

determine the condition to leave the wall following mode [11] are as shown in Figure 10.

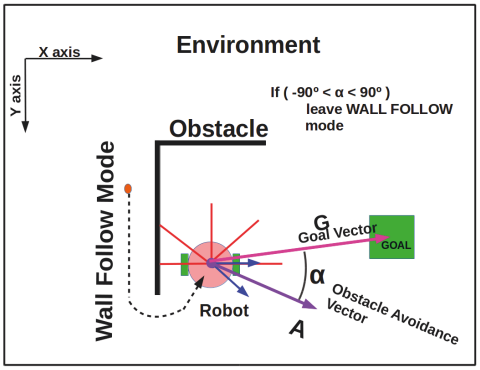


Fig. 10: Calculation of the Angle between obstacle avoidance vector and robot to target vector to determine the condition to leave wall following mode

In Figure 10, the Obstacle Avoidance Vector **A** is calculated as a resultant of vectors with directions opposite to the ultrasonic beam direction and magnitude inversely related to the proximity from the obstacle. The equations to calculate the obstacle avoidance vector are shown below:

$$\mathbf{A} = \mathbf{V}_{US\_LEFT} d(US\_LEFT) + \mathbf{V}_{US\_LEFT45} d(US\_LEFT45) + \mathbf{V}_{US\_FRONT} d(US\_FRONT) + \mathbf{V}_{US\_RIGHT45} d(US\_RIGHT45) + \mathbf{V}_{US\_RIGHT} d(US\_RIGHT) \quad (9)$$

In equation 9, the vector $\mathbf{V}_{US\_SENS}$ is a unit vector in the direction opposite to the sonar beam of the ultrasonic sensor *SENS* where *SENS* $\in$ {*LEFT, LEFT45, FRONT, FRONT45, RIGHT, RIGHT45*} . The term $d(US\_SENS)$ is defined as follows: $d(US\_SENS) = 0$ if distance measured by US_SENS is greater than 40 and $d(US\_SENS)$ = 40 - (distance measured by US_SENS) otherwise.
The goal vector **G** is the vector from the robot to the target. The condition used to decide if the robot should leave wall follow mode is :

$$|\alpha| \leq 90^{\circ}$$

where α is the angle between **G** and **A**. If the condition above is satisfied, the robot moves to the *PRE-GO-TO-GOAL* navigation mode.

5) *TARGET-REACHED*:
   a) The robot reaches its destination. The spiking neural network is now rewarded with a global scalar reward. That is the weights of synapses to the output neurons LEFT and RIGHT are changed by the amount equal to the value of eligibility trace multiplied by this scalar reward.
   b) The robot is placed in its initial position for another learning trial. All of the eligibility traces are set to zero.

Figure 11. shows all navigation modes of the robot for a simple environment.

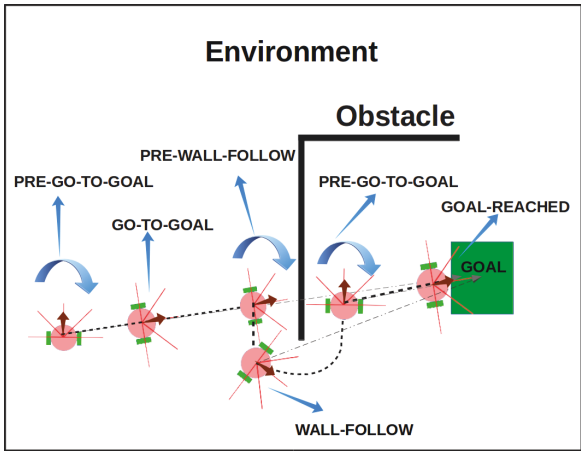


Fig. 11: All robot navigation modes in a simple environment. The curved arrows show the direction of turn of the robot.

V. Simulation Setup, Results and Discussion

The simulation environment is written in C++ using a 2-D game library Allegro *http://liballeg.org*. The simulator used for spiking neural network simulation is CARLsim [16] available at *http://www.socsci.uci.edu/ jkrichma/CARLsim*. CARLsim uses the Izhikevich model [17] of neurons which are reasonably biologically realistic yet computationally efficient.

The navigation environment is a 2-D rectangular space with dimensions 1000 pixels x 800 pixels. The obstacles considered are wall-like, rectangular in shape (unknown to the robot). The obstacle free space is shown in white color and obstacles are colored black. The target location is labeled "GOAL". The robot is circular in shape with 5 ultrasonic sensors as shown in Figure 9. Each of the 5 ultrasonic sensors has a range of 40 pixels detection distance. That is any object within 40 pixels of the ultrasonic sensor is detected. The values of parameters used in the simulation of the spiking neural network learning algorithm are shown in Table I .

The simulation is performed using 3 different environments : Env 1, Env 2, and Env 3. The snapshots of simulation before and after training are shown in Figures 12, 13, and 14.

The number of trials taken to converge are shown in Table II for each of these environments. A training session is assumed to terminate when equation (8) holds for all *PRE-WALL-FOLLOW* states the robot encounters.

To analyze the working of the algorithm, consider the eligibility traces for the actions taken at state labeled '1' in Fig. 12 "Before Training" and "After Training" Figures. These traces are plotted in Figure 15.

Figure 15 (a), shows the trace of the robot action "Follow the obstacle on the right side" taken at state 1 in Env 1. The

TABLE I: Simulation Parameters and their Values

Parameter	Description	Value
A_+	CARLsim LTP parameter for STDP in eq 1	0.0001
A_-	CARLsim LTD parameter for STDP in eq 2	0.0005
τ_+	CARLsim LTP parameter for STDP in eq 1	20ms
τ_-	CARLsim LTD parameter for STDP in eq 2	20ms
β	Eligibility trace decay in eq 4	0.99
f_0	Constant in eq 5	100 Hz
σ_1	Constant in eq 5	25 pixels
σ_2	Constant in eq 5	10°
$reward$	Scalar reward on reaching the Goal	10
$f_{threshold}$	Threshold firing rate difference in eq 8	$10Hz$

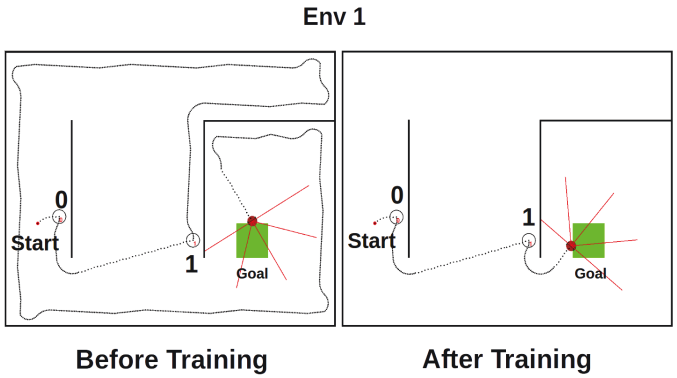


Fig. 12: Simulation results for Env 1

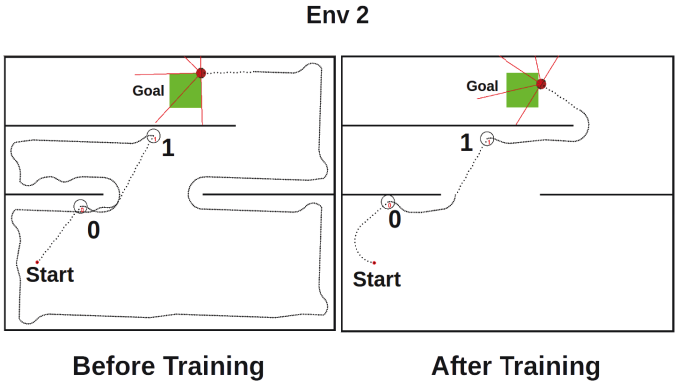


Fig. 13: Simulation results for Env 2

TABLE II: Number of Trials taken for the learning to converge

Environment	No. of Trials
Env1	14
Env2	16
Env3	44

label "1" indicates the timestep in which the robot is in state 1 and "Goal" indicates the timestep the robot reaches the target. The eligibility trace for this action is almost zero by this time. In contrast, the robot action "Follow the obstacle on the left

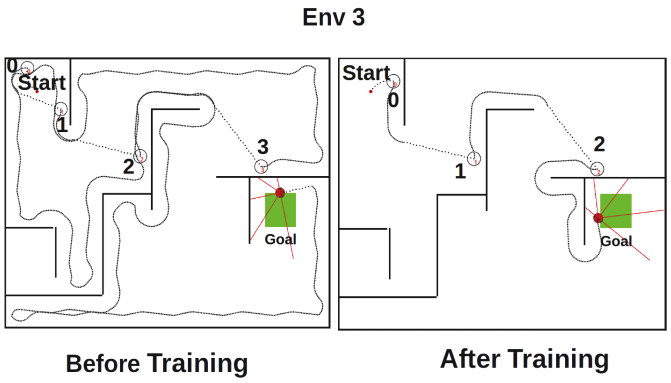
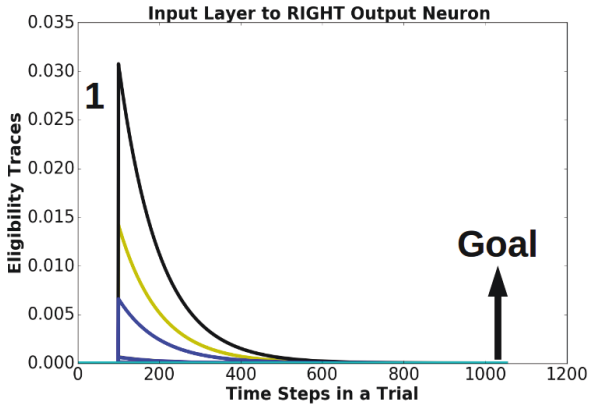


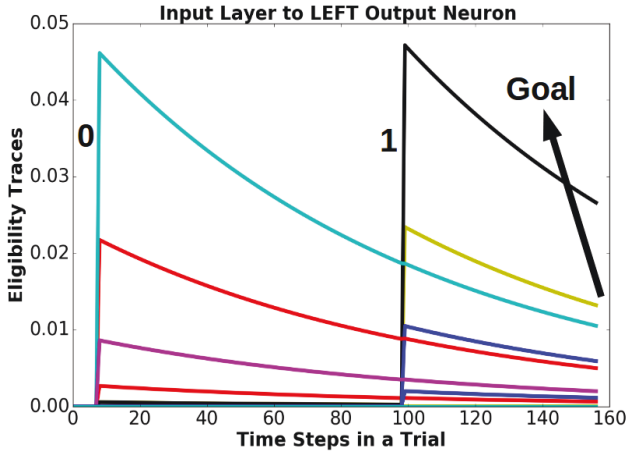
Fig. 14: Simulation results for Env 3

A nice little title



(a) Trace of the robot action: Follow the obstacle on the right side at state 1 in Env 1

A nice little title



(b) Trace of the robot action: Follow the obstacle on the left side at state 1 in Env 1

Fig. 15: Eligibility traces for input layer to RIGHT output neuron (top) and input layer to LEFT output neuron (bottom).

side" at the same state 1 in Figure 15 (b) has a trace that has a positive value for all traces produced at state 1. So in the scenario of Figure 15 (b), the synapses between input layer and LEFT output neuron are strengthened more than the synapses in scenario of Figure 15 (a), between input layer and RIGHT output neuron. This results in higher output firing rate for LEFT output neuron than the RIGHT output neuron in subsequent network simulations. Therefore the state-action value at state 1 is higher for the action "Follow the obstacle on the left side" than the action "Follow the obstacle on the right side". This happens essentially because when the robot takes turn decisions resulting in the relatively shorter path, the eligibility trace decays by a smaller amount than when the robot takes turn decisions resulting in longer time paths. Thus the robot eventually learns the actions that lead it to the target in the near shortest time.

VI. Conclusion

The go-to-goal approach to navigation problem can be enhanced using a spiking neural network based reinforcement learning solution equipped with eligibility trace mechanism. The proposed solution is demonstrated to have provided an algorithm for the robot to reach its destination in shortest possible time.

The number of trials needed to achieve convergence of the solution depends on the complexity of the environment as seen from Table II.

The limitations of the proposed solution are:

1) It does not consider complex obstacle shapes,
2) It is assumed the environment is bounded by walls,
3) Danger zones where the robot can get stuck are not considered,
4) The proposed approach does not give an absolute shortest time path as the termination condition (8) does not guarantee it.

We intend to address these limitations in our future work. The current work is to be seen as exploring the use of eligibility trace mechanism to solve the navigation problem.

VII. Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant No. CCF-1639995 and the Semiconductor Research Corporation (SRC) under Task 2692.001.

References

[1] Khatib O.,"Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", *The International Journal of Robotics Research, Vol. 5, No. 1, Spring 1986, pp. 90-99*

[2] Q. Zhu, Y. Yan, and Z. Xing, "Robot Path Planning Based on Artificial Potential Field Approach with Simulated Annealing", *The International Conference on Intelligent Systems Design and Applications, 2006*

[3] L. Zhou and W. Li, "Adaptive artificial potential field approach for obstacle avoidance path planning", *The International Conference on Intelligent Systems Design and Applications, 2006*

[4] Sutton RS, Barto AG. 1998, "Reinforcement learning: an introduction", Cambridge (MA): The MIT Press.

[5] G. Tesauro , "Temporal Difference Learning and TD-Gammon", *Communications of the ACM, 1995*

[6] S.Schaal and Christopher Atkeson. "Robot juggling: An implementation of memory-based learning" *Control Systems Magazine, 14, 1994*

[7] B. Zuo, J. Chen, L. Wang, and Y. Wang, "A Reinforcement Learning Based Robotic Navigation System" *International Conference on Systems, Man, and Cybernetics, 2014*

[8] Watkins, C.J. and P. Dayan, "Q-Learning" *Machine Learning, May 1992*

[9] G. Li, Jie Pang, "A Reinforcement Learning with Adaptive State Space Construction for Mobile Robot Navigation", *2006*

[10] Y. Yusof, H.M Asri, H. Mansor, H.M Dani Baba, "Simulation of Mobile Robot Navigation Utilizing Reinforcement and Unsupervised Weightless Neural Network Learning Algorithm", *2015*

[11] M. Egerstedt, "Controls for the Masses", *IEEE Control Systems Magazine, Vol. 33, No. 4, pp. 40-44, Aug. 2013* and Coursera lectures on "Control of Mobile Robots", *https://www.coursera.org/*

[12] Maas, Wolfgang(1996), "Networks of Spiking Neurons: The Third Generation of Neural Network Models"

[13] Eugene M. Izhikevich, "Solving the Distal Reward Problem through Linkage of STDP and Dopamine Signaling", *2007*

[14] J. Sjostrom and W. Gerstner, "Spike-timing dependent plasticity", *Scholarpedia 2010*

[15] Razvan V. Florian, "Reinforcement Learning Through Modulation of Spike-Timing-Dependent Synaptic Plasticity", *Neural Computation 2007*

[16] Beyeler, M.*, Carlson, K.D.*, Chou, T.-S.*, Dutt, N., and Krichmar, J.L. (2015). A User-Friendly and Highly Optimized Library for the Creation of Neurobiologically Detailed Spiking Neural Networks. *International Joint Conference on Neural Networks (Killarney, Ireland)(CARLsim v3.0)*

[17] Eugene M. Izhkevich, "Simple Model of Spiking Neurons", *IEEE transactions on Neural Networks (2003)*

[18] N. Fremaux, H. Sprekeler, W. Gerstner, "Reinforcement Learning Using a Continuous Time Actor-Critic Framework with Spiking Neurons", *PLOS Computational Biology (April 2013)*

[19] X. Wang, Z.G. Hou, Lv Feng, M. Tan, Y. Wang, "Mobile robots' modular navigation controller using spiking neural networks", *Neurocomputing (June 2014)*