Effect of Localization on the prediction of the action of the robot

Mr.Sameer Shastri, Ritesh Dandotia

Abstract— Learning paradigms offer a great deal of insightful work that may lead the robots closure to human. Reinforcement learning is a learning paradigm to explore tactics that are feasible and to acquire a policy that is near – optimal, it exploits reward experiences that are already known. This work would explore the affect of localization value in input to the Self organinzing Map based Q learning for the prediction of the action of the robot. The error difference of the two cases will be compared to see effect of localization value of the robot.

Index Terms— Action, Learning Paradigms, Localization, SOMQ.

1 INTRODUCTION

The process through which intelligent systems acquire knowledge is called Learning. The ability of intelligent systems to self-learn is a must for the accumulation of knowledge and improvement of its performance. Self-exploration is important in case of intelligent systems. It makes an intelligent agent acquire knowledge through exploration of the field.

- The goal of mobile robotics research is to develop systems capable of taking decisions autonomously. Autonomous Navigation is achieved using sensory information taken from sensors installed on robots. The key problem that the robot faces is the mapping of the environment. The ability of a system using its own sensors, to move in its environment autonomously is referred to as Navigation. The 3D structure not belonging to the plane on which robot moves can harm the robot if it crashes in one of these 3D structures, and therefore are called obstacles. The term reactive is used as there is a lack of information regarding the environment on which robot is moving i.e. the exact position of obstacles is unknown for robot.

The mapping that will be performed is actually the correlations existing between sensor data which is actually the representation of environment and the response of the robot.

The problem statement of field of view states that the robot should be able to construct its own field of view based on the sensory information received.

The obstacles are varying in size and distribution. The robot is trained for the whole map. The data is fed to the network based on the

Reinforcement learning (RL), a sub-area of machine learning, is derived from operant conditioning theory of psychologist Pavlov [1]. . It is a method of exploring feasible tactics and exploiting already known reward experiences to obtain a near-optimal policy. RL is one of the most effective methods of solving optimization problems [2]. RL is suitable for building intelligent systems such as autonomous helicopter [3], computer chess player [4], and spoken dialogue system [5].

- *Mr. Sameer Shastri is currentlylecturer in Government Engineering College Raipur, India,*
- *Ritesh Dandotia is currently pursuing masters degree program in information technology in ABV IIITM Gwalior, India,. E-mail:ritesh.dandotia@gmail.com*

2 RESEARCH METHOD

2.1 SOMQ

The Self Organizing Map gets its motivation by the feature of brain of the human being. The neurons are organized in multi-dimensional lattice. The neurons compete against themselves to be activated according to competitive learning scheme. The weight vector associated with winning neuron is only updated in the scheme "winner takes all". In "soft-max" rule, however , not only winning neuron but also other neighborhood neurons take part in the self organizing process. Kohonen, introduced a novel neighborhood concept, where the topology of input-data space can be learned through Self Organizing Map. In this scheme also, the neuron lattice can be one or multi-dimensional. A neighborhood concept among individual neurons in a lattice priori embedded. As neurons update their weights upon competition, a meaningful coordinate system for different input feature over the lattice is developed.

The neural lattice can be multi dimensional. The input vector x, excites each neuron. Each neuron is associated with a weight vector w_i, the dimension of the weight vector is same as that of input vector. If the input vector is N x 1, then the weight vector will also be of N x 1. A specific neuron wins on the basis of distance measure $f(||x - w_i||)$. The first step is to initialize the weight of the network.

There are three processes that are involved in the training of the Self Organizing Map:

a. Competition
b. Cooperation
c. Weight Update

Self Organizing Map topologically distributes the neurons over its entire domain.

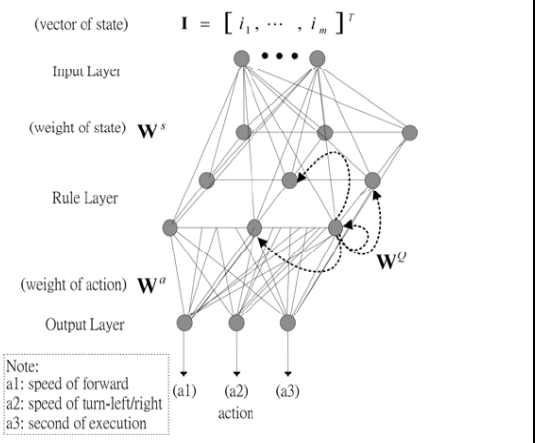

Fig. 1. SOMQ Network

The input layer takes the sensory input as the input to the layer. This represents the state of the robot in the environment. The next layer is rule layer that map the input state of the robot to the possible actions of the robot. This layer is a layer of nonlinear units that are connected to all the nodes in the input layer.

Q-learning is a popular approach to reinforcement learning, which learns an action-value function stored in a large Q-table. In Q-learning, an agent tries to obtain the greatest reward by taking an action in the current state according to its action value function, which is to be improved after a reward is given by the environment. the effectiveness and efficiency of the learning process depends much on the quality of the basic knowledge. Touzet proposed a method of learning with a continuous state space by combining reinforcement learning and neural network [7].

The output layer consists of units fully connected to the hidden layer. The output layer finally predicts the best action that should be taken by the robot to the given sensory input.

The input vector I has dimensions m and the output vector A representing action of the robot has dimension p.

There are three kinds of synaptic connections in the network:

- Feedforward connections from the input nodes to each of the neurons in the hidden layer. These connections to neuron j in hidden layer are represented by weight vector.

$$W_j^s = [\, w_{j1}^s, w_{j2}^s, \ldots\ldots, w_{jm}^s \,]^T$$

- Lateral connections from individual neuron to each of its neighbor neuron. These connections are represented by the feedforward weight vector for a neuron j.

$$W_j^Q = [w_{s1,j}^Q, w_{s2,j}^Q, \ldots\ldots\ldots, w_{sr,j}^Q]^T$$

here s1,s2.. represents the neighbor of the neuron j.

- Feedforward connections from each of the neuron of rule layer to the action layer. These connections for a neuron j are represented by weight vector.

$$W_j^a = [\, w_{1j}^a, w_{2j}^a, \ldots\ldots, w_{pj}^a \,]^T$$

The complete algorithm to train network is given as:

- Initialize all the parameters of the network which include effective width σ of the topological neighbourhood, number of iterations, learning rate η, time constant **λ**.
- All the weights associated with all the connections be it lateral or feedforward connections, are randomly initialized.
- An input vector I is given to the first layer. Using this I based on the competition the winning neuro is selected using equation (1).

$$S_j = arg \quad min \quad || \quad I \quad - \quad Wj^s \quad || \tag{1}$$

where j = 1,2,3…..,n

- The winning rule unit A_k is selected using (2)

$$A_k = arg \quad max \quad W_{x,j}^Q \tag{2}$$

- Identify the proposed action as the weight vector W_{Ak}^a.
- Get the reward r from the environment corresponding to the given input vector I. This basically represents the action output that the robot takes for the given input I.
- If $r + \gamma \times \max W_{x,Sj}^Q > W_{Ak,Sj}^Q$ where γ is discount rate, then update all weight vector W_k^a by using (3), if $d^2_{k,Ak} < \sigma^2$.

$$W_k^a = W_k^a \quad + \quad \eta \quad h_{k,Ak}(W_{Ak}^a \quad - \quad W_k^a) \tag{3}$$

Where h is the value of neighbourhood function.

$h_{k,Ak} = \exp(- d^2 / 2\sigma^2)$.

Here d denotes the lateral distance between winning neuron and excited neuron.

- The weight vector corresponding to the state is updated

$$W_{k,j}^Q = W_{k,j}^Q + \alpha \times h_{j,S_j} \times h_{k,A_k} \times (r + \gamma \times \max_{\lambda \in N(S_j)} W_{\lambda,S_j}^Q - W_{k,j}^Q), j = 1,2,\ldots,n; k = 1,2,\ldots,n. \tag{5}$$

using equation (4), if $d^2_{j,Sj} < \sigma^2$.

$$W_j^s = W_j^s \quad + \quad \eta \quad h_{j,Sj}(\quad I \quad - \quad W^s{}_j) \tag{4}$$

- The weight of lateral connections are updated using
- Learning rate and width of neighbourhood is decreased.

t=t +1.

Learning is terminated when the termination condition is met. Otherwise the algorithm continues from step 2.

After learning is completed using the above algorithm,

$$action = (1 - \frac{\|\mathbf{I} - \mathbf{W}_y^s\|}{\|\mathbf{I} - \mathbf{W}_y^s\| + \|\mathbf{I} - \mathbf{W}_z^s\|})\mathbf{W}_y^a + (1 - \frac{\|\mathbf{I} - \mathbf{W}_z^s\|}{\|\mathbf{I} - \mathbf{W}_y^s\| + \|\mathbf{I} - \mathbf{W}_z^s\|})\mathbf{W}_z^a$$

when the input vector corresponding to the sonar values of the robot is fed into the trained network, the two neurons are selected with most similar value to input vector represented as y and z. The output is calculated using equation:

(6)

2.2 Root Mean Square Error

Once the network is trained the projected value and the expected value can be found for some known input values. Using this one can find the error involved in the prediction of the action of the robot corresponding to the sonar input of the robot i.e. state of the robot in the environment. The third action i.e. time is taken to be constant.
RMSE is calculated using the equation:

$$Error = ((a1_{expected} - a2_{projected})^2 + (a2_{expected} - a2_{projected})^2))^{1/2} \quad (7)$$

3 Experiment

The robot is designed in Player/Stage which has the following parameters. It has five sonars mounted on it which are at an angle of 30 degrees. In all, it covers 180 degrees of the field of view of the robot. The environment that has been selected is unknown and unstructured obstacles.

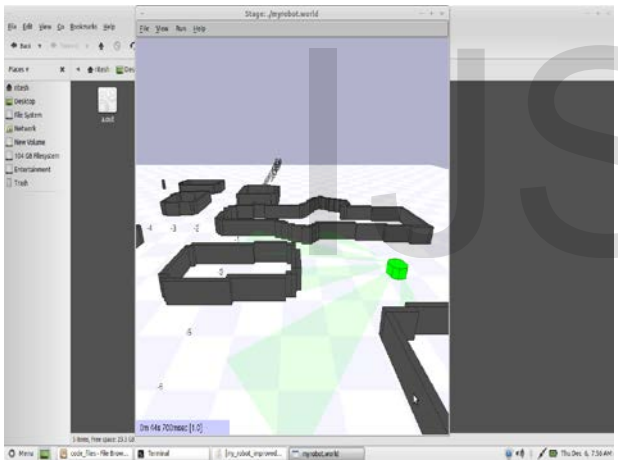

Fig.3.1: It gives the perspective view of the robot and its environment

Using player stage, the data is collected for the training of SOMQ.

Method 1: The data is collected representing sonar values and the corresponding action values of the robot. The sonar mounted upon are 5 in numbers.

Method 2: The data is collected having two more factors representing the x position and y position of the robot and the corresponding action of the robot.

The data is collected for target seeking robot with same conditions for both the methods.

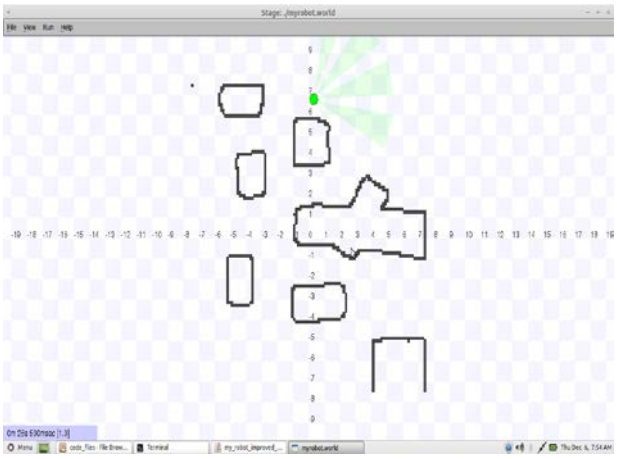

Fig.3.2: It represents the movement of the robot in the environment

4 Experimental Results

If Once the network of the neuron is trained using the algorithm mentioned in the Section 5, the error is obtained over 100 iterations using the equation (7).

The input or the state of the robot in the environment is known and the corresponding action of the robot is also known. When the input is fed into the trained network it predicts the corresponding action of the robot. The two values expected and the projected is known. Using this error can be calculated.
The error is calculated and is depicted in the figure below.

The localization value of the robot is included in the input to the network. The values are calculated using getX and getY function of the robot.

This input value to the network is known and corresponding action value or the reward is known. Using this network is trained again and the corresponding root mean square error is calculated. The error is calculated using the expected and the actual values.

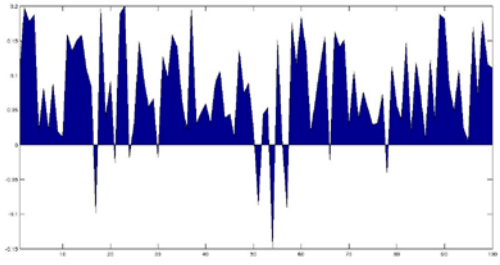

Fig.4: Error difference between the RMSE calculated by the two methods proposed.

.

The error difference of the two networks is calculated and plotted against the number of iterations. As can be seen, the error is significantly reduced by including the localization factor in the input.
Root mean square error for both the cases is calculated and compared. As can be seen the error decreased for the localization added input..

5 CONCLUSION

The error was calculated for the two methods discussed and the corresponding error difference is calculated. As can be seen due to the localization factor, the error has decreased. After 500 iterations, for both the cases the robot was able to predict the action of the robot. The error difference is calculated over the hundred iterations.

The algorithm may be improved using genetic programming and dividing SOM networks for each category. The decision of the selection of the network is taken by genetic algorithm and the SOM network is trained in the similar way.
The algorithm may be checked on the real robot to check its effect and improve it accordingly

REFERENCES

[1] N.D. Daw, "Reinforcement learning models of the dopamine system andtheir behavioral implications," Ph.D. dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA, 2003..

[2] R. Sutton and A. Barto, *Reinforcement learning: An introduction*, The MIT Press, 1998.

[3] P. Abbeel, A. Coates, M. Quigley, and A.Y. Ng, "An application of reinforcement learning to aerobatic helicopter flight," *IN ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 19, vol. 19, 2007*.

[4] Kajal Sharma, Sung Gaun Kim and Manu Pratap singh, "An improved feature matching technique for stereo vision applications with the use of self organizing map" International Journal of Precision Engineering and Manufacturing, August 2012, Vol. 13, Issue 8, pp 1359-1368.

[5] K. Kiviluoto: "*Topology preservation in Self-Organizing Maps"* Proc. IEEE International Conference on Artificial Neural Networks, 1996, vol.1, pages 249–254.

[6] Figueiredo, M.; Botelho, S.; Drews, P.; Haffele, C., "Self-Organizing Mapping of Robotic Environments Based on Neural Networks," *Neural Networks (SBRN), 2012 Brazilian Symposium on* , vol., no., pp.136,141, 20-25 Oct. 2012

[7] C. Touzet, "Neural reinforcement learning for behaviour synthesis," *ROBOTICS AND AUTONOMOUS SYSTEMS*, vol. 22, 1997, pp. 251-281.

[8] Wing-Kwong Wong, Hsin-Yu Chen, Chung-You Hsu, Tsung-Kai Chao, "Reinforcement Learning of Robotic Motion with Genetic Programming and Self-Organizing Map", Conference on Technologies and Applications of Artificial Intelligence 2011.

[9] T. Kohonen, "Self Organized Formation of Topologically Correct Feature Maps," Biological Cybernetics, Vol. 43, No. 1, 1982, pp.59-69.

[10] Latombe, J.C. 1991,Robot Motion Planning, Boston: Kluwer.

[11] P. Abbeel, A. Coates, M. Quigley, A. Y. Ng, "An application of reinforcement learning to aerobatic helicopter flight" IN ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 19, Vol. 19, 2011, pages 35-49.

[12] W.K. Wong, H.H. Chen and S.C. Hsu, "Reinforcement learning for training a program for computer chess," International Journal of Intelligent Information and database System, Vol. 6, 2012, pp. 246 – 258.