World Modeling in Disaster Environments with Constructive Self-Organizing Maps for Autonomous Search and Rescue Robots

Çetin Meriçli, I. Osman Tufanoğulları, and H. Levent Akın

Boğaziçi University,
Department of Computer Engineering,
34342 Bebek, Istanbul, Turkey
{cetin.mericli, akin}@boun.edu.tr
osmantuf@garanti.com.tr

Abstract. This paper proposes a novel approach for a Constructive Self-Organizing Map (SOM) based world modeling for search and rescue operations in disaster environments. In our approach, nodes of the self organizing network consist of victim and waypoint classes where victim denotes a human being waiting to be rescued and waypoint denotes a free space that can be reached from the entrance of debris. The proposed approach performed better than traditional self-organizing maps in terms of both the accuracy of the output and the learning speed. In this paper the detailed explanation of the approach and some experimental results are given.

Keywords: Search & Rescue Robotics, Self-organizing Maps, Mobile Robotics, World Modeling.

1 Introduction

Search and rescue (SR) robotics is one of the promising areas of mobile robotics. The main aim of the SR robots is exploring the debris after a disaster (especially, after an earthquake) and locating the living victims in the collapsed buildings, if any. Since a map of the debris is usually not available, the robot has to make the map of the environment simultaneously with the exploration and victim detection process while marking the locations of victims to be saved on the generated map.

The robot should have the ability to determine its position and orientation in the debris by using sensory inputs. One of the most important methods for localization is using natural or artificial landmarks in the environment[1] [2] [3]. Since there are no known landmarks in an unknown environment, this method can not be used. Odometry sensors providing the relative displacement of the robot with respect to its initial position can also be used for localization. Because of friction, slippage and encoder errors in the locomotion parts, odometry data are fairly noisy. This noise increases the error in estimation cumulatively. In order to overcome this problem, position information must be corrected periodically. Using GPS is another alternative for position estimation but accuracy of commercially available GPS receivers are not so high. In this work, we

D. Nardi et al. (Eds.): RoboCup 2004, LNAI 3276, pp. 467–473, 2005.

assumed that the robot is equipped with a GPS receiver with high accuracy. Since the accuracy of the GPS is limited (outputs of GPS sensors are also noisy), the map making algorithm should be able to use noisy inputs.

Some approaches for world modeling and position estimation using self-organizing networks for mobile robots have been proposed in literature. Marques $et\ al$ have proposed a system which uses sensory layer inputs for training a self-organizing network and after the training phase, finding the most similar neuron for each perception from sensors, and assuming the position of the robot is the position of the winner neuron in the network [4]. Nehmzow $et\ al$ have used self organizing feature maps for position estimation in which they are feeding the network with history of motor actions instead of sensory data[5].

Topological map representation is appropriate for mapping the unknown environments because of its property of learning both distribution and topology of the data. In this work, we have used a constructive variant of Kohonen's Self Organizing Map for marking accessible free locations in the universe (which we call a Waypoint) and the locations of detected humans (which we call a Victim). We have used a constructive network architecture because it is assumed that we do not have any prior information about the disaster environment[6] so we should make a map of the environment in order to be able to mark detected humans and free spaces for reaching them on the map.

The rest of the paper is organized as follows: Brief background information about self-organizing networks is given in Section 2. Detailed description of proposed approach is given in Section 3. Section 4 covers experimental work and the last section is dedicated to conclusions.

2 Background

2.1 Self-Organizing Maps

Self-organizing maps are a special kind of neural networks that can learn to detect regularities and correlations in their input and adapt their future responses according to that input [7]. Competitive learning is used to learn to recognize groups of similar input vectors in such a way that neurons physically near each other in a layer respond to similar input vectors. For a given input, the most similar neuron (called The Winner) is selected based on the distance between input and the neurons (neurons compete with each other in order to be the winner). Then, the winner neuron and the neurons in a certain neighborhood of the winner neuron are updated with a rule called Kohonen Learning Rule:

$$w_{ij}(t) = w_{ij}(t-1) + h(w_i, w_g) \cdot \alpha \cdot (p_j(t) - w_{ij}(t-1)) \tag{1}$$

where, w_{ij} is the j^{th} weight of i^{th} neuron, p_j is the j^{th} component of input, α is the learning rate, $h(w_i, w_g)$ is the vicinity function that depends on the distance between the updating neuron and the winner neuron, w_i is the updating neuron and w_g is the winner neuron. The vicinity function decreases as the distance of the updating neuron to the winner neuron increases, and becomes zero if the neuron is not in the vicinity of the winner neuron. Each time a neuron is updated, all the neurons in the vicinity of the neuron are also updated. Self-organizing maps learn both the distribution and the topology of the input vectors which they are trained on[8].

3 The Proposed Approach

In our approach, we have used a constructive self-organizing map for representing the free spaces and victims in the environment, accessibility of victims by using connectivity among free spaces and connectivity between victims and free spaces. It is assumed that, the sensorial layer of the robot supplies two kinds of signals: obstacle information and people detection information. Both of the signals are real numbers in the range of [0, 1] and represent the confidence about existence of either victim or free space ahead of the robot. Receiving a victim signal strengthens the belief about the existence of a victim node in the observed location and receiving an obstacle signal weakens the belief about the existence of a free space (a waypoint) in the observed location. Since usually the debris has more obstacles than free spaces (we can consider it as a maze-like environment), it is reasonable to keep track of free spaces and the connectivity of these spaces instead of marking obstacles on the map. For this purpose, we divide the nodes into two types: $Waypoints$ and $Victims$. Waypoints denote the free spaces that the robot can pass through and Victims denote detected humans. The nodes contains information about node type, the 3D position information, the number of hits of the node and the average confidence value of the node. Since only one network is used for map generation and the map contains two classes of nodes, class specific update rules and environment specific linking methods among and between classes have been developed.

In this work, we assumed that there are obstacle and victim detection modules that supply the probability of encountering an obstacle or a victim. Each time an observation is received, only the nodes belonging to the same class with the observation are updated (i.e. a waypoint node is neither updated nor selected as a winner when an observation of waypoint signal is received even if it is in the vicinity of the winner or it is the winner itself in terms of distance to the observation). If none of the nodes belonging to the same class are closer to the observation point than a certain threshold, a new node of the observation type is introduced to network. When a new node is constructed, neighborhood information of the nodes should be updated. In this phase, not only the distances between the nodes but also the physical accessibility from one node to another is considered. If a node is not accessible from another node, it is not added to the neighbor list of that node even if it is in the neighborhood range. Whenever a victim observation is made with a confidence greater than a certain threshold, the victim nodes are updated according to Kohonen Learning Rule but unlike the conventional rule, the amount of update is multiplied by the confidence of the signal. This prevents the nodes from diverging from the correct position when a signal with a low confidence is received for a long time. Each node, waypoint or a victim, can be of fixed type or variable type. A fixed node is not updated through learning even if it is in the vicinity of the winner neuron. A node is set as fixed if it has a number of wins greater than a certain number and its average confidence is greater than a certain threshold. The idea is that if there are more than a certain number of observations denoting that there is a victim in a location with a confidence over a certain threshold, that node should be fixed, and should not be updated anymore. The network starts with an initial waypoint (the first node of the system) representing the entrance point of the robot into the debris and is set to be fixed. A pseudo-code of the algorithm is given in Figure 1.

```
Initialize the network by defining a starting node
at (0,0,0) and fixed.

If an observation is received
    Find the winner node by comparing the distances
    between nodes and incoming observation

    If distance between winner and observation is
    less than a threshold,
        Update the nodes in te vicinity of the winner
        node belonging to the same type with observation
        type and set the node status to variable

        Update hits and average confidence fields of
        the winner

    Else
        Create a new node with type of observation and
        set the status to variable

        Update the neighborhood information for the new node
        and the nodes in its vicinity.

    End

    If the winner has a count of hits over a threshold and
    its average confidence is over threshold,
        set the node status of winner to fixed

End
```

Fig. 1. Pseudo-code implementation of the proposed approach

After the exploration is finished and training of the network is completed, the resultant network is a graph consisting of free spaces and victims and the links between nodes denoting the accessibility of nodes.

4 Experimental Work

4.1 Simulation Environment

For the experimental work, The Webots™ Mobile Robot Simulator version 3.2.22 is used as the testing environment. Webots™ uses a VRML97 compliant scene representation scheme and allows the user to develop C/C++ and Java robot controllers by providing API for accessing the simulator functions[9].

We have used a model of a Khepera robot equipped with a color camera as the prototype of SR robot and the controller for the robot is written in Java. For the experimentation, the robot is guided remotely and we mimic the signals from the people

detection module by performing image processing. The light grey cylinders represent the victims in the environment. The proportion of the number of light grey pixels in the image obtained from the camera of the robot to the number of all pixels in the image is a number between [0, 1] indicating the confidence of the existence of a victim in the visual field of the robot. The people detection module of the robot is assumed to return the estimated coordinates of the detected people (if any) and the confidence about this estimation. In our experiments, the location of the robot is considered as the signal location and the results from image processing are considered as the confidence level of the signal. As the robot wanders around, the observations are passed to the network depending on the strength of the signals received. For the waypoint signal production, the information from infrared (IR) range sensors in front of the robot is used. The IR sensors of Khepera return a number in the range $[0, 1]$ where 0 means that there are no physical obstacles in the range of the sensor and 1 denotes a very close obstacle to the sensor. In our experiments, we have used the average of values obtained from four front IR sensors of the robot as the obstacle information which is again a number in the interval $[0, 1]$. Since an obstacle signal with a value of 1 means that the robot is confronted with a very close obstacle, we expressed the amount of free space ahead of the robot as $1 - \gamma$, where γ is the obstacle signal obtained by taking average of four frontal infrared sensors. It is assumed that the robot is equipped with a GPS receiver for its self localization. To mimic the noise in the GPS system and IR sensors, uniformly distributed random numbers are added to the exact coordinates of the robot and IR data.

4.2 Experiments

For comparison, two different networks are used in the experimentation phase: A conventional Kohonen's Self Organizing Map and our modified version of self organizing network. For the conventional SOM part, Matlab Neural Network Library is used whereas our modified version of the network is implemented in Java. The conventional SOM used is a network of 5 x 1 x 5 nodes since the displacements in the Y axis can not be obtained due to the limitations of the simulator. Our network starts with one initial node. The SOM network is trained using victim observations in 100 epochs. 1876 victim observations were obtained during the simulation by making the robot to follow the path given in Figure 2 where the circles represent the victims in debris. The number of the observations depends on the running time of the simulation.

Both the outputs of our implementation in one epoch and the output of SOM in 100 epochs can be seen in Figure 3. Here, light grey nodes denote waypoints, dark grey nodes denote victims, circles denote variable nodes and squares denote fixed nodes. It can be seen easily that the output of our approach has some major advantages over the classical self-organizing map approach. Our approach can represent both the victim locations, the free spaces and the connectivity of these free spaces for reaching victims and because of its custom update rules and spatially constrained vicinity definition, the victim locations are found with a higher accuracy than classical SOM. The training phase of the traditional SOM took approximately 10 minutes on a Pentium 4 based computer whereas, our approach does not need such a training phase since it generates the map online while the robot is wandering in the debris.

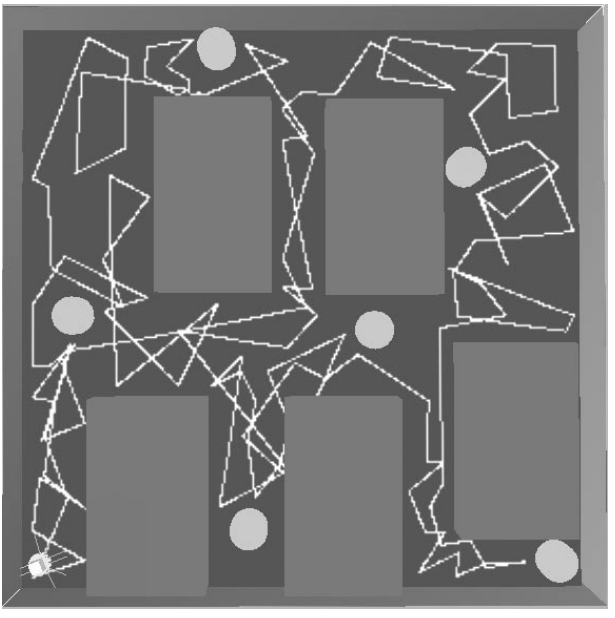
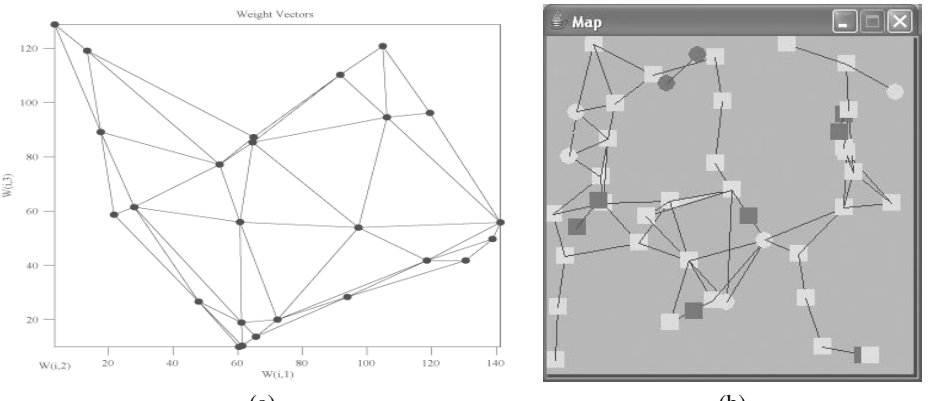


Fig. 2. Path of the robot in debris



(a) (b)

Fig. 3. Comparison of outputs of (a) traditional SOM and (b) our approach

5 Conclusions

SR robotics is gaining importance as the disasters causes large scale loss of human life and most of this loss is due to inefficiency in SR methods. By using SR robots in collapsed buildings, the chance of detecting the existence and locations of victims in debris can be increased.

In this work, an incremental self-organizing network based map generation methods for a search and rescue robot is presented. The main aim of this work was to come up with a world modeling algorithm that can represent the locations of human beings in a collapsed building and a list of free spaces that allows reaching those victims. The proposed approach is implemented on a simulator and compared with a traditional self-organizing map. Since debris is a maze-style environment, it is not easy to define neighborhood between neurons. In the definition of neighborhood functions, spatial constraints such as physical accessibility between nodes are used in addition to Euclidean distance between nodes. Using spatial accessibility prevents updating unrelated nodes even if they are in the vicinity of the winner neuron in terms of euclidean distance. There are two types of nodes in the network: Waypoints and Victims. Waypoints denote the free space that the robot is passed on in the exploration and Victim denotes a detected living human being waiting for to be rescued. Since we have two classes of signals to be learned and there are some spatial constraints in vicinity functions, a traditional self-organizing map is not sufficient for representing the world model. With its spatial constrained neighborhood definition and partial update method for updating different types of nodes, our proposed approach has brought a novel idea to the map making area in unknown environments and performed better than traditional SOM in both accuracy and learning speed.

Acknowledgements

This project is supported by Boğaziçi University Research Fund project 03A101D.

Special thanks to Prof. Ethem Alpaydın and Hatice Köse for valuable discussions and reviews.

References

1. A. Arsenio, "Active Laser Range Sensing for Natural Landmark Based Localization of Mobile Robots", M.Sc. Thesis, IST, Tech U Lisbon, 1997.
2. M.I. Ribeiro and J. G. M. Goncalves, "Natural Landmark Based Localization of Mobile Robots Using Laser Range Data", *Proceedings of the 1st Euromicro Workshop on Advanced Mobile Robots,* Kaiserslautern, Germany, 1996.
3. H. L. Akın *et al*, "Cerberus 2003 Team Report", Bogazici University, Istanbul, 2003.
4. R. Marques, E. Zalama, J. G. García-Bermejo, and J.R. Peran, "World Modeling and Position Estimation for a Mobile Robot Using Self-Organizing Networks" *4th IFAC International Symposium on Intelligent Components and Instruments for Control Applications,* SICICA 2000, Buenos Aires, Argentina, 2000.
5. U. Nehmzow, T. Smithers and J. Hallam, "Location Recognition in a Mobile Robot Using Self-Organising Feature Maps", *Information Processing in Autonomous Mobile Robots,* Springer-Verlag, 1991.
6. S. Zrehen and P. Gaussier, "Why Topological Maps Are Useful for Learning in an Autonomous Agent", *In Proceedings PerAc, IEEE Press,* Lausanne, September 1994.
7. T. Kohonen, "Self-Organization and Associative Memory.", *Series in Information Science, Vol. 8. Springer-Verlag,* Berlin, Heidelberg, New York, 1984.
8. R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification*, Second Edition, Wiley & Sons, 2001
9. "Webots Mobile Robot Simulator", *http://www.cyberbotics.com*, 2003.