Learning Image-Based Landmarks for Wayfinding using Neural Networks

Jeremy D. Drysdale Damian M. Lyons
Robotics and Computer Vision Laboratory
Department of Computer Science
340 JMH Fordham University
441 East Fordham Rd.
Bronx, NY 10458
drysdale@fordham.edu

Keyword list 1: Neurally-Based Networks
Keyword list 2: Adaptive Backpropagation, Artificial Neural Network, Backpropagation Learning Algorithm, Feature Identification, Feature Recognition, Fuzzy Neural Networks, Image Data Acquisition, Image Interpretation, Image Processing, Mean-Square Error, Mobile Robots, Neural Network Applications, Neural Networks, Real Time Neural Network Model, Robotics

Abstract:

In this paper we address the problem of automatically selecting and predicting landmarks for use in wayfinding on a mobile robot. Our constraints for wayfinding landmarks are that we be able to uniquely recognize the landmark from a visual image and be able to tell whether the landmark is growing nearer or farther away. We employ back-propagation to teach a multilayer neural network to predict the *image location* and appearance of future landmarks based on the appearance and *image location* of currently visible landmarks.

We employ a hybrid reactive/deliberative architecture [2]. The reactive component collects candidate landmark feature measurements while the robot explores its (previously unknown) environment. For the deliberative component, we use a neural network running on a Beowulf cluster to process the large amount of data being collected from camera. We describe our architecture and algorithm in detail and present data gathered using our Pioneer DX2 robot to learn wayfinding landmarks.

1.0 Introduction:

A robot traversing unknown terrain needs to have a method to find its way back. For example a Search & Rescue robot [9] winding through a complicated rubble field looking for victims or other targets needs to have a way to return the target to base for medical help or backup. Clearly an a-priori map of a rubble field would not be available. One approach could be to construct a detailed geometric map of the area as the robot moves. This approach has many advantages but has several practical problems. The geometric features of a rubble field are less easy to define than indoor or outdoor natural terrain; and the landscape may be ephemeral ñ changing rapidly due to weather, or due to local rubble motion, or even due to motion of the robot. An alternative approach is to learn qualitative features of the landscape: landmarks that will help in navigation at a course level.

When the robot is in an area it doesnít know, it becomes necessary for it to identify and learn a useful set of landmarks in its surroundings, so that it can return through the area, knowing exactly where to go, and whatís around. Throughout this learning process the robot must identify potential landmarks based on sensory information, learn where those landmarks are, and then be able to predict, with reasonable accuracy, where to expect to sense the next landmark.

Insects appear to employ a ì visual snapshotî concept of landmark navigation [5] and we are inspired by this in our purely image based representation and prediction approach. In this paper we will present an approach to landmark identification and learning. We employ a multilayer neural network to learn to predict expected image landmark locations based on the image locations of current landmarks. Because we

expect landmark learning to be an ongoing, data-intensive background computation, we also propose a novel landmark learning architecture based on the hybrid reactive/deliberative architecture [2] consisting of reactive landmark identification & prediction coupled to a concurrent and separate landmark learning system running on a 15 node Beowulf Cluster.

2.0 Literature Review:

Bayesian landmark learning [6] for robot localization involves the use of recursive Bayesian estimation to accurately locate features of the environment around the robot so that the robotís position and orientation with respect to the environment can be calculated. In [6] the network was trained to minimize sensory extraction errors so it was better able to extract landmarks in the future. Previous work done to solve the problem of learning landmarks for wayfinding include using edge density to determine what is a candidate for a landmark [3]. High edge density tended to mean that there was furniture or a wall, something that would make a good landmark present. The candidates were then grouped by which candidates corresponded to similar visual regions of the environment.

Figure 1: Pioneer DX2 Robot - front overhead

In [4], a pattern search engine was used to learn the patterns of the landmarks. Patterns were 2-D images containing odd but distinct shapes, symbols, text or a combination of a few of these. This worked well with landmarks that contained text, because the text created the patterns that were picked up by the system.

For our application, we expect that landmarks may modify their appearance and location while the robot is active. We would like therefore to have a more qualitative concept of landmark, while retaining the ability to use the landmark to guide wayfinding. One interesting piece of work was the studying of insects and how they were able to learn landmarks [5]. One such method was the Turn-Back-and-Look method used by bees. When observing the behavior of bees it was discovered that whenever a bee finds itself in an unfamiliar place, it turns around and views the landscape behind it. In this manner it is able to analyze where it has been and teach itself how where it is relates to what it already knows. This is similar to how we designed our system, it stores what landmarks it has seen previously, and where they are. It can use this information to get an idea of where the new landmarks it is seeing are.

In [7], a neural network is used to learn the terrain of the land as viewed from a camera mounted on a tractor. This work employs a grid overlayed on the camera view. Features can be observed and calculated for each cell in the grid, thus allowing for easy comparison between areas of an image. We will also adopt this approach for landmark feature calculations.

3.0 Identifying Visual Landmarks:

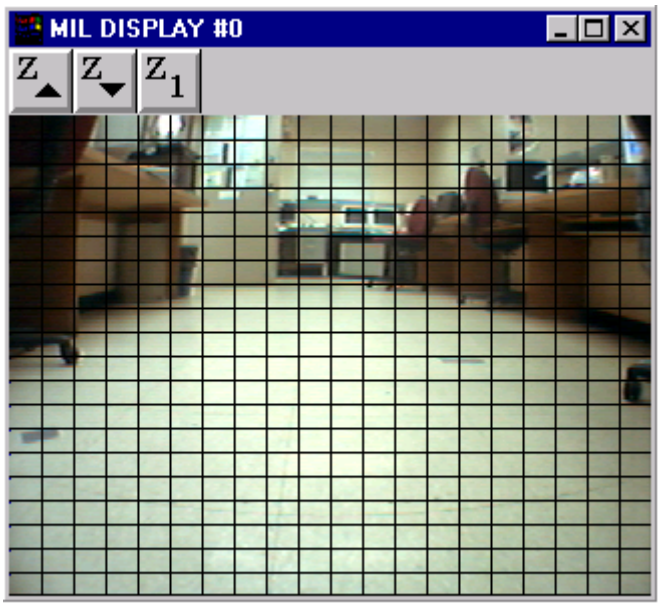


Figure 2: Grid

To identify visual landmarks we took as input a visual image as collected by the X10 camera mounted on the Pioneer. The image was divided into a grid Figure 2. This way, feature measurement could take place on it, and be evaluated by analyzing the features on particular grid squares and how they compared with the same features in the surrounding grid squares. The features calculated for each grid square were normalized average color and intensity where *i* and *j* refer to individual pixel coordinates within the grid square:

$$r_n = \frac{256 * r}{r+g+b}$$

$$g_n = \frac{256 * g}{r+g+b}$$

$$Y = \frac{r+g+b+1}{3}$$

$$\left(\sum_{i=0}^{xPixSize}\sum_{j=0}^{yPixSize} r_n\right) \Big/ \left(xPixSize * yPixSize\right)$$

$$\left(\sum_{i=0}^{xPixSize}\sum_{j=0}^{yPixSize} g_n\right) \Big/ \left(xPixSize * yPixSize\right)$$

$$\left(\sum_{i=0}^{xPixSize}\sum_{j=0}^{yPixSize} Y\right) \Big/ \left(xPixSize * yPixSize\right)$$

and standard deviation:

$$\frac{\Sigma(r_n - r_{navg})^2}{(xPixSize * yPixSize) - 1} \qquad \frac{\Sigma(g_n - g_{navg})^2}{(xPixSize * yPixSize) - 1}$$

A good landmark is one in which the standard deviation of the average color between grid squares is fairly low. When this is the case, it signifies that the landmark is large enough to cover multiple grid squares, and the color is contiguous enough such that there isnít too much difference.

A good landmark can be represented mathematically. If both the average normalized red and average normalized green fall within one standard deviation of the averages of an adjacent grid square, the adjacent squares are considered to be good landmarks.

$$(r_{navg2} - r_{nstdDev2}) < r_{navg1} < (r_{navg2} + r_{nstdDev2})$$

$$(g_{navg2} - g_{nstdDev2}) < g_{navg1} < (g_{navg2} + g_{nstdDev2})$$

A bad landmark is one in which the standard deviation of the average color is high. When this is the case it can signify a few things. One of which is that the landmark does not span multiple grid squares. This is not a good landmark because you cannot distinguish it from other aspects of the image easily. Another thing it can signify is a massive change in coloration. This tends to not be a good landmark because typically massive color changes indicate smaller objects around one another, which are difficult to see from a distance.

As the robot moves forward, the landmarks tend to move away from the center of the image at an angle. The direction they move depends on what quadrant the landmark is currently located in. As would be expected, when the robot moves backward, the landmarks tend to move toward the center of the image at an angle.

$x_{new} = x - 1$ $y_{new} = y - 1$	$x_{new} = x + 1$ $y_{new} = y - 1$
$x_{new} = x - 1$ $y_{new} = y + 1$	$x_{new} = x + 1$ $y_{new} = y + 1$

4.0 Learning Landmarks:

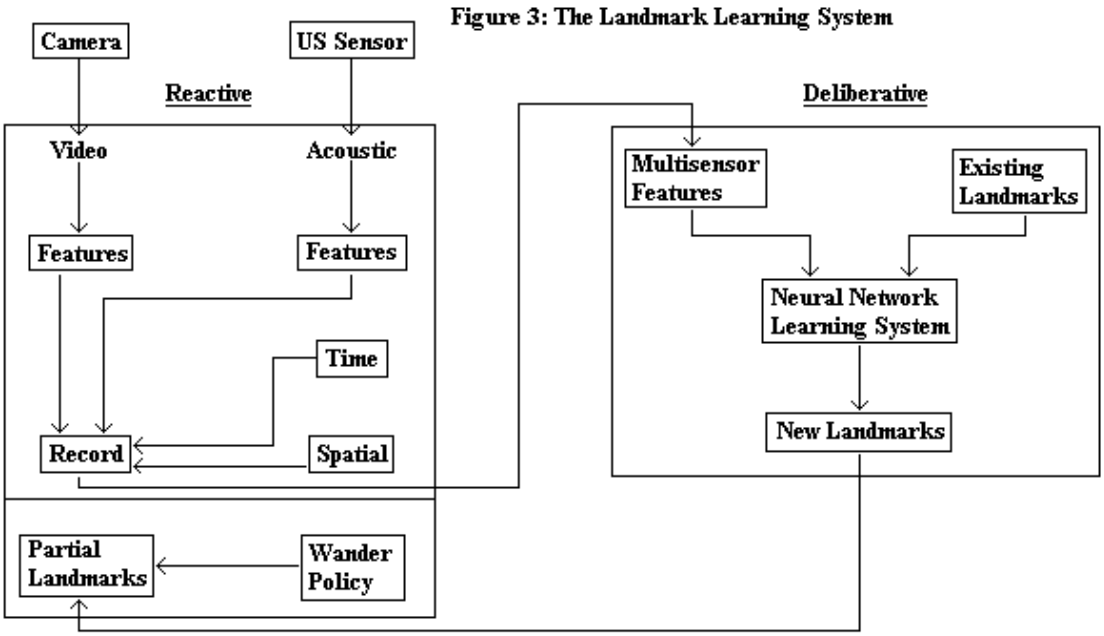


Figure 3: The Landmark Learning System

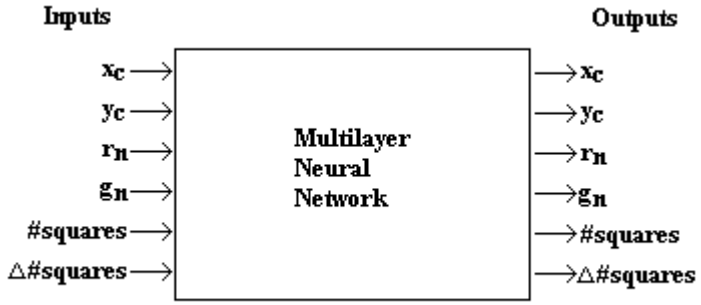


Figure 4: Neural Network Overview

The neural network in Figure 4 contains six (6) inputs, six (6) outputs, and six (6) hidden nodes. The function of this neural network is to take in the landmark information, and predict where the next landmark will appear. This neural network uses a standard backpropagation algorithm [8].

The input to the neural network is information about the closest landmark to the robot. This information is the center Cartesian x and y grid coordinate of the landmark,

the normalized average red value, the normalized average green value, the number of grid squares the landmark takes up, and the change in the number of grid squares since the last image was observed. The output of the neural network is the same information, except for the predicted landmark.

Because of the massive amount of data the robot will be taking in and processing, the landmark prediction will be extremely data intensive and always active ñ as long as the robot is active, data is being collected and analyzed. Because of this, we used a Beowulf cluster.

We plan to implement a system that will allow the robot to communicate with the Beowulf cluster through an open socket. The PC will send the Beowulf the data collected by the robot. The Beowulf cluster will receive this data and feed it through the neural network. Once this is complete, the Beowulf cluster will send the PC the information on the predicted landmark.

Currently the Beowulf cluster is being used to train the neural network. The training data is copied by hand to the Beowulf, and after the training is complete, the weights data is copied by hand again to the PC controlling the robot.

Experimental Method:

We began by creating a small course Figure 5 for the Pioneer robot to traverse so that it could gather learning data to train the network. The images collected by the camera were divided into a grid, so that the robot could use the grid squares as spatial coordinates. As the robot traversed the course it gathered information on landmark

Figure 5: Course

locations, their average colors, their sizes, and the difference in size between the current and previous images. This data was outputted to a file for use at a later time.

The Pioneer robot Figure 1 is outfitted with two sonar arrays, each containing eight (8) sonar for a total of sixteen (16). The camera mounted on the front is an X10 Wireless Camera, Model #VR36A. There is a wireless modem mounted on the back of the Pioneer for wireless connectivity. It is an ACT0106 Lo-speed Serial Ethernet device.

There were three training landmarks that were used. The first of which was an orange box containing black writing and a green image of a computer hardware card Figure 7. The box was of length 12î and width 10î and height 1.5î. The second was a dark blue bound notebook of length 11î, and width 8.75î and height 0.5î Figure 8. The third was a silver Aibo robotic dog standing on all fours of length 10î and width 6î and height 11î Figure 9. These landmarks were positioned at alternating points on either side of the robot starting from the left approximately two (2) feet apart.

As the robot traversed its data gathering course the camera image was displayed on the screen containing where it thought all landmarks where. This was accomplished by outlining the grid squares containing the landmarks in green Figures 10-12.

Figure 7: Landmark #1, Orange box

Figure 8: Landmark #2, Blue notebook

Figure 9: Landmark #3, Aibo robot dog

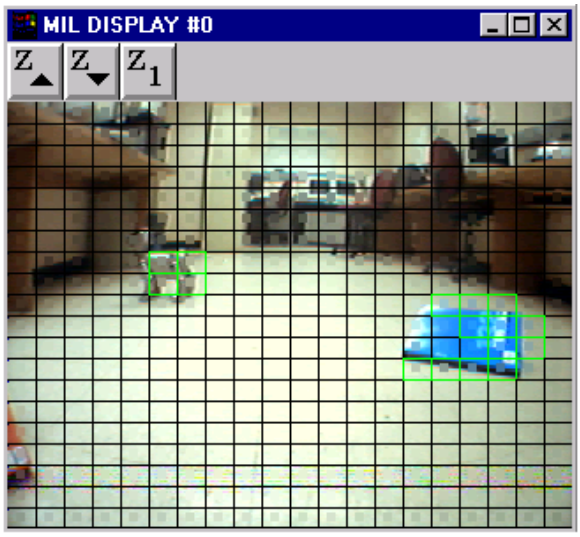


Figure 10: Landmark recognition

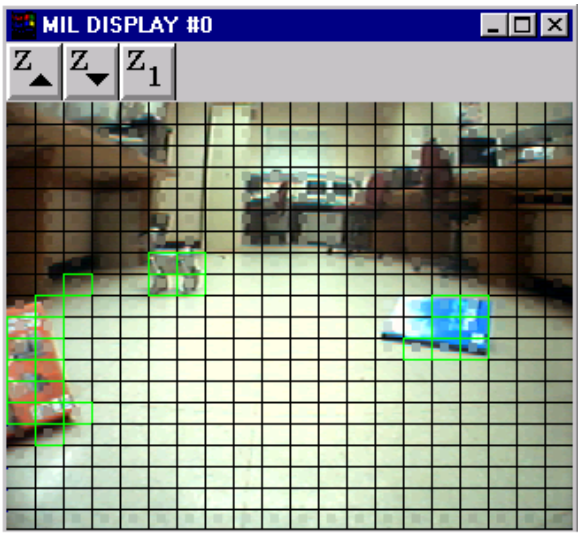


Figure 11: Landmark recognition

A three layer neural network was built next containing six (6) nodes in the hidden layer. This network was trained on the data gathered from the robots traversal of the course. The learning part of the network took place on a sixteen (16) node Beowulf cluster and the weights data was exported back to the PC containing the robotics code for testing. A backpropagation algorithm was implemented to learn the weights.

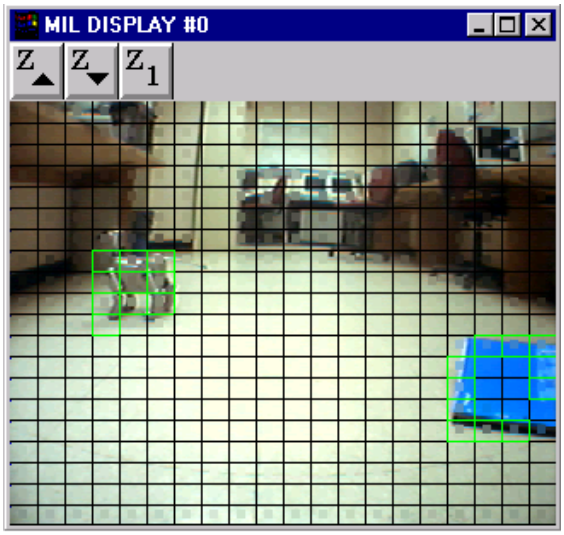


Figure 12: Landmark recognition

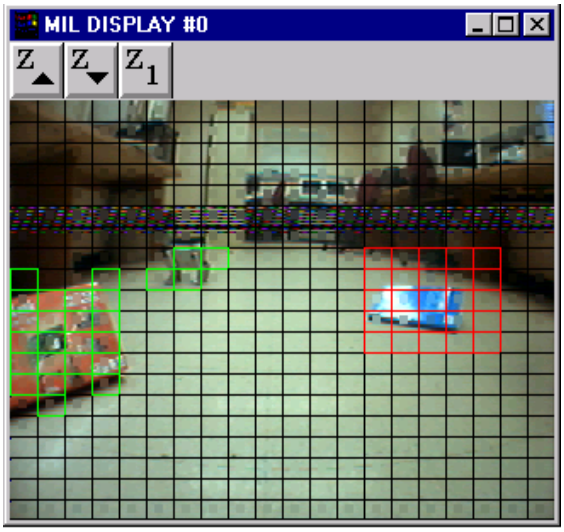


Figure 13: Correct Landmark Prediction

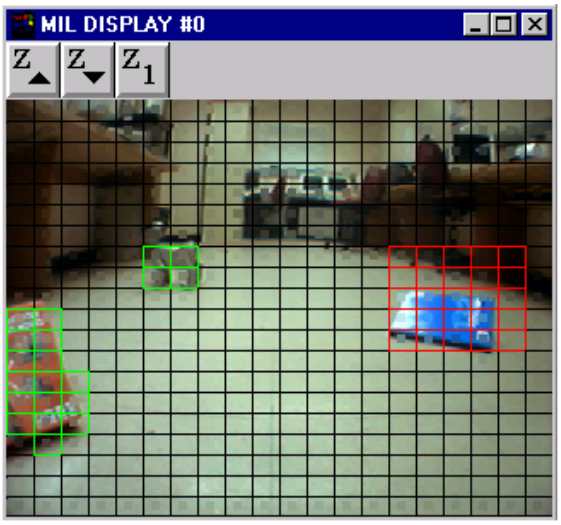


Figure 14: Correct Landmark Prediction

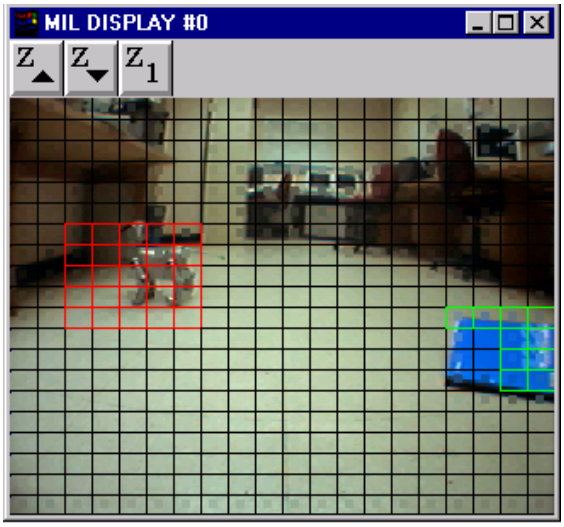


Figure 15: Correct Landmark Prediction

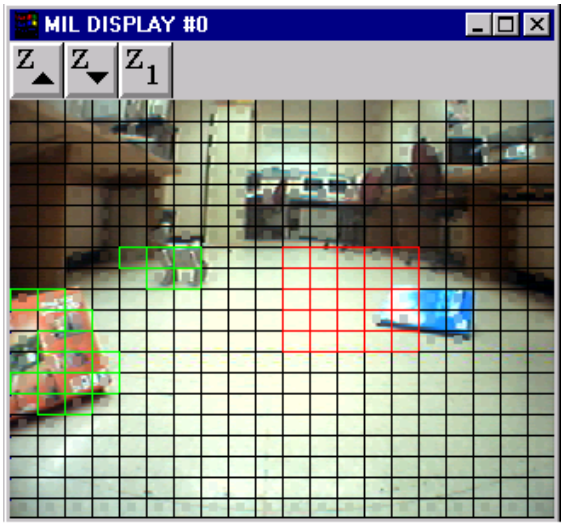


Figure 16: Incorrect Landmark Prediction

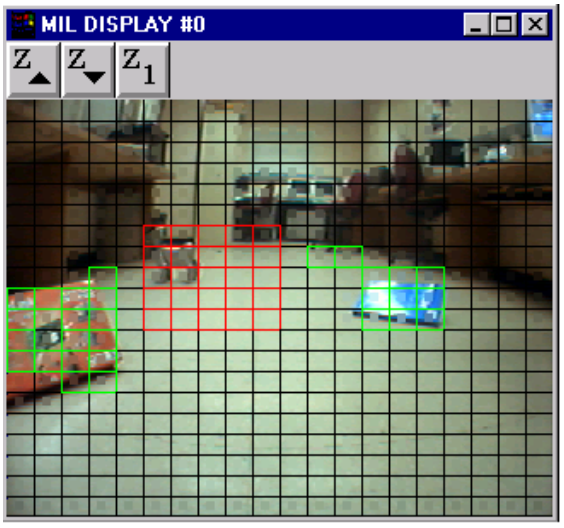


Figure 17: Incorrect Landmark Prediction

Now that the network was built and trained, the testing process could begin. There were four testing cases that took place on the network: forward landmark recognition, reverse landmark recognition, arbitrary start forward landmark recognition and arbitrary start reverse landmark recognition:

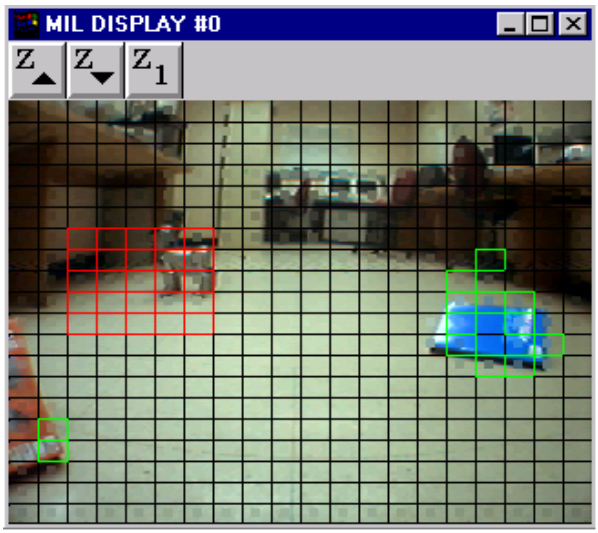


Figure 18: Incorrect Landmark Prediction

- Forward landmark recognition testing was done by traversing the robot through the course in the same direction as the training.
- Reverse landmark recognition testing was done by traversing the robot through the course in the reverse direction as the training.
- Arbitrary start forward landmark recognition was done by placing the robot in an arbitrary location on the course, and traversing it through in the same direction as the training.
- Arbitrary start reverse landmark recognition was done by placing the robot in an arbitrary location on the course, and traversing it through in the reverse direction as the training.

In all cases, the neural network took as input the landmark closest to the robot, and the output was the location and size of the next landmark. The predicted landmark was displayed on the screen by outlining the grid squares containing the predicted location of the landmark in red Figures 13-18.

5.0 Results:

The results of all the testing cases were good. The standard of error was calculated using the formula:

$$(predicted_x \text{ ñ } actual_x)^2 + (predicted_y \text{ ñ } actual_y)^2 + (predicted_{size} - actual_{size})^2$$

The forward landmark recognition testing results can be seen in Figure 19. The reverse landmark recognition testing results can be seen in Figure 20. The arbitrary start location forward landmark recognition testing results can be seen in Figure 21. The arbitrary start location reverse landmark recognition testing results can be seen in Figure 22.

The results of the forward landmark recognition testing were good. The neural network was able to pinpoint the y-axis location of the next landmark 38% of the time. Outside of this it was able to predict within three grid squares 62% of the time. It was able to pinpoint the x-axis location 45% of the time. Outside of this it was able to predict within three grid squares 16% of the time. The other 39% of the time the neural network was off by more. The neural network was able to pinpoint the size of the next landmark 21% of the time. Outside of this it was able to predict within three grid squares of the size 45% of the time. The other 34% of the time the neural network was off by more. These results are graphed in Figures 23ñ25.

The results of the reverse landmark recognition testing were also good. The neural network was able to pinpoint the y-axis location of the next landmark 33% of the time. Outside of this it was able to predict within two grid squares the other 67% of the time. It was able to pinpoint the x-axis location of the next landmark 40% of the time.

Outside of this, it was able to predict within three grid squares 20% of the time. The other 40% of the time, the neural network was off by more. The neural network was able to pinpoint the size of the next landmark 12% of the time. Outside of this it was able to predict within three grid squares of the size 52% of the time. The other 36% of the time, the neural network was off by more. These results are graphed in Figures 26-28.

The results of the arbitrary start location forward landmark recognition testing were the best of all the tests performed. The neural network was able to pinpoint the y-axis location 24% of the time. Outside of this it was able to predict within two grid squares the other 76% of the time. It was able to pinpoint the x-axis location 57% of the time. Outside of this it was able to predict within three grid squares 34% of the time. The other 9% of the time, the neural network was off by more. The neural network was able to pinpoint the size of the next landmark 19% of the time. Outside of this it was able to predict within three grid squares the size 62% of the time. The other 19% of the time, the neural network was off by more. These results are graphed in Figures 29-31.

The results of the arbitrary start location reverse landmark recognition testing were also very good. The neural network was able to pinpoint the y-axis location 29% of the time. Outside of this it was able to predict within two grid squares the other 71% of the time. It was able to pinpoint the x-axis location 33% of the time. Outside of this it was able to predict within three grid squares 24% of the time. The other 43% of the time the neural network was off by more. The neural network was able to pinpoint the size of the next landmark 5% of the time. Outside of this it was able to predict within the three grid squares the size 43% of the time. The other 52% of the time, it was off by more. These results are graphed in Figures 32-34.

Figure #19: Forward landmark recognition testing results

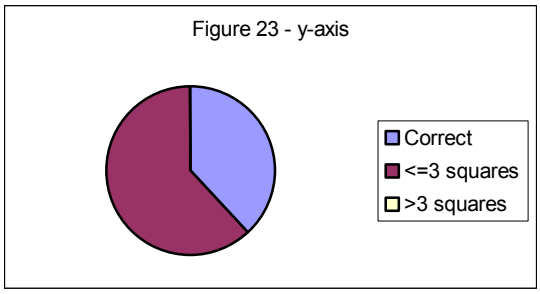
Predicted			Actual			$(p-a)^2$			$\Sigma(p-a)^2$
y_c	x_c	size	y_c	x_c	size	y_c	x_c	size	
8	9	11	8	13	11	0	16	0	16
8	4	10	9	14	6	1	100	16	117
8	6	10	9	14	18	1	64	64	129
8	4	10	7	4	5	1	0	25	26
8	4	10	7	4	5	1	0	25	26
8	4	10	8	4	6	0	0	16	16
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	7	1	0	9	10
9	15	12	8	4	9	1	121	9	131
8	5	10	7	3	10	1	4	0	5
8	8	11	8	3	11	0	25	0	25
8	4	10	9	17	10	1	169	0	170
8	4	10	8	3	14	0	1	16	17
8	8	11	8	13	11	0	25	0	215
8	4	10	9	15	12	1	121	4	126
9	14	12	9	14	19	0	0	49	49
8	4	10	7	4	5	1	0	25	26
8	4	10	7	4	4	1	0	36	37
8	4	10	8	4	6	0	0	16	16
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	7	1	0	9	10
9	15	12	7	4	10	4	121	4	129
8	5	10	8	4	12	0	1	4	5
8	6	10	8	3	12	0	9	4	13
8	4	10	9	17	10	1	169	0	170
8	4	10	8	3	14	0	1	16	17
8	9	11	8	13	11	0	16	0	16
8	4	10	9	15	12	1	121	4	126
8	8	11	9	15	16	1	49	25	75
8	4	10	7	4	6	1	0	16	17
8	4	10	8	4	6	0	0	16	16
8	4	10	11	15	7	9	121	9	139
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	8	1	0	4	5
8	9	11	8	4	10	0	25	1	26
8	5	10	8	3	12	0	4	4	8
8	5	10	8	3	13	0	4	9	13
8	4	10	9	17	10	1	169	0	170
8	3	13	8	3	13	0	0	0	0

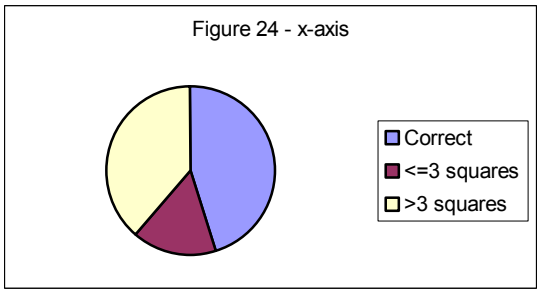
Figure #20: Reverse landmark recognition testing results

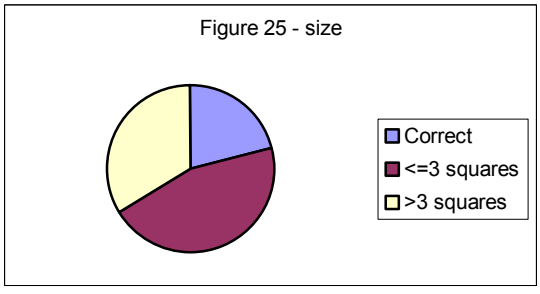
Predicted			Actual			$(p-a)^2$			$\Sigma(p-a)^2$
y_c	x_c	size	y_c	x_c	size	y_c	x_c	size	
8	4	10	8	3	17	0	1	49	50
9	15	12	8	3	15	1	144	9	154
9	11	11	8	3	11	1	64	0	65
8	4	10	10	17	7	4	169	9	182
8	10	11	8	3	10	0	49	1	50
8	4	10	7	4	11	1	0	1	2
8	4	10	7	4	8	1	0	4	5
8	4	10	7	4	8	1	0	4	5
8	4	10	7	4	8	1	0	4	5
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	5	1	0	25	26
9	16	12	9	14	21	0	4	81	85
8	5	10	9	15	16	1	100	36	137
8	4	10	9	15	10	1	121	0	122
9	15	12	8	3	16	1	144	16	161
8	4	10	8	3	12	0	1	4	5
8	4	10	10	18	6	4	196	16	216
9	14	12	8	3	13	1	121	1	123
8	4	10	8	3	13	0	1	9	10
8	6	10	8	4	10	0	4	0	4
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	5	1	0	25	26
8	4	10	7	4	6	1	0	16	17
8	6	10	9	15	15	1	81	25	107
8	11	10	9	15	11	1	121	1	123
8	5	10	9	13	9	1	64	1	66
8	4	10	8	3	12	0	4	4	5
8	4	10	9	17	10	1	169	0	170
8	4	10	8	3	13	0	1	9	10
8	4	10	8	3	13	0	1	9	10
8	8	11	8	4	11	0	16	0	16
8	4	10	7	4	8	1	0	4	5
8	4	10	7	4	8	1	0	4	5
8	4	10	8	4	6	0	0	16	16
8	4	10	7	4	5	1	0	25	26
8	4	10	7	4	5	1	0	25	26
8	14	12	9	14	20	0	0	64	64
8	7	11	9	14	18	1	49	49	99
9	11	11	9	14	6	0	9	25	34
8	4	10	8	13	11	0	81	1	82

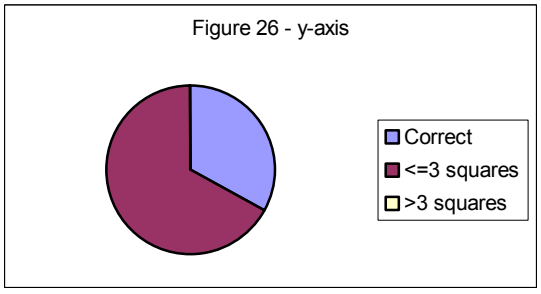
Figure #21: Arbitrary start forward landmark recognition testing results

Predicted			Actual			$(p-a)^2$			$\Sigma(p-a)^2$
y_c	x_c	size	y_c	x_c	size	y_c	x_c	size	
8	4	10	8	4	6	0	0	16	16
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	8	1	0	4	5
8	4	10	7	4	8	1	0	4	5
8	6	10	7	4	10	1	4	0	5
8	6	10	7	4	11	1	4	1	6
8	4	10	10	18	6	4	196	16	216
8	4	10	8	4	6	0	0	16	16
8	4	10	7	4	7	1	0	9	10
8	4	10	7	4	8	1	0	4	5
8	5	10	7	4	8	1	1	4	5
8	4	10	7	4	10	1	0	0	1
8	7	11	8	3	13	0	16	4	20
8	4	10	8	3	13	0	1	9	10
8	4	10	7	4	6	1	0	16	17
8	4	10	7	4	8	1	0	4	5
8	4	10	7	4	8	1	0	4	5
8	4	10	7	4	8	1	0	4	5
8	6	11	7	4	9	1	4	4	9
8	5	10	7	3	10	1	4	0	5
8	5	10	8	3	10	0	4	0	4









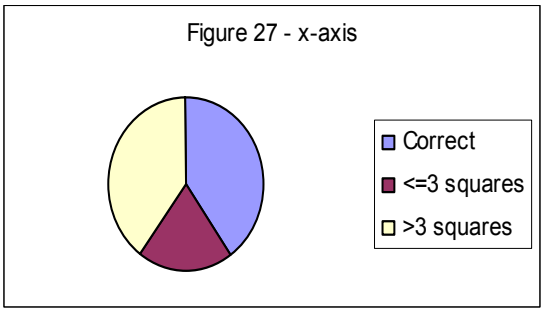


Figure 27 - x-axis
Correct
<=3 squares
>3 squares

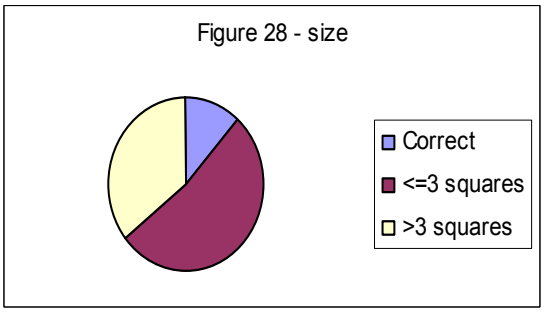


Figure 28 - size
Correct
<=3 squares
>3 squares

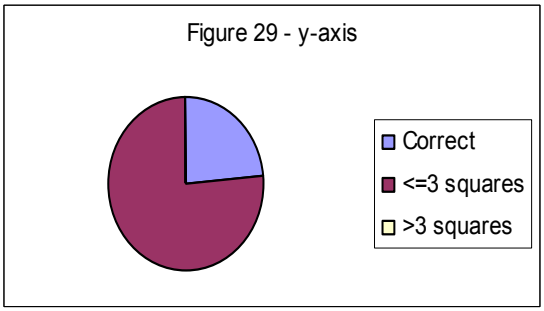


Figure 29 - y-axis
Correct
<=3 squares
>3 squares

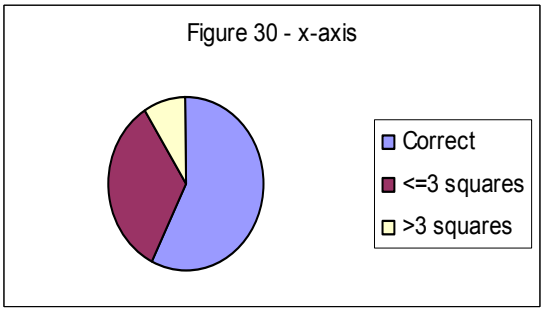


Figure 30 - x-axis
Correct
<=3 squares
>3 squares

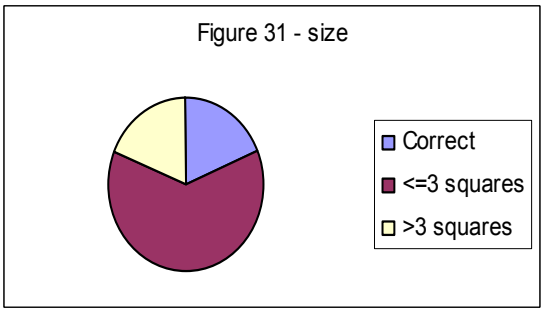


Figure 31 - size
Correct
<=3 squares
>3 squares

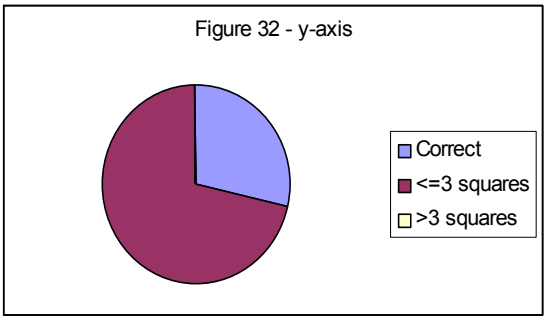


Figure 32 - y-axis
Correct
<=3 squares
>3 squares

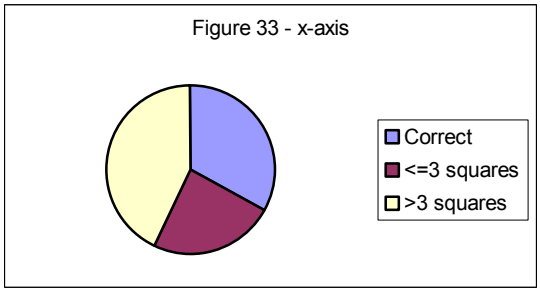


Figure 33 - x-axis
Correct
<=3 squares
>3 squares

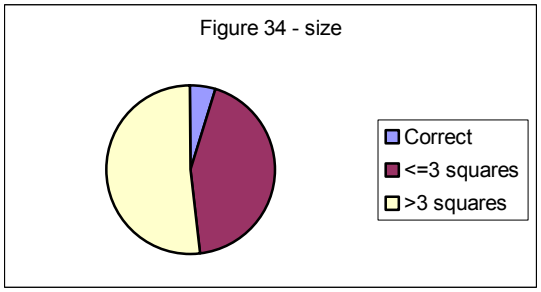


Figure 34 - size
Correct
<=3 squares
>3 squares

Figure #22: Arbitrary start reverse landmark recognition testing results

Predicted			Actual			$(p-a)^2$			$\Sigma(p-a)^2$
y_c	x_c	size	y_c	x_c	size	y_c	x_c	size	
8	4	10	11	15	7	9	121	9	139
8	4	10	7	4	5	1	0	25	26
9	15	12	9	14	21	0	1	81	82
8	8	11	9	15	15	1	49	16	66
9	15	12	9	15	11	0	0	1	1
9	12	11	8	13	11	1	1	0	2
8	4	10	9	13	7	1	81	9	91
8	4	10	8	4	6	0	0	16	16
8	4	10	7	4	5	1	0	25	26
8	4	10	7	4	5	1	0	25	26
8	7	11	9	15	15	1	64	16	81
8	4	10	9	14	6	1	100	16	117
9	11	11	8	13	12	1	4	1	6
8	5	10	9	13	7	1	64	9	74
8	4	10	8	4	6	0	0	16	16
8	4	10	7	4	7	1	0	9	10
8	5	10	9	15	14	1	100	16	117
8	8	11	9	14	17	1	36	36	73
8	6	10	9	14	7	1	64	9	74
8	10	11	8	12	10	0	4	1	5
8	11	11	8	13	8	0	4	9	13

6.0 Conclusions:

This paper has presented a method for learning to predict image-based landmarks for wayfinding using a multilayer neural network. The method was evaluated by creating a training course, collecting training data, and using that data to train the neural network to predict landmarks.

The experimental results indicate that this is a very feasible method of predicting landmarks. Current experiments have shown good results, but with a few outlying results. One approach to address improve performance would be to train the network on more landmark data. Our goal is to construct a system that continually learns landmark positions and locations.

The landmarks used in this example were fairly simple. Our goal was to apply this qualitative, image-based approach to landmark recognition in a Search and Rescue application. Our next step is to use landmarks that are more representative of that applications.

To increase performance, a Beowulf cluster can also be used not only to learn the appropriate weights for the neural network, but also to predict the next landmark location. A socket can be opened between the PC running the robot and the Beowulf cluster, thus allowing transmission of data between components.

Finally, this work has just used visual information. However, we believe that a combination of sonar and visual information would be more useful for representing and predicting the kind of landmarks we expect to see in our application.

7.0 References:

[1] C. Olson, *Landmark Selection for Terrain Matching*, IEEE Int. Conf. On Robotics and Automation, San Francisco, CA, 2000
[2] R. Arkin, *Behavior-Based Robotics*, MIT Press, Cambridge, MA, 2000
[3] R. Sim and G. Dudek, *Learning Visual Landmarks for Pose Estimation*, Centre for Intelligent Machines, McGill University, Montreal, Canada, 1999
[4] M. Mata, J. M. Armingol, *Learning Visual Landmarks for Mobile Robot Navigation*, Division of Systems Engineering and Automation, Madrid, Spain, 2002
[5] G. Bianco, A. Zelinksy, M. Lehrer, *Visual Landmark Learning*, Verona, Italy; Canberra, Australia; Zurich, Switzerland, 2000
[6] S. Thrun, *Bayesian Landmark Learning for Mobile Robot Localization*, Computer Science Department and Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 1997
[7] C. Wellington, A. Stentz, *Online Adaptive Rough-Terrain Navigation in Vegetation*, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2004
[8] L. Fausett, *Fundamentals of Neural Networks*, Prentice-Hall, Englewood Cliffs, NJ, 1994
[9] A. Jacoff, E. Messina, J. Evans, *A Standard Test Course for Urban Search and Rescue Robots*, Performance Metrics for Intelligent Systems Workshop, 2000