Vision Processing for Robot Learning*

Ulrich Nehmzow
Department of Computer Science
Manchester University
Manchester M13 9PL, UK
ulrich@cs.man.ac.uk

Abstract

Robot learning — be it unsupervised, supervised or self-supervised — is one method of dealing with noisy, inconsistent, or contradictory data that has proven useful in mobile robotics.

In all but the simplest cases of robot learning, raw sensor data cannot be used directly as input to the learning process. Instead, some "meaningful" preprocessing has to be applied to the raw data, before the learning controller can use the sensory perceptions as input.

In this paper, two instances of supervised and unsupervised robot learning experiments using vision input are presented. The vision sensor signal preprocessing necessary to achieve successful learning is also discussed.

1 Background

By virtue of their ability to change location, mobile robots are exposed to noisy, inconsistent, or contradictory sensory perceptions, to a larger degree than, for instance, stationary robots. Any sensor signal processing mechanism that can cope with such data is therefore particularly suitable for mobile robot control.

Artificial neural networks have been shown to be a suitable mechanism for this purpose (e.g. [Nehmzow 92, Zalzala & Morris 96]).

In this paper, we discuss two experimental scenarios, in which a Nomad 200 mobile robot (see figure 1) acquires fundamental sensory-motor competences through neural network learning, using input from a CCD camera.

In the first example, supervised teaching is used to train the robot to avoid obstacles, follow walls and traverse corridors. Robot training is a fast and reliable control method ([Nehmzow 95b]), and the experiment presented here demonstrates that it can successfully be applied to vision data.

In the second experiment, vision data is clustered autonomously by the robot, using unsupervised neural network learning, to differentiate between images containing boxes and those not containing boxes.

Both experiments share similar image preprocessing procedures, and would most probably not have succeeded without those preprocessing methods. The paper discusses these vision data preprocessing methods in detail.

2 Experimental Scenario

2.1 The Manchester Mobile Robot *FortyTwo*

All experiments reported here were conducted at Manchester University, using a Nomad 200 mobile robot called *FortyTwo* (see figure 1).

The robot is a hexagonal robot, 50 cm in diameter and 80 cm high (weight 59 kg). It is equipped with sixteen sonar range finding sensors (range 15 cm to 6.50 m), sixteen infrared

*J. Industrial Robot, Vol. 26, No. 2, pp. 121-130, 1999.

Figure 1: THE MANCHESTER MOBILE ROBOT *FortyTwo*.

proximity sensors (active range, depending on surface colour, approximately 60 cm), twenty tactile bumper sensors and a monochrome camera (490 x 512 pixels) with frame grabber.

The robot is controlled by a distributed controller, the master being a 486 PC (66 MHz, 20 MIPS). Sensor modules are controlled by a 68HC11F1 microprocessor.

Turret and base rotation in FortyTwo are independent, the maximum speed of the robot is 0.5 ms^{-1}.

3 Acquiring sensor motor competences from vision input, using supervised learning

3.1 Motivation

Provided mathematical models of a plant that is to be controlled are available, classical control methods or adaptive control can successfully be applied (see, for instance, [Åström 89]). Speed control, orientiation control and force control are but three examples that come to mind.

If, on the other hand, robots are to be used in unstructured environments, if models of robot and environment are only partially or not at all available, reinforcement learning techniques provide a viable, if not necessary alternative ([Barto 89, Sutton 91]. Earlier experiments have demonstrated that such techniques enable robots to learn from reinforcement, successfully and in real time ([Nehmzow 94, Nehmzow 95a]).

The following experiments have taken our previous approach to robot control ([Nehmzow 92]) — *autonomous* competence acquisition by means of connectionist computing — further: feedback for the reinforcement learning process is provided by the user, allowing the robot to acquire task achieving competences through observation of a human operator.

This approach is advantageous for industrial applications. Because the robot can very quickly be trained to perform the desired task, the same robot can be used for a variety of different tasks without the need for (expensive) re-programming, thus reducing setup costs. Contrary to reinforcement learning architectures enabling autonomous competence acquisition (for example [Mahadevan & Connell 91, Kaelbling 91, Nehmzow 92]), learning here is supervised by a human operator, an aspect that can be relevant regarding safety aspects of industrial applications.

3.2 Learning through Supervision

The central element of the controller used in the experiments presented here is an associative memory (see figure 2), implemented by means of an artificial neural network. It associates incoming (possibly preprocessed) sensor signals (input vector $\vec{\imath}$) with the robot's motor actions (outputs o_j):

$$o_j = \vec{w_j} \cdot \vec{\imath}, \tag{1}$$

with $\vec{w_j}$ being the weight vector of output unit j.

Training this network is achieved through a supervised learning process, in which the operator's commands (issued by joystick control) are used to generate the necessary teaching signal τ (see equation 2).

$$\vec{w_j}(t+1) = \vec{w_j}(t) + \eta(t)(\tau_j - o_j)\vec{\imath}. \tag{2}$$

$\eta(t)$ is the learning rate (or "gain"), specifying the degree of change applied at each learning step (η was initially set to 0.3 in our experiments, and reduced by 1% each learning step).

As a result of the training process, meaningful associations between the robot's sensory input and its actuator responses develop. The generalisation property of artificial neural networks is exploited here: the developing links generate meaningful response patterns of the robot even in situations not encountered before.

3.3 Vision Data Preprocessing

Previous experiments in supervised learning with the controller shown in figure 2 (e.g. [Nehmzow 95b]) have shown that the robot is able to acquire fundamental sensor-motor competences such as obstacle avoidance, wall following, box pushing, box clearing and floor cleaning, using sonar, infrared or tactile sensors. We were now interested to acquire such competences using vision data, which is more complex than the range data obtained from either sonar or infrared sensors.

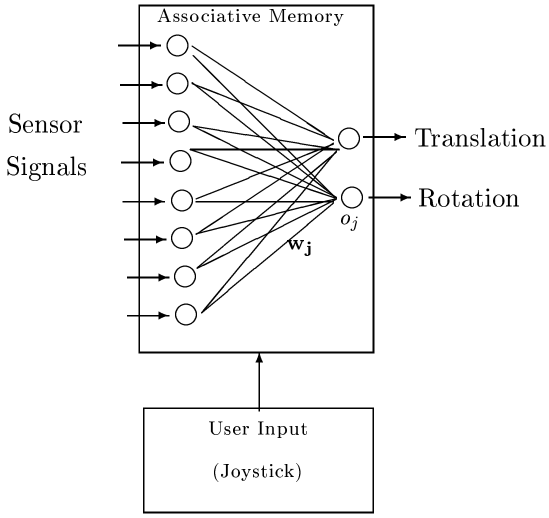

Figure 2: THE GENERAL CONTROLLER ARCHITECTURE

The information content of visual data, regarding fundamental tasks like obstacle avoidance, is far less accessible than that of, for instance, sonar range data. Whilst the latter provides direct indications as to the presence and distance of objects, the former merely supplies a distribution of grey level values over the visual field of the camera.

In order to overcome this fundamental problem of image processing, the information coming from the CCD camera was preprocessed before being used as input to the artificial neural network. These preprocessing stages, some of which have counterparts in biological vision systems (for example edge detection ([Marr & Hildreth 80]) and orientational selectivity ([Campbell & Robson 68])), are described in the following sections.

The complete vision preprocessing system is shown in figure 3, sample images resulting from the vision data preprocessing are shown in figure 4.

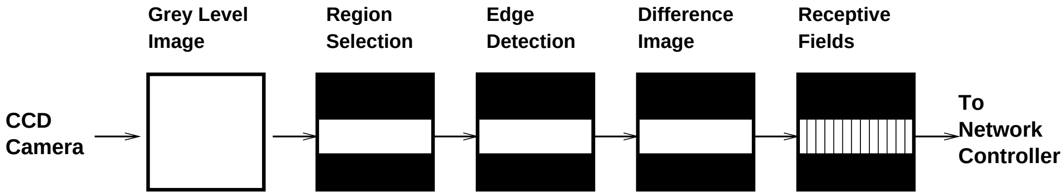

Figure 3: THE VISION PREPROCESSING SYSTEM.

3.3.1 Narrowing the Field of View

In the first stage, the amount of data is reduced by limiting the field of view (see, for example, [Dickmanns & Graefe 88] for an application of this to vehicle control).

We assumed here that the robot operates in environments containing tall objects, rather than small objects on the floor or objects suspended from the ceiling — a justified assumption for environments containing walls, doors, furniture and people. This allows the field of view to be limited to the central 20% of the image, discarding the top and bottom 40% of the data supplied by the CCD camera. The resulting image size is 320 x 50 pixels. Obviously, for some applications (like the detection of descending stairs) other windows may have to be chosen.

3.3.2 Detecting Vertical Edges

To control robot motion, texture and shade of objects are less useful than position and size of objects. Therefore, in a second stage of visual data preprocessing texture and shade information are discarded, and vertical edges in the image only are retained. This is achieved by applying an edge detecting operator as shown in figure 9.

Based on the assumption that every object that is to be detected by the vision system has some vertical edges (boxes, people, tables, chairs, doors and walls all fulfill this assumption), not only texture and shade information are eliminated, but also information about horizontal edges, which further reduces the amount of data to be processed.

3.3.3 Differentiation

For applications of computer vision to robot control *distance information* is the most useful. The *optical flow*, i.e. the movement of pixels resulting from robot motion, can be used to compute distance to an object provided the *correspondence problem* (identifying the pixel belonging to the same object in subsequent frames) is solved (see, for instance, [Marr 82, p.212ff]). This, however, is difficult, particularly if the environment the robot is operating in is (partially) unknown. As a vision preprocessing system, generating sensor information for the artificial neural network used in the controller, however, optical flow can be used *without* having solved the correspondence problem. By computing the difference between subsequent frames (whilst the robot is in forward motion) changes in the image will correspond with distance to objects (far away objects will generate little optical flow, whilst nearby objects will generate a large one). This information is sufficient for the network to acquire useful sensor-motor couplings.

Therefore, a differentiation is performed in the following manner: If the difference between pixel grey levels in subsequent frames exceeds a threshold of eight, the respective pixel is set to '1', otherwise set to '0' (the threshold value of eight was arrived at by experimentation, and chosen such that small fluctuations due to camera noise fall below the threshold). This provides sufficient immunity against background noise whilst still detecting changes between subsequent frames reliably.

3.3.4 Receptive Fields

The final input vector to the control system is generated by dividing the image of 320 x 50 pixels into 16 receptive fields of 20 x 50 pixels each.

The number of '1' pixels in each receptive field is determined (see figure 4), the whole input vector (consisting of 16 integer values) is then normalised and presented to the robot controller (figure 2).

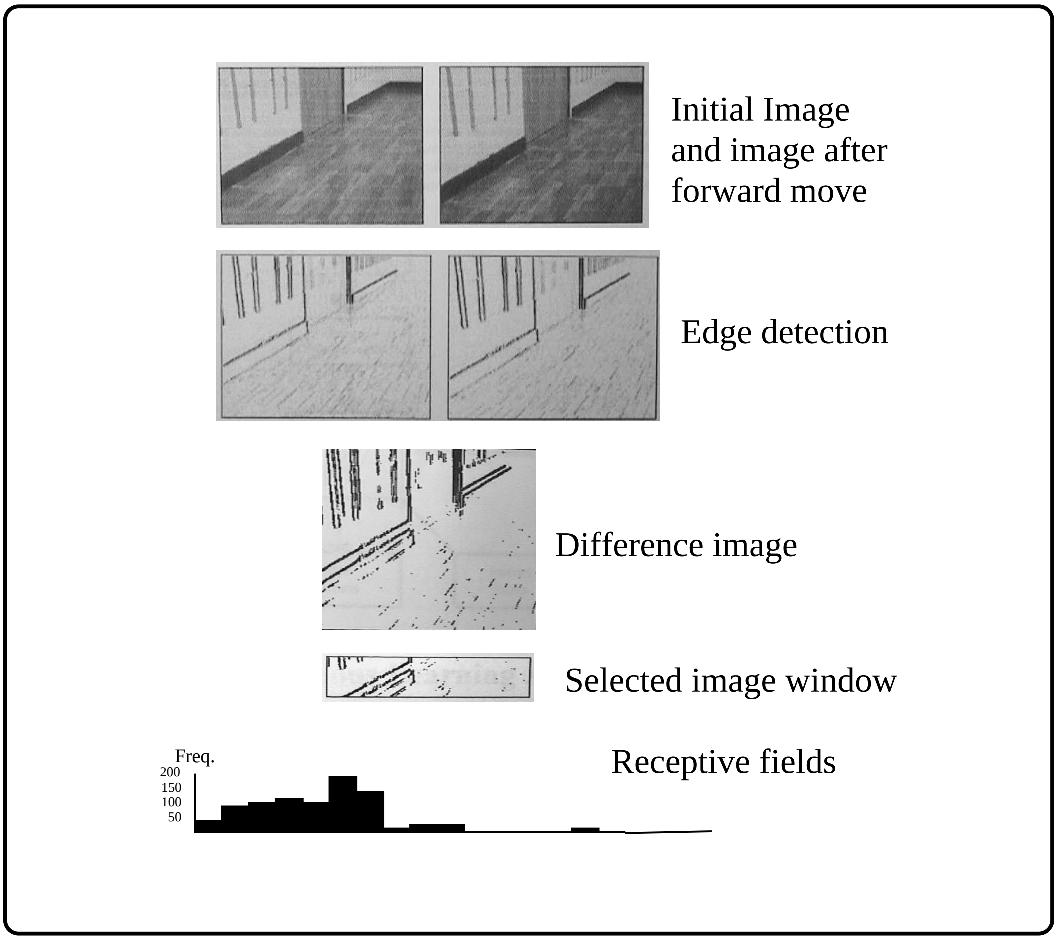

Figure 4: RESULTING IMAGES AFTER EACH STEP OF VISION DATA PROCESSING (CF. FIG 3).

One important property of the vision preprocessing system discussed here is that it is computationally cheap and can successfully be used on a 486 PC, in real time. The maximum speed of the robot, under visual guidance, was 37 cms^{-1}. (Kosaka and Kak, for example, report mobile robotics experiments on vision and model based reasoning for landmark recognition, in which the robot travels at 15 cms^{-1} ([Kosaka & Kak 92])).

3.4 Obstacle Avoidance

The following experiments were conducted in a room with white walls, which were covered with dark vertical strips at approximately 30 cm intervals, to allow visual detection.

In the obstacle avoidance experiment, the camera was mounted facing the direction of travel, pointing downwards at an angle of about 30 degrees. The camera was focussed at approximately 75 cm. Only preprocessed vision data (see figures 3 and 4) was used in this experiment, infrared and sonar sensors were not used.

The robot was trained to approach an obstacle (a stack of cardboard boxes) and to pass it either on the left or the right hand side (see figure 5). The robot learned to perform this task autonomously after about 10 learning steps.

Because vision input is only generated when subsequent frames differ, the robot learned to perform additional rotational movements as well as translational ones when approaching an obstacle. This emergent motion would generate suitable visual stimuli for the vision preprocessing system to generate input vectors to the network. Once the object was sufficiently clearly situated on one side of the robot, *FortyTwo* would pass it using translational movements only.

Depending on the sensory stimuli received, *FortyTwo* moved at varying speeds whilst approaching and avoiding the obstacle, the average speed being about 5 cms^{-1} and the maximum speed being 37 cms^{-1} (the robot rarely travelled at this speed).

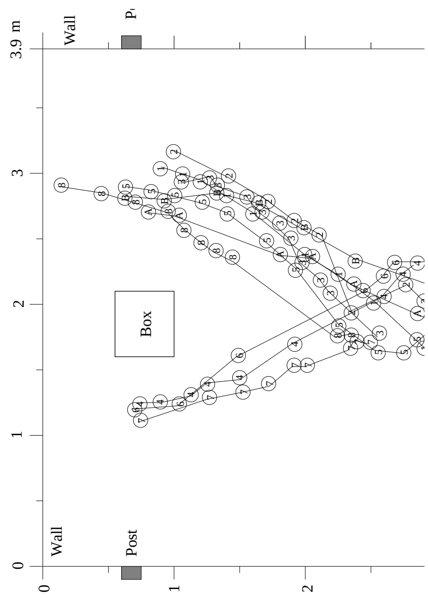

Figure 5: Obstacle Avoidance, using optical input. Distances shown are in metres.

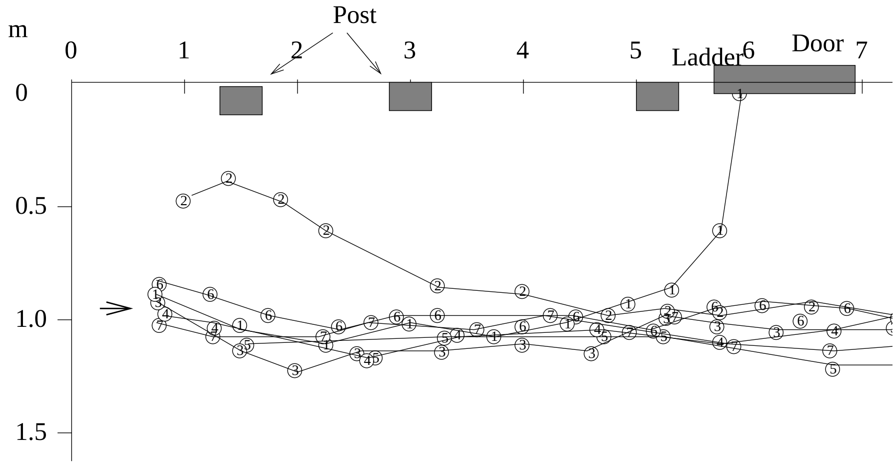

Figure 6: Wall-following, using optical input.

3.5 Wall Following

In this experiment, the camera was placed pointing about 20 degrees downwards, and about 20 degrees to the left of the center of the robot. The other camera parameters were set as before.

The robot was trained to maintain a distance of about 70 cm to a left-hand wall, using preprocessed visual data only. The robot acquired this competence after about 10 learning steps and was then able to move and remain parallel to the wall. However, if the visual input changed considerably from that observed during training (i.e. a sideways view of the wall) no successful wall following was possible. In one of the experiments the robot began to face the wall directly, and became unable to resume a parallel path to the wall (see figure 6). This was due to the fact that the robot had not been trained to avoid head-on collisions, but only to stay parallel to a left-hand wall.

Depending on the sensory stimuli received, *FortyTwo* moved at varying speeds whilst following a left hand wall. The average speed was approximately 6 cms^{-1}, the maximum speed 9 cms^{-1}.

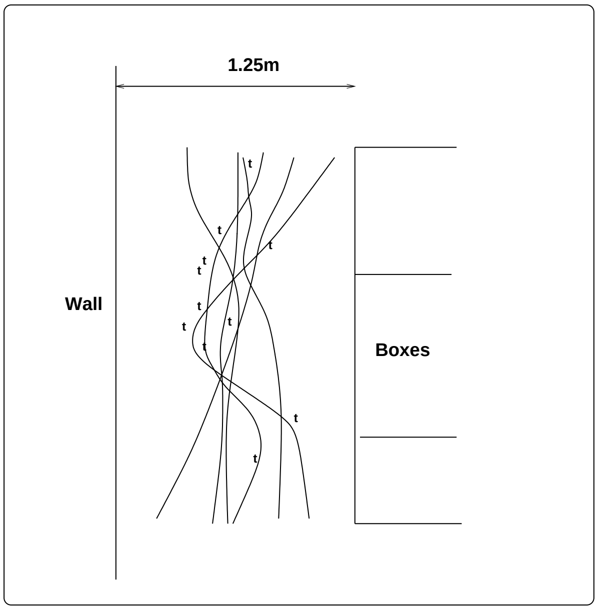

Figure 7: Corridor-following, using optical input. 't' indicates locations where teaching occurred.

3.6 Corridor following

Using the same vision preprocessing as described in section 3.3, the robot learned to move down a corridor formed by a wall on one side, and boxes on the other. The width of the corridor was 1.25m, and the camera was focused at approximately 75 cm.

Using an *autonomous* learning procedure (as described in [Nehmzow 94b]), the robot acquired the corridor following behaviour in 14 learning steps. Some typical trajectories are shown in figure 7.

4 Clustering and concept formation, using unsupervised learning on vision data

For many robotics applications, for example those of object identification and object retrieval, it is necessary to use internal representations of these objects. These models — abstracted (i.e. simplified) representations of the original (i.e. the aspect of the environment that is to be modelled) — are to capture the essential properties of the original, but to eliminate unnecessary detail.

The "essential properties" of the original are not always directly available to the designer of an object recognition system, nor is it always clear what constitutes unnecessary detail. We contend that in these cases the only feasible method is that of model acquisition, rather than model installation.

In a second series of experiments, therefore, we used a self-organising structure that acquired models in an unsupervised way through robot-environment interaction. User predefinition was kept at a minimum.

In this particular instance, a mobile robot's task was to identify boxes within its visual field of view, and to move towards them. No generic model of these boxes was used, instead a model was acquired through a process of unsupervised learning in an artificial neural network.

4.1 Experimental Scenario

In the experiments, the robot's task was to determine whether or not a box was present in its visual field, and, if present, to move towards the box. The camera was the only sensor used. The boxes carried no particular distinguishing features; boxes were chosen as target objects for experimental reasons only, rather than for any of their specific properties.

A trainable object recognition system, based on a self-organising feature map, was used to achieve this.

4.1.1 Vision Data Preprocessing System

The complete vision data preprocessing system is shown in figure 8.

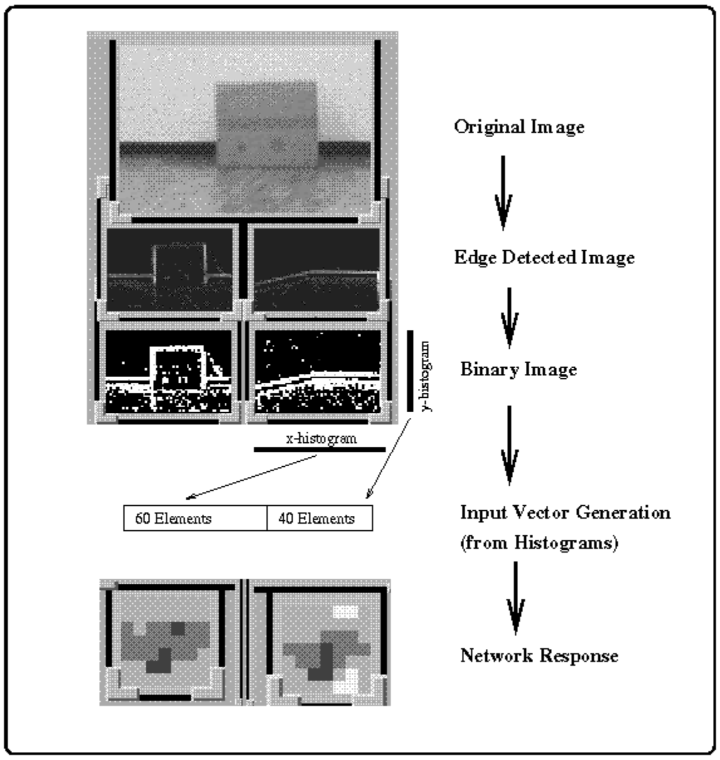

Figure 8: THE ENTIRE BOX RECOGNITION SYSTEM.

As a first processing step, the raw grey level image of 320 by 200 pixels was reduced to 120 by 80 pixels by selecting an appropriate window that would show the target object at a distance of about 3m.

The reduced grey level image was then subjected to a convolution with the edge-detecting template shown in figure 9, where each new pixel value is determined by equation 3.

-1	-1	0
-1	0	1
0	1	1

a	b	c
d	e	f
g	h	i

Figure 9: EDGE DETECTING TEMPLATE.

$$pixel = |(c + f + i) - (a + d + g)| \tag{3}$$

The edge-detected image was then coarse coded by averaging the pixel values of a 2x2 square, yielding an image of 60 by 40 pixels.

Following the coarse coding stage, we computed the average pixel value of the entire image, and used this value to generate a binary image (such as the one shown in figure 10) by thresholding.

Finally, by computing the histogram along the vertical and horizontal axis of the binary image, we obtained a 60+40 element long input vector, which was used as input to our box detection algorithm. One such input vector is shown in figure 11.

Figure 10: BINARY IMAGE.

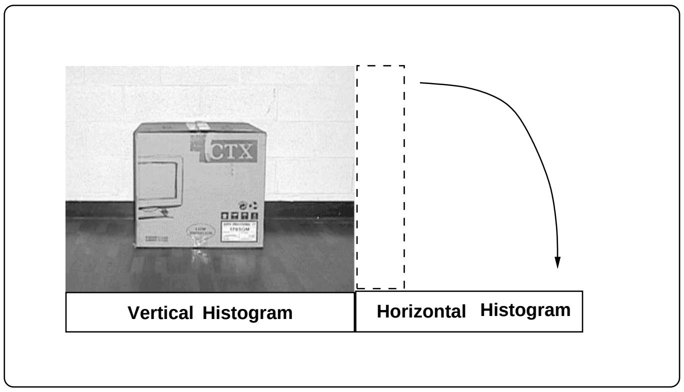

Figure 11: INPUT TO THE BOX-DETECTING SYSTEM.

4.2 The Box Detection Algorithm, using Visual Input

We then used the 100-element long input vector described above as an input to a self-organizing feature map (SOFM) of 10 by 10 units ([Kohonen 88]).

The SOFM is an Artificial Neural Network that performs a topological clustering of its input data using an unsupervised learning mechanism. The network consists of one layer of cells typically arranged as a two dimensional grid. Figure 12 shows the basic structure of such a network.

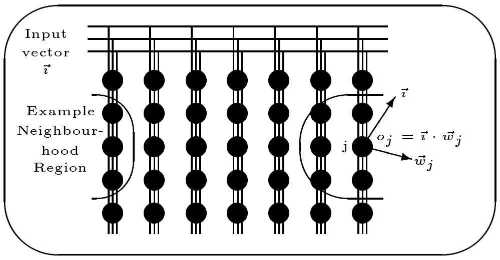

Figure 12: STRUCTURE OF THE SOFM

Each unit of the network receives the same input pattern (in this case consisting of preprocessed vision data). This pattern is presented to the network as a vector:

$$\vec{\imath} = [i_1, i_2, ..., i_n]$$

These inputs are generated as the robot is exploring its environment. The weights on the connections to a single output unit are given by:

$$\vec{w_j} = [w_{j1}, w_{j2}, ..., w_{jn}]$$

where j identifies unit j within the output layer and n is the nth element of the input vector. These weight vectors are normalised to unit length (see [Kohonen 88]).

To find the output o_j of unit j we calculate the weighted sum of its inputs:

$$\sum_{k=1}^{n} w_{jk} i_k = \vec{w_j} \cdot \vec{\imath} \tag{4}$$

with n being the number of elements in the input vector.

In its initial state each cell of the network has a unique set of weights. Therefore, when presented with an input vector, one cell will respond more strongly than the others.

The weight vector of this "winning" unit along with its eight neighbours[1] is updated according to equation 5:

$$\Delta w_{jk} = \alpha(i_k - w_{jk})$$

and

$$w_{jk}(t+1) = w_{jk}(t) + \Delta w_{jk}, \tag{5}$$

where α is the learning rate parameter. The learning rate in our experiments was set to 0.8 for the first ten learning steps, after that to 0.2 for the entire remaining training period. Weight vectors were normalised again after being adjusted.

As this process continues the network organises into a state whereby dissimilar input vectors/patterns map onto different regions of the network, whilst similar patterns are clustered together in groups. In using the SOFM to process the robot's sensor signals, distinct sensory perceptions (with distinct 'perceptual signatures' - the sensory patterns represented in the input vectors) will map onto distinct regions of the network, with similar perceptual patterns clustering together in a region. In this way, regions of the network can be seen as representing sensory perceptions within the robot's environment.

4.3 Experimental Results

A test set of sixty images (thirty with boxes, thirty without — see figure 13) was used to train the network, and evaluate the system's ability to differentiate between images containing boxes, and those not containing boxes.

As can be seen in figure 13, the network's response in both cases is similar, but not identical. The difference in network response can be used to classify an image.

Fifty images of aligned boxes were used for the training phase of the network, of the remaining ten test images all were correctly classified by the system.

In a second set of experiments, boxes were placed in *various* positions and angles. The training set consisted of 100 images, the test set contained 20 images.

In the case of images showing boxes, 70% of all test images were classified correctly, 20% were wrong, and 10% were "not classified". Of the images not showing boxes, 60% were classified correctly, 20% were incorrectly classified and 20% were not classified.

To assess the ability of the system to classify images under more "realistic" situations, we conducted a third set of experiments, in which images were used that were similar to those in the previous experiment, i.e. they contained images of boxes in varying positions and angles. In addition to this, images of stairs, doors and other objects that had similarities with boxes were included ("difficult" images).

The training set consisted of 180 images, the test set comprised 40 images. 70% of all "box" images were classified correctly, 20% were incorrectly classified and 10% were not classified at all.

Of the "no box" images, 40% were classified correctly, 55% were incorrectly classified as "box" and 5% were not classified.

4.4 Associating Perception with Action

Having established that the system is indeed able to distinguish static images containing boxes from those not containing boxes, we were interested to use the system to guide the robot towards boxes, if any were identified within the image.

[1] This neighbourhood was static throughout the entire training period.

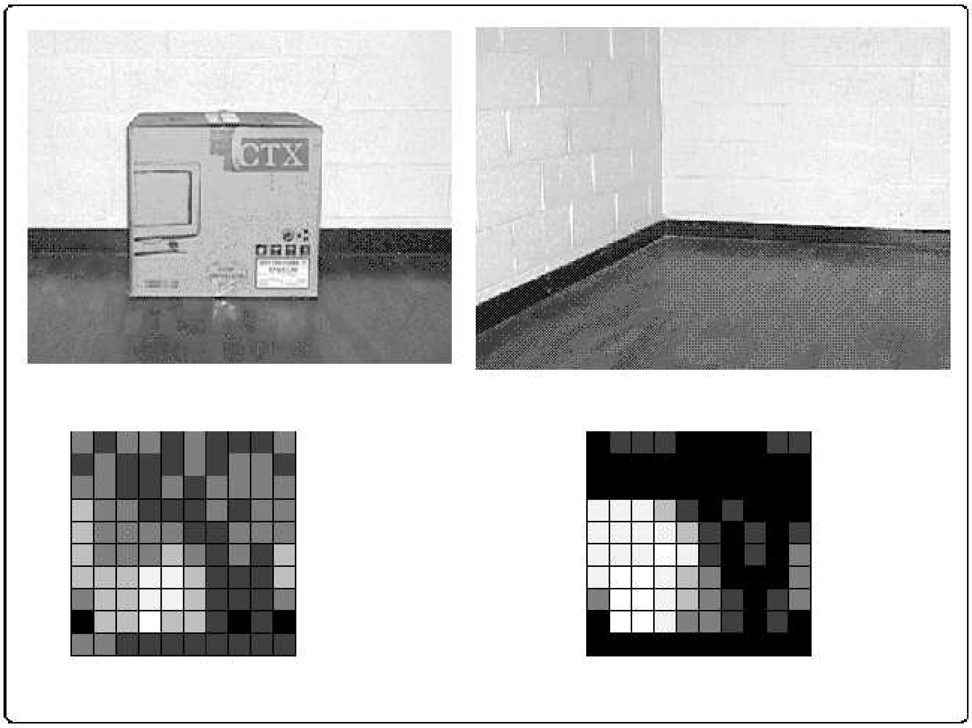

Figure 13: Two example images and the network's response to them. The colour coding indicates the strength of a unit's activation.

In SOFMs, this can be achieved by extending the input vector (as described above) by adding an action component to it. The entire input vector (and therefore the weight vector of each unit of the SOFM) now contains a perception-action pair.

In the training phase, the robot is driven manually towards a box in its field of view. Input vectors are generated by combining the preprocessed vision input and the user-supplied motor command.

In the recall phase, the robot is then able to move towards a box autonomously, by determining the winning unit of the SOFM (i.e. that unit that resembles the current visual perception most closely), and performing the motor action associated with that unit.

Our observation was that *FortyTwo* was well able to move toward a single box in its field of view, regardless of orientation of the box, or initial orientation of the robot. As the robot approached the box, lateral movements of the robot decreased, and the approach became faster and more focussed, until the box filled the entire field of view, and thus became invisible to the system. The robot approached boxes reliably under these conditions.

However, the robot could get confused by other box-like objects in the field of view (like the stairs in our robotics laboratory). In this case, the robot would approach the misleading object in question, to abandon it later when the error was detected. At this stage, however, the robot was often no longer able to detect the original box, because it had moved too far off the direct approach route.

5 Conclusions

The experiments described in this paper are examples of machine learning applications in mobile robotics, using vision input. They demonstrate that a robot can be taught fundamental sensor motor competences such as obstacle avoidance, wall following or corridor following, using input from a CCD camera.

They also demonstrate that through an unsupervised learning process (i.e. without user intervention) the robot is able to cluster images into appropriate categories — in our case "box" and "no box". These perceptual concepts can subsequently be used for motion control.

Both examples of machine learning applications to mobile robotics show that vision data preprocessing is necessary, before the learning stage can be entered. In both experiments we applied similar methods to reduce the amount of vision data: selecting a suitable field of view, detecting edges (because these are the relevant asppects for most sensor-motor behaviours), coarse coding, and finally histogram construction.

Using these vision processing methods, we reduced the amount of input data from 64000 to 15 and 100 values respectively, without compromising the robot's ability to perform the respective tasks.

Further to the understanding of video image processing necessary to make learning algorithms succeed, the experiments presented here (and elsewhere — see, for instance [Nehmzow 94b, Nehmzow 95b]) demonstrate that robot learning is a fast and successful method of establishing fundamental sensor-motor competences in mobile robots.

Our second set of experiments demonstrates that, provided suitable image preprocessing is applied, self organising neural networks can be used to learn simple concepts such as "box" and "no box". Whether this approach can be extended to more complex concepts (e.g. "people") is subject to ongoing research.

Acknowledgments

The experimental work described in section 3 was conducted by Paul Martin, the experiments described in section 4 were conducted by James Elson. I thank both for their contribution.

References

[Åström 89] Karl Johan Åström, *Toward Intelligent Control*, IEEE Control Systems Magazine, April 1989.

[Barto 89] A. Barto, *Connectionist Learning for Control: An Overview*, Department of Computer and Information Science, University of Massachusetts at Amherst, COINS technical report 89-89.

[Campbell & Robson 68] F.W.C. Campbell and J. Robson, *Application of Fourier analysis to the visibility of gratings*, J. Physiol. (London) 197, 551-566.

[Dickmanns & Graefe 88] Ernst Dieter Dickmanns and Volker Graefe, *Dynamic monocular machine vision*, Universität der Bundeswehr München, Technical Report UniBwM/LRT/WE 13/FB/88-3.

[Kohonen 88] Kohonen, T., *Self Organisation and Associative Memory,* Springer Verlag, Berlin, Heidelberg, 1988.

[Kosaka & Kak 92] Akio Kosaka and Avinash Kak, *Fast Vision-Guided Mobile Robot Navigation Using Model-Based Reasoning and Prediction of Uncertainties*, CVGIP: Image Understanding, Vol 56 No. 3, pp. 271-329, 1992.

[Kaelbling 91] Leslie Kaelbling, *An Adaptable Mobile Robot*, in F. Varela and P. Bourgine, *Toward a practice of autonomous systems*, MIT Press 1992, pp. 41-47.

[Mahadevan & Connell 91] Sridhar Mahadevan and Jonathan Connell, *Automatic Programming of Behavior-Based Robots using Reinforcement Learning*, AAAI 1991.

[Marr & Hildreth 80] D. Marr and E. Hildreth, *Theory of edge detection*, Proc. R. Soc. Lond. B 207, 187-217 (after [Marr 82]).

[Marr 82] David Marr, *Vision*, W.H. Freeman and Company, San Francisco 1982.

[Nehmzow 92] Ulrich Nehmzow, Experiments in Competence Acquisition for Autonomous Mobile Robots, PhD thesis, University of Edinburgh 1992.

[Nehmzow 94] U. Nehmzow, *Acquisition of Smooth, Continuous Obstacle Avoidance in Mobile Robots*, in H. Cruse, H. Ritter and J. Dean (eds.). Proc. Workshop "Prerational Intelligence in Robotics", ZIF, University of Bielefeld, 1994, pp. 97-106.

[Nehmzow 94b] Ulrich Nehmzow, *Autonomous Acquisition of Sensor-Motor Couplings in Robots*, Technical Report UMCS-94-11-1, Department of Computer Science, Manchester University.

[Nehmzow 95a] U. Nehmzow, *Flexible Control of Mobile Robots through Autonomous Competence Acquisition*, Trans. Institute for Measurement and Control (IMC), 3/1995, pp. 48-54.

[Nehmzow 95b] Ulrich Nehmzow, *Applications of Robot Training: Clearing, Cleaning, Surveillance*, Proc. International Workshop on Advanced Robotics and Intelligent Machines, Salford, UK, 5.-6.4.95.

[Sutton 91] R. Sutton, *Reinforcement Learning Architectures for Animats*, in J.A. Meyer and S. Wilson (eds), *From Animals to Animats*, MIT Press 1991, pp. 288-296.

[Zalzala & Morris 96] A. Zalzala and A. Morris, *Neural Networks for Robotic Control*, Ellis Horwood, Hemel Hempstead 1996.