Machine Learning, 31, 115–139 (1993)

Learning from Innate Behaviors: A Quantitative Evaluation of Neural Network Controllers

NOEL E. SHARKEY noel@dcs.shef.ac.uk
Department of Computer Science, University of Sheffield, U.K.

Editors: Henry Hexmoor and Maja Matarić

Abstract. The aim was to investigate a method of developing mobile robot controllers based on ideas about how plastic neural systems adapt to their environment by extracting regularities from the amalgamated behavior of inflexible (non-plastic) innate subsystems interacting with the world. *Incremental bootstrapping* of neural network controllers was examined. The objective was twofold. First, to develop and evaluate the use of prewired or *innate* robot controllers to bootstrap backpropagation learning for Multi-Layer Perceptron (MLP) controllers. Second, to develop and evaluate a new MLP controller trained on the back of another bootstrapped controller. The experimental hypothesis was that MLPs would improve on the performance of controllers used to train them. The performances of the innate and bootstrapped MLP controllers were compared in eight experiments on the tasks of avoiding obstacles and finding goals. Four quantitative measures were employed: the number of sensorimotor loops required to complete a task; the distance traveled; the mean distance from walls and obstacles; the smoothness of travel. The overall pattern of results from statistical analyses of these quantities supported the hypothesis; the MLP controllers completed the tasks faster, smoother, and steered further from obstacles and walls than their innate teachers. In particular, a single MLP controller incrementally bootstrapped by a MLP subsumption controller was superior to the others.

Keywords: neural network controllers, machine learning, innateness, biologically inspired robotics, quantification in robotics

1. Introduction

Neural computing techniques are becoming increasingly popular for training controllers for mobile robots (Bekey & Goldberg, 1993, Dorigo, 1996, Sharkey, 1997). Some of the most used methods have been concerned with techniques such as reinforcement learning (Krose, 1995) or evolutionary learning (Nolfi, 1997) that require little *a priori* knowledge of the task domain. Supervised learning, on the other hand, has been neglected because of a belief that the designer must provide precise teaching vectors to train controllers, i.e., the experimenter must understand the domain in enough detail to calculate exactly the correct control signals for every move (Sharkey, 1997 (b)). However, an alternative approach to supervised learning is presented here in which controllers are trained by the system in which they are embedded.

This research began with an inspiration from biological systems that successful adaption may involve the use of innate hardwired behaviors to bootstrap learning in plastic neural nets. For example, simple observation shows that a number of quadrupeds, such as giraffes and zebras, are "born on the run" with jerky behavior that quickly becomes smoother as the animal adapts to its environment. In detailed experimental work, Johnson (Johnson, 1992, Johnson & Bolhuis, 1991) found that newly hatched chicks show a limited range of automatic behaviors or predispositions that are triggered by particular environmental stimuli such as pecking at static objects with certain dimensions and contrast and

running toward warmth. Considering a number of findings on the neural system of the chick, Johnson (1992) proposed two comparatively independent processes; one concerned with predispositions and the other with a learning system sub-served by the neural structure IMHV. The general idea is that the first process ensures that the second learns.

The studies reported here assess the development of the use of prewired or *innate* controllers to *bootstrap* learning in Multi-Layer Perceptrons (MLP) to navigate a mobile robot to goals in an obstacle laden environment (c.f. Nolfi & Parisi, 1997). This task was chosen because of its common use as a behavioral benchmark for controllers (Anderson & Donath, 1990, Donnart & Meyer, 1996, Meeden, 1996, Floreano & Mondada, 1996, Touzet, 1997, Salomon, 1997, del Millan, 1996) and its employment in autonomous robotics for several decades (Walter, 1950).

Given the previous work it seemed reasonable to assume that an already existing controller could be used to train a neural network controller. However, there are two novel questions here. First, could an improvement be gained over the performance of simple innate hardwired reactive controllers by using them to bootstrap learning in neural network controllers? Second, could further improvements be gained by training on the back of previously trained neural network controllers? Such improvements may be possible if the robot's behavior exhibits an underlying systematic aspect when it interacts with the world. It is this systematic aspect that would be extracted by the networks during training.

A MLP, trained with backpropagation, can be used to construct a predictive model of a data generator. If this is a noisy generator then a MLP with an appropriate number of free parameters may capture the regularities in the data and make novel predictions that are within a small margin of error. In these terms, the innate controller may be thought of as a noisy data generator in which behavior considered to be noise in one type of environment may be considered to be appropriate in another. Thus, the systematic aspect of the behaviors of an innate controller will depend on the particular environmental circumstances, e.g., swamp, forest, desert, office building, etc. Seen this way, the problem of finding the systematic aspect of the behavior is analogous to using polynomials to fit curves to noisy data. If the order of the polynomial is too high, e.g., there are as many free parameters as there are data points, then the data will be over-fitted and the approximation to the underlying function (the generalization performance) will be poor. If, on the other hand, the order of the polynomial is at an appropriate level, the curve will pass smoothly through the noisy data and generalize well on novel inputs.

The general behavior of an innate controller may be represented by the function $e_i(x)$, where x can be external and/or internal stimuli. In a particular type of environment only a subset of the behavior may be employed and this can be represented by another (sub) function $e_w(x)$. Unless $e_w(x)$ is optimal, adaption consists of finding a function $a(x)$ that is appropriate for the current environmental circumstance. In order to do this, the learning method for a plastic neural net must find regularities in the behavior generated by $e_w(x)$. That is, the neural net must treat $e_w(x)$ as as if it were $a(x)$ embedded in behavioral noise. In other words, adaption depends on the quality of the network's approximation of $a(x)$.

Two "innate" modules were developed for the current research; one for avoiding obstacles and walls and the other for finding goal locations. These were used individually as controllers, as illustrated in Figure 1(a) or combined in a simplified subsumption architecture control system (Brooks, 1986) as shown in Figure 1(b). The innate obstacle avoidance con-

troller was essentially a localized (reactive) version of the *artificial potential field* method developed by Khatib (Khatib, 1986). That is, obstacles were treated as repellent forces and free space as an attractor, and see also (Kassim & Kumar, 1995). This innate controller was used to bootstrap avoidance learning by having a MLP look over its shoulder whilst driving the robot through a laboratory robot pen. The basic training method is illustrated in Figure 2. The first two experiments present an evaluation and comparison of both controllers.

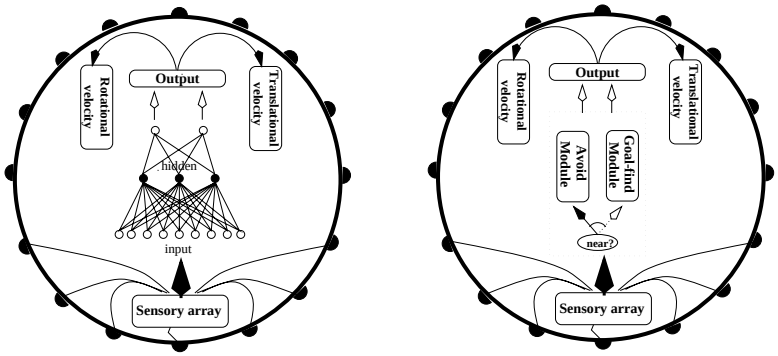

Figure 1. Two diagrams of the robot control structures. The surrounding black circles illustrate the robot body and the closed outer semi-circles illustrate the sonar/infra-red sensors. Only the front 7 pairs of sensors were used as shown by the wiring into the sensory array in the figures. (a) shows a neural network controller for obstacle avoidance and goal-finding. The inputs to the network are the 7 sonar/infra-red pairs and the angle and distance to the goal. The two outputs are for translation and rotational velocity. (b) shows a very simlified subsumption architecture. Which module is in charge is determined by nearness to obstacles. These module could be either hardwired or trained or a mixture.

For the innate goal finding module, a *path integration* or *dead reckoning* method, commonly used by biological systems, was employed to continuously calculate the distance and direction to the goal from the current position. There is substantial evidence that hymenopterous insects such as ants (Wehner et al., 1996) and bees (Esch & Burns, 1996, Srinivasan et al., 1996) and also many mammals (Etienne et al., 1996) use this method to negotiate their environment. However, unlike the robot systems developed here, many of the biological systems use external references such as landmarks and celestial cues and, more controversially, cognitive maps (Gould, 1986, Gallistel, 1990). For example, it appears that hymenopterans use optic flow to measure distance traveled and a solar compass to measure their heading (Wehner, 1992).

The system developed here did not use external references. Instead, its dead reckoning relied exclusively on proprioception by using the movement of the wheels to keep track of angles and distances. Similarly, night active rodents use self-generated information from somatosensory feedback (Etienne et al., 1996). However, according to Etienne and her colleagues (Etienne et al., 1996), when animals use path integration without external reference, it is mainly for short scale navigation as a safeguarding strategy. This has especially high value for young inexperienced individuals or for individuals placed in an unknown environment. Like our innate controller, it, "... seems to depend on a prewired system of information processing that functions automatically whenever the subject lo-

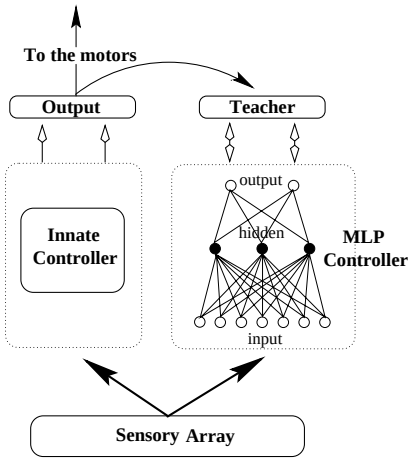

Figure 2. An illustration of the learning method for training a single neural network controller from a hand-coded "innate" controller. The innate controller drives the robot and outputs the teaching signal for the network.

comotes." [(Etienne et al., 1996) page 206]. Over longer range navigation, the absence of external cues for re-calibration means that there can be considerable odometric drift (Benhamou & Bovet, 1990). This is similar to what happens with the odometry of the Nomad 200 robot used in the new research reported here. Owen and Nehmzow (1997) showed that the Nomad 200 odometry drifts during *middle scale* navigation. They found that over four circuits of a route, each approximately 270 feet long, the estimate of the angle could be out by as much as $90°$. However, in the current studies, the maximum distance of a return journey using dead reckoning was 166.25 feet and most of the distances were much shorter. Thus, the Nomad odometry was as reliable as needed for the current task.

At the start of each goal-finding experiment, the robot was presented with a number of goal vectors each giving the direction and distance to the goal without regard to any obstacles in the path and its task was to cycle through the goals and return home. Such a goal vector is believed to be similar to the sort of goal information that is passed on by a bee's dance to let its hive mates know about the location of a single food source. Decades of research (von Frisch, 1967) have shown that the angle between the sun's azimuth and the food source is indicated by the direction of the waggle run and the distance is represented by the number of waggle runs in a unit of time (see also (Esch & Burns, 1996)). Interestingly, it has been argued that, like our robot, the goal vector passed to bees in the dance gives distance and angle information "as the crow flies", i.e., irrespective of obstacles such as mountains (Carthy, 1963)).

Although the robot can be given an individual goal vector via a keyboard rather than by dancing in front of it, the method employed for the experimental work was to place up to 5 goals in the robot's memory such that they were recalled in sequential order as each goal position was reached. This is rather like hymenopterans whose goal-finding behavior is believed to be dictated by internal instructions. As Collett (1996) points out, goal vectors

may be sometimes, particularly on the first trips of the day, recalled from longer-term memory storage.

Finally, it should be noted that no attempt was made here to model precise biological data or to model specific species. Rather, the work grew out of general abstract intuitive notions about animal intelligence and a belief that "nature knows best". It is important, however, to point out where the biological inspirations, analogies, and biological data helped to form the ideas. By being explicit, the correctness or incorrectness of the relationship to natural systems can be discussed and this may help us to move forward towards the longer term goal of imitating natural intelligence.

2. Quantification of performance and statistical evaluation

One of the main problems addressed by the current research was how to evaluate the performance of the MLP controllers. The purpose of training the MLPs was to improve on the performance of their teachers, the innate controllers, and so they must not imitate their behavior too closely. Since the systematic aspect of the behavior of the innate controller is not known *a priori*, it is difficult to find an error measure with which to assess the generalization performance of the nets. A MLP with limited resources will extract a pattern from the noise, but it is not known in advance whether the extracted pattern will lead to improvement or deterioration. That requires evaluation of the performance of the controllers. In standard neural net research on supervised learning the quality of the generalization performance is usually judged by how well a net can imitate "ideal" targets. However, for the tasks employed here, the ideal targets were not known; the MLPs were trained on the less than ideal targets generated by the innate controllers. The approach taken here was to develop quantitative measures that could be used to assess the relative performance of the robot controllers independently of their method of training.

The evaluation of avoidance and goal finding behaviors in modern robotics has most frequently been concerned with the development and testing of a single control system. Thus, often the only evaluation required is simply a demonstration that it finds goals and avoids obstacles, i.e., it works. The demonstrations can take a number of forms such as running the robot in the laboratory for a fixed period of time and counting the number of bumps or remarking on their absence (Brooks, 1986). Another common method is to provide a graphical representations of the robot's trajectory through an obstacle laden environment. In an early example of this method, Walter (1953) placed a candle on top of the robot (turtle) and used a long exposure time in a darkened room to photograph the trajectory of the light attracted obstacle avoidance behavior. More recently, the graphical displays of robot simulators have been used to effectively illustrate behavior in combination with verbal descriptions of how the robot coped with particular environmental features (Anderson & Donath, 1990, Donnart & Meyer, 1996).

These methods of evaluation have been fruitful in demonstrating the properties of individual robot control systems, but now that there is an abundance of controllers that all appear to do the job adequately, we need more precise methods for quantification and analysis of behaviors. Some quantitative measures have already been used to assess or determine the course of learning avoidance and goal-finding. In evolutionary learning, fitness has been determined by parameters such as speed, straight direction, and obstacle avoidance

(Floreano & Mondada, 1996) or speed and maximum sensor distance (Salomon, 1997). In reinforcement learning, Meeden (1996) measured the amount of punishment received by different controllers and used this to assess the progress of learning and the difference between two controllers. Touzet (1997) used two main measures to measure the performance of the controllers: distance to objects over time and the proportion of moves executed by the obstacle avoidance module that receive positive reinforcement. del Millan (1996) used robot steps to measure the length of goal paths.

The purpose of the experiments reported here was to assess the relative performances of three different avoidance and goal-finding controllers and two different avoidance controllers. To evaluate and compare the behaviors, it was important to find dimensions that would indicate the quality of the performance on pertinent aspects of the task and would show up differences between controllers. As a first step, four dimensions were chosen: (i) the speed of completing the task; (ii) how far the robot stays away from obstacles and walls; (iii) the distance traveled in completing the task; and (iv) the smoothness of travel. In order to compare the robot controllers on these dimensions, the following four quantitative measures were employed.

- *Sensorimotor Loop Values* (SLV): The time taken for the robot behaviors is partly dependent on the type of computer used and its load. To get around this, a measure in robot time was used. Each time step of the robot may be characterized by a sense-process-act step in which the incoming sensory stimuli are transformed into motor output. This is called a sensorimotor loop. The sensorimotor loops were equated for time for all of the controllers used (about 0.15 seconds on a machine free of other processes). The number of sensorimotor loops was used to measure how long the robot took, in robot time, to complete a task.

- *Sensor Readings* (SR): For each sensorimotor loop, the mean of the raw sensor (distance) readings for the front seven pairs of IR/sonar sensors was recorded. This mean provides a measure of the mass of obstacles to the front of the robot as opposed to a minimum distance measurement which tells us only about one sensor reading. The mean of these means over an experimental trial provides a measure of the overall distance that the robot maintained from walls and obstacles and tells something about how much the behavior is directly influenced by the environment. Using such means is a slightly crude measure but it is suitable for the current purposes.

- *Distance Traveled*: The distance is measured by integrating the coordinates for each sensorimotor loop. Since only movement through coordinate space for each sensorimotor loop is counted, the distance measurement filters out movement attributable to the robot rotating about its axis.

- *Smoothness*: Observation of the behavior of the controllers used here has shown that a sensory input may result in a rotation of the robot rather than a movement through coordinate space. Thus, the robot may rotate about its axis to a greater or lesser degree as it travels in the presence of obstacles and, especially, when passing between obstacles. Although the number of sensorimotor loops provides a measure (in robot time) of how long the robot takes to complete a task, this could be attributable to either or both the distance traveled or the amount of rotation (as mentioned above, the *distance traveled*

measure filters out the rotations). Dividing the distance traveled by the number of sensorimotor loops provides a measure of how far the robot traveled, on average, for each sensorimotor loop. Thus Smoothness = Distance/Sensorimotor loops. This is not to be confused with the smoothness of trajectory. The higher the smoothness value the lower the amount of rotation.

Data from the bumper sensors were also recorded and the total numbers are displayed in the data tables for each experiment. These data were too sparse to enter into the statistical analyses.

All of the experiments were conducted for several runs of the robot for each controller (10 minimum). For each experiment, the data from the four measures described above were subjected to statistical comparisons. Since all of the quantities were measured on an interval scale, a parametric test, the Student's t statistic[1] was appropriate. This gives the probability of the null hypothesis that two sets of scores belong to the same distribution. A criterion probability value of $p < 0.05$ is a fairly standard level for rejecting the null hypothesis and saying that two sets of scores are significantly different. Correlated t tests were used for the *repeated measures* comparisons in Experiments 1(a), 1(b), 2(a), 3(a) and Independent t tests were used in Experiments 2(b), 2(c), 3(b), 3(c).

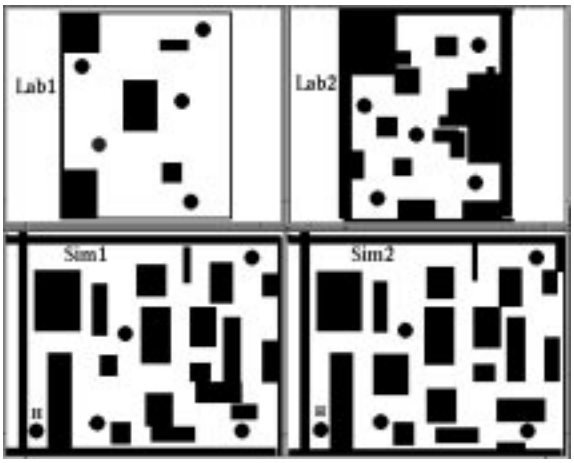

Figure 3. The four simulator worlds used in the three experimental studies. The black rectangles represent wall and boundaries. The filled circles show the goal locations used in the experiments and **H** marks the home position. Reading clockwise from the top left, the figures are, Lab1 world, a laboratory scale model, used in Experiment 1(a), Lab2 world, a laboratory scale model, used in Experiments 2(a) and 3(a), Sim1 world used in Experiments 1(b), 2(b), and 3(b); and Sim2 world used in experiments 2(c) and 3(c).

Three series of experiments in the four different worlds shown in Figure 3 were designed to test the general hypothesis that MLP controllers can outperform their innate 'teaching' controllers. In Study 1, comparisons were made between a MLP obstacle avoidance control module, see Figure 1(a) and an innate avoidance control module used to train it. In Study 2, a simple subsumption architecture was used to combine the goal and avoidance modules into a single control structure as shown in Figure 1(b). The reported comparisons were

between a controller consisting of two MLP modules and its teacher, a controller consisting of the two innate control modules. In Study 3, the performances of the two subsumption controllers, innate and MLP, were compared with a single MLP controller that was trained on the combined task of avoiding obstacles whilst finding goals. Figure 6 shows a trace of the trajectories of all three controllers performing in the Sim1 world. This was incremental bootstrapping. The specific experimental hypothesis was that the single MLP controller would extract the systematic aspect of both the robot's interaction with the word and the interaction of the two behavioral modules in the MLP subsumption controller used to train it.

2.1. The robot hardware and sensors

The Nomad 200 mobile robot is 35 inches high, 18 inches wide and weighs 60 kg. Its dimensions are suited to operation in human like environments such as rooms, hallways, and offices. It is equipped with 16 ultra-sonic or sonar sensors for long range detection (6 to 255 inches), 16 infra-red (IR) proximity sensors (up to 30 inches), 20 tactile sensors, and an odometer to keep track of its position. See Figure 4. All sensors are evenly spaced in circles around the Nomad body. The sonar sensors are situated around the upper body of the Nomad and the IR sensors and bumpers are situated around the lower body.

For all of the experiments described in this paper, only the front seven sonar and seven IR sensors were used. These covered approximately 157° around the front portion of the robot. The output from each sensor pair, s_i, was preprocessed as the reciprocal of the distance to the nearest obstacle. The sensors were read in IR/sonar pairs (one directly above the other) such that if both members of a pair were "on", then the minimum reading was taken, otherwise a reading was taken from whichever of the pair was "on". This greatly increased the chances of detecting all obstacles in the vicinity of the Nomad regardless of their height.

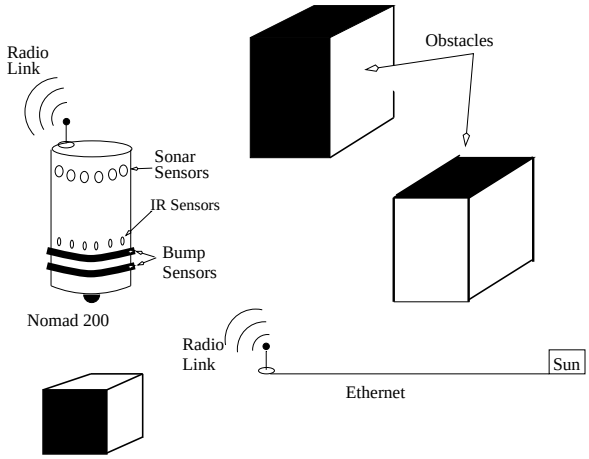

Figure 4. An illustration of the laboratory setup depicting the Nomad 200 robot and some obstacles. The communication between host and robot is also illustrated.

The Nomad 200 was controlled by a Sun workstation through a radio Ethernet link. During operation, the control structure of the Nomad system performed a *sensorimotor* loop that mediated the incoming sensory stimuli and transformed them into motor output. Inputs to the control structure were sensor readings, goal information (user specified xy coordinates), position information (from the odometer) and bumper information. The Nomad robot and its environment were also simulated on the Sun workstation. This simulator provided a flexible way to test programs and record trajectories of the Nomad's behavior. All training reported here was carried out using the Nomad 200 in the "real" laboratory. However, because of the large number of experimental runs, it was more convenient to conduct the quantitative evaluations on the simulator.

3. Avoidance behavior: Innate and Learned

3.1. An Innate avoidance controller

In order to bootstrap the learning of the MLP controllers, the robot was given an *innate* obstacle avoidance behavior. The chosen method was essentially a localized (reactive) version of the *artificial potential field* method developed by Khatib (1986) and see also (Kassim & Kumar, 1995). In this method, obstacles are treated as repellent forces and free space as an attractor. A simple model was used in which each sensor value, s_i, was weighted according to the cosine of the absolute angle between the sensor location and the forward direction of the Nomad. The absolute angles from the front along one side are $0^\circ, 22.5^\circ, 45^\circ, 67.5^\circ$ and the cosines are, respectively, 1, 0.92, 0.71, 0.38. The rotational velocity of the robot was directly related to the force of obstacles. Force was represented by the sum of the sensor readings multiplied by the cosines of their respective angles. An empirically estimated rotational constant, κ_r, was also used, where $\kappa_r = 1000$ for the controller described here. The mapping from the sensory input onto the rotational velocity is given by,

$$R = \kappa_r \left(\pm s_4 + \sum_{i=1}^{3} Cos(\theta_i) s_i - \sum_{i=5}^{7} Cos(\theta_i) s_i \right)$$

where the rotational velocity has a negative weighting for the port sensor readings ($i = 5, 6, 7$), and a positive weighting for the starboard sensor readings ($i = 1, 2, 3$). Thus, the Nomad rotates towards starboard when there is a greater repellent force to the port, i.e., when the sum of the weighted sensor values is greater to port than starboard, and vice versa when the force to starboard is greater. The sign of the front sensor, s_4, depends on the distribution of the forces to port and starboard, i.e., if there is a greater force from obstacles to port, the Nomad should rotate towards the starboard and thus s_4 would be signed negatively.

The translational velocity, T, was obtained by dividing an empirically estimated translational constant, κ_t, by the sum of the weighted sensor values, $T = \kappa_t / \left(\sum_{i=1}^{7} Cos(\theta_i) s_i \right)$. For the present studies, $\kappa_t = 5$. The robot was set to always drive forward so that, in combination with avoiding obstacles, a natural wandering behavior was produced. All of this could be hardwired into a McCullough-Pitts net (McCulloch & Pitts, 1943) with a competitive output.

The performance of the Innate controller was tested extensively in the laboratory pen (19 X 24 feet) and in numerous student demonstrations and exhibitions. It produced some collisions in irregular corners and it could catch its bumper on one obstacle while avoiding another. These occurrences were fairly infrequent and did not happen on every run. Evaluation of the Innate behavior on the simulator is provided in Study 1 and the results are given in Tables 1 and 2. These results showed a very similar performance to the laboratory observations and are discussed in the detailed comparisons of Experiments 1(a) and 1(b).

3.2. Training a MLP for obstacle avoidance

For training a MLP using back-propagation, input and target data are required. The input data were vectors of seven combined IR/sonar sensor readings (reciprocals of the distance to the nearest object). For every sensory input vector a specific target vector is needed to tell the network the required translational and rotational velocity (scaled) to drive the motors. MLPs were trained in two modes.

In "wake" mode the Innate controller drove the robot around the laboratory for data collection. During the drive, the MLP watched the driver closely. As shown in Figure 2, it received the same sensory input as the Innate controller and attempted to imitate the Innate controller's outputs. The input/output pairs were recorded in a file. If the Nomad 200 collided during wake learning, the vector pairs leading to the collision were removed from the recorded training set. After a few thousand sensorimotor loops wandering in the environment, the robot was put into "sleep" mode where it remained at a standstill to accelerate the MLP training. In this mode the learning algorithm repeatedly cycled through the training data that had been collected in wake mode.

For the current evaluation of the performance of the trained obstacle avoidance, the chosen MLP had a 7-3-2 architecture. A small number of hidden units was used to prevent data over-fitting. It was trained in wake mode for 2K sensorimotor loops and then in sleep mode for 11K passes through the training data. With a learning rate = 0.8, a momentum factor = 0.1, and an error tolerance of 0.1, learning was terminated with a root mean square error, RMSE=0.065. The trained controller produced collision free behavior during several hours of running in the laboratory and elsewhere with the obstacles moved often. The visual impression of the two controllers in the laboratory was that the MLP controller produced smoother trajectories than the Innate controller and seemed to stay further away from obstacles in general. The relative performances of the two controllers were experimentally evaluated in Study 1.

Study 1: Comparing Innate and MLP controllers for avoidance

The relative performances of the Innate and MLP controllers were evaluated in two experiments in two worlds of increasing difficulty. The first simple simulator world, Lab1, shown in the upper left of Figure 3, is a rough, approximately scale, model of the laboratory pen. The second world, Sim1, shown in the lower left of Figure 3, is larger and more cluttered. For baseline measures, the same boundaries were used but were emptied of obstacles. In both experiments the robot was driven for a fixed interval of 1000 sensorimotor loops from each of 5 different locations (as shown by the Filled circles in Figure 3). This procedure was

repeated twice for each controller providing 10 trials each for statistical comparisons. The experimental hypothesis was that the MLP controller will extract the systematic structure of the behavior of the innate controller in its interaction with the world, as a result, the robot will improve on its innate performance.

Experiment 1(a)

The experiment was run in the world shown in Figure 3, Lab1. The data means and standard deviations for the mean raw sensor readings (SR) and the mean distances traveled are shown in Table 1. The Statistical analyses (Correlated t tests) revealed no significant difference between the mean raw sensor readings for the two controllers, $t = 1.19, df = 9, p > 0.26$, where df =degrees of freedom. This means that, on average, both controllers maintained similar distances from obstacles and walls. More importantly, the MLP controller drove the robot significantly further than the Innate controller, $t = 4.93, df = 9, p < 0.00082$. This finding cannot be attributed to the general operational speed of the two controllers since the Innate controller actually drives further in an empty world as can be seen from the Baseline Distance in Table 1. The results show that the MLP controller improves on the performance of the Innate controller in a relatively simple obstacle laden environment (Lab1 in Figure 3).

Table 1. The results of Experiment 1(a) (Lab1 world): the mean distance, in feet, traveled by the robot and the mean raw sensor readings (SR) in inches for the front 7 sonars/IRs for each controller over 10 runs (from 5 different positions) in the pen with a number of rectangular obstacles (Figure 3, Lab1). The baseline conditions were taken from a run with both controllers in the pen with all of the obstacles removed. Standard deviations (Std) are also shown.

	Innate	MLP
Mean SR	57.32	60.06
Std	6.51	2.47
Mean Distance	46.26	56.78
Std	4.48	5.99
Total Bumps	3	6
Baseline SR	155.90	156.79
Baseline Distance	114.04	110.06

Experiment 1(b)

The experiment was run in the world shown in Figure 3, Sim1. The data means and standard deviations are shown in Table 2. The statistical analysis of the mean sensor readings yielded no significant difference between the MLP and Innate controllers, t = 0.12, df = 9, $p > 0.9$. However, the robot traveled significantly further with the MLP controller than it did with the Innate controller, t = 7.69, df = 9, $p < 0.00004$. As in the previous experiment, this performance cannot be attributed to the general operational speed of the two controllers since the Innate controller actually drives further in an empty world as can be seen from the Baseline Distance in Table 2. The findings show that the MLP controller improves on the

performance of the Innate controller in an environment cluttered with obstacles (Figure 3, Sim1).

Table 2. The results from Experiment 1(b) (Sim1 world): the mean distance traveled by the robot, in feet, and the mean sensor readings (SR) in inches for the front 7 sonars and IRs for each controller over 20 runs with the simulator (Figure 3, Sim1). The baseline conditions were taken from a run with the simulator in an empty world with the same boundary walls as used in the other conditions. Standard deviations (Std) are also shown.

	Innate	MLP
Mean SR	61.39	61.66
Std	7.06	2.21
Mean Distance	63.96	83.81
Std	7.66	4.42
Total Bumps	5	3
Baseline SR	194.83	201.14
Baseline Distance	229.58	217.90

Summary of the results from Study 1

The findings from Study 1 show that both controllers performed well; the total numbers of bumps for the two experiments (40 runs) were 8 for the innate controller and 9 for the MLP. These were mainly from repeated bumping on single occasions. These data were too sparse to enter into the analyses. However, overall the MLP did better. This leads to the rejection of the null hypothesis in favor of the experimental hypothesis that the MLP avoidance controller can improve upon its teacher, the Innate avoidance controller. While both controllers maintained approximately the same mean distance from the walls and obstacles, the MLP controller drove the robot significantly further than the Innate controller in the alloted time. This difference was not attributable to the relative operational speed of the two controllers since, according to the baseline measures, in empty worlds the Innate controller drove the robot further. These findings lend support to the notion that the MLP controller is extracting an underlying systematic aspect of the behavior of the Innate controller from the environment-specific behavioral noise in which it is embedded.

4. Goal-finding

The aim of the goal-finding behavioral module was to drive the Nomad to any goal location, specified by xy coordinates, within the laboratory pen or elsewhere. The exclusive use of proprioception here, as noted in the introduction, is analogous to the behavior of animals (hymenopterans and mammals) when they are largely prevented from using normal external cues, e.g., night active rodents.

4.1. An Innate goal-finding controller

A "dead reckoning" method was employed to continuously calculate the distance and direction to the goal from the Nomad's current position. The Nomad's odometry system provides the current position, $\mathbf{r}$, of the robot and its angular direction, θ_r. This information was used to calculate the distance and heading to the goal. For driving the robot to the goal specified by a coordinate vector $\mathbf{g}$, the translational velocity, T, and the rotational velocity, R, were calculated by, $T = {d_{(\mathbf{r},\mathbf{g})}}^2(1- \mid 2\theta/\pi \mid)$, up to a maximum of $T = 100$ and $R = 450(\theta/\pi)$, where $d_{(\mathbf{r},\mathbf{g})}$ is the distance of the goal from the robot, and θ is the direction of the goal from the robot. The exception being when $\mid \theta \mid > \pi/2$ (the goal is behind), then $T = 5$.

In many tests in an empty robot pen in the laboratory the Nomad never failed to reach a goal using the Innate goal-find module. The main use here, in combination with the Innate avoid module, was to train the other MLPs to perform the task.

4.2. Training a MLP goal-finding controller

For training a MLP to find a goal, the input values were given as, $u_1 = \theta/\pi$, and $u_2 = d_{(r,g)}/5000$, and the target values were, $t_1 = (T + 200)/400$ and $t_2 = (R + 450)/900$. u_2, t_1 and t_2 were scaled between 0 and 1 and u_1 was scaled between -1 and +1. The neural net controllers were trained on 66 vector pairs generated such that they were equally spaced within the input space for the laboratory. The input vectors consisted of two values giving the scaled angle and distance to a goal. The target vectors consisted of the required (scaled) translational and rotational velocities generated by the Innate goal-finding module.

Network controllers trained off-line from these data performed well, always finding a path to the goal coordinates when all of the obstacles were removed from the laboratory space. The MLP selected for more extensive evaluation here had a 2-2-2 architecture. It was trained with backpropagation learning for 51K passes through the training data. With a learning rate = 0.8, a momentum factor = 0.1, error tolerance = 0.1, the MLP was trained to a RMSE = 0.075. It always located the goal in numerous tests in the laboratory pen and on the simulator. Laboratory tests were difficult and great care had to be employed to ensure that the robot did not crash. The goal-finding module is really only useful when combined with the avoidance behavior and this is how it is evaluated in the next section.

Study 2: Comparing two controllers for goal-finding and obstacle avoidance

A much simplified subsumption architecture (Brooks, 1986) was designed to combine avoid and find-goal modules into a single controller. The two innate modules formed an Innate controller and the two MLP modules formed a Two-MLP controller. Goal-finding was the top level behavior in the innate implementation and it was in control by default. Control switched to the avoid module when the minimum raw sensor reading was less than 15 inches. It reverted to goal-finding again when the minimum raw sensor reading was greater than 15 inches.

These controllers have been successfully tried and tested in the laboratory pen and in external demonstrations and exhibitions of the robot with different numbers of obstacles

in many different configurations. See Figure 6 for a trace of the trajectories of the Innate and One-MLP controllers in the Sim1 world (Figure 3) driving to 4 goals and returning home. The controllers were not perfect. There are a number of circumstances in which the robot can get trapped in dead ends or can bump when the density of obstacles is high. The two controllers were compared here in 3 different simulator worlds of increasing difficulty shown in Figure 3: the Lab2 world was used for Experiment 2(a), Experiment 2(b) used the Sim1 world, and Experiment 2(c) used the Sim2 world. The experimental hypothesis was that the Two-MLP controller would improve on the performance of its Innate 'teaching' controller.

Experiment 2(a)

Experiment 2(a) was conducted in a simulator world that was a rough, approximately scale, model of the laboratory pen in which the controllers were trained as shown in Figure 3, Lab2. The experimental procedure was to use the Two-MLP and Innate controllers to drive the robot around the pen through different goal locations. The Nomad was placed in turn in each of the 5 chosen locations and the task was to visit the other 4 locations in a prespecified order (allocated randomly in advance). The locations are shown as filled circles in Figure 3, Lab 2. This procedure was repeated twice for each controller making 20 experimental trials altogether.

Table 3. The results from Experiments 2(a) and 3(a) (Lab2 world): the mean sensor readings (SR) in inches, the mean number of sensorimotor loop values (SLV), and the mean distance traveled in feet, and the smoothness as recorded for the robot for each controller to find goal locations over 10 runs (from 5 different positions) in the pen. The mean smoothness is also shown. The baseline conditions were taken from a run in the pen with all of the obstacles removed. Standard deviations (Std) are also shown.

	Innate	Two-MLPs	One-MLP
Mean SR	50.48	50.75	54.47
Std	0.62	3.91	0.75
Mean SLV	933.70	857.4	749.60
Std	167.49	165.02	89.99
Mean Distance	56.85	58.55	55.04
Std	12.43	12.24	9.03
Mean Smoothness	0.061	0.068	0.074
Std	0.0098	0.0075	0.0085
Total Bumps	6	2	2
Baseline SR	110.97	114.13	113.69
Baseline SLV	673	707	701
Baseline Distance	43.79	46.10	44.65
Baseline Smooth.	0.065	0.065	0.064

The robot successfully reached all of the goals on every run with both controllers. The data means and standard deviations for Experiment 2(a) are shown in the first two columns of Table 3. The third column concerns the results from Experiment 3(a) and will be discussed later. Analysis of the mean sensor readings data yielded no significant difference between the Innate controller and the Two-MLP controller $t = 0.19, df = 9, p > 0.85$. Neither was

there a significant difference observed between the mean distances traveled by the robot using the two controllers, $t = 1.30, df = 9, p > 0.22$. However, the Two-MLP controller was significantly faster than the Innate controller in terms of the number of sensorimotor loops required to complete the task, $t = 4.12, df = 9, p < 0.0026$ and it was significantly smoother, $t = 4.51, df = 9, p < 0.0015$. Comparisons with the baseline measures in Table 3 showed that in an empty world the Innate controller traveled faster than the Two-MLP controller and was equally smooth. Thus, the results support the hypothesis that the Two-MLP controller has learned to improve upon the performance of the Innate controller in an obstacle laden environment.

Experiment 2(b)

In this experiment the robot traveled from its home to four known goal locations, indicated by filled circles in Figure 3, Sim1, and then returned home again. The home position is indicated by an H in Figure 3, Sim1. Each controller was run for 20 trials from the same location. For the purposes of illustration, one trajectory for one run from each of the controllers is shown in Figure 6. Careful study of the figure shows clear differences between the Innate and Two-MLP controllers. However, it is the statistical analyses of the experimental quantities over several trials that show up the main differences.

Table 4. The results from Experiments 2(b) and 3(b) (Sim1 world): the mean sensor readings (SR) in inches, the mean number of sensorimotor loop values (SLV), the mean distance in feet traveled by the robot for each controller to find goal locations over 20 runs in a simulated pen crowded with obstacles. The mean smoothness is also shown. The baseline conditions were taken from a run in the simulated world with all of the obstacles removed. Standard deviations (Std) are also shown.

	Innate	Two-MLPs	One-MLP
Mean SR	52.94	50.75	55.25
Std	0.18	0.25	0.18
Mean SLV	1892.70	1607.00	1344.60
Std	16.13	5.87	2.28
Mean Distance	142.80	144.54	145.24
Std	0.08	0.19	0.17
Mean Smoothness	0.090	0.108	0.108
Std	0.00034	0.00015	0.00015
Total Bumps	6	0	0
Baseline SR	139.83	146.09	146.42
Baseline SLV	678	752	554
Baseline Distance	87.89	88.28	89.38
Baseline Smooth.	0.13	0.12	0.16

The robot successfully reached all of the goals on every run with both controllers. The data means and standard deviations for Experiment 2(b) are shown in the first two columns of Table 4. The third column concerns the results from Experiment 3(b) and will be discussed later. Statistical analyses were not required for any of the comparisons since the populations of scores do not overlap. The Innate controller always traveled a shorter distance to complete the task than the Two-MLP controller. However, the Two-MLP controller completed the

task faster than the Innate controller (in sensorimotor loops), and produced a smoother performance. Comparisons with the baseline measures in Table 4 showed that in an empty world the Innate controller traveled faster than the Two-MLP controller and was slightly smoother. These results add support to the hypothesis that the Two-MLP controller learns to outperform the Innate controller in an obstacle laden environment.

Experiment 2(c)

The experiment was run in the world shown in Figure 3, Sim2. As in the previous experiment, in all of the experimental trials, the two controllers drove the robot to 4 goal locations and returned home. This is similar to the world used in the previous experiment; it has the same boundary walls but some more difficult turns are required by the robot. This provides quite a tough test for the controllers. After running the Innate controller for 20 trials, only 12 were usable because the bumper stuck on 8 of the trials which terminated the experiment. It was then decided to use only 12 trials. It took 18 runs to get 12 trials for the Two-MLP controller since it had its bumper stuck on 6 of the trials.

The robot successfully reached all of the goals on each of the 12 experimental runs for each of the controllers. The mean data and standard deviations are shown in the first two columns of Table 5. The third column concerns the results from Experiment 3(c) and will be discussed later. Although the difference between the means for the distances traveled by the Innate (137.78 feet) and the Two-MLP (137.49 feet) controllers was very small (0.29 feet) the differences between the measures were stable, $t = 3.62, df = 22, p < 0.0016$. The Two-MLP controller performed the task faster than the Innate controller with no overlap between the distributions of scores. It also kept further away from walls and obstacles than the Innate controller (mean SR), $t = 2.44, 22, p < 0.024$ and was overall smoother $t = 2.69, df = 18, p < 0.016$. The same baseline measures measure were used as for the previous experiment, since they both used the same empty world. The comparisons with the baselines in Table 4 showed that in an empty world the Innate controller traveled faster than the Two-MLP controller and was slightly smoother. This is another confirmation instance of the hypothesis that the MLP learns to improve on the performance of the Innate controller.

Summary of the results from Study 2

The overall pattern of results from the three experiments reported in Study 2 confirmed the experimental hypothesis that the generalization performance of the subsumption architecture, consisting of the Two-MLP controller trained using backpropagation, would improve upon the performance of the Innate 'teaching' controller. This was demonstrated in three worlds of increasing difficulty. The results for the distances traveled and the mean sensor reading were inconclusive. In the easiest world (Experiment 2(a)) the controllers maintained an approximately equal average distance from obstacles and walls. But in the two more difficult environments, the Two-MLP controller maintained a greater average distance in one experiment and less in the other. There was no significant difference for the distance traveled in Experiment 2(a), but, although the differences were small in Experiment 2(b), the Two-MLP traveled significantly further, and the Innate controller traveled significantly

Table 5. The results from Experiments 2(c) and 3(c) (Sim2 world): sensor readings (SR) in inches, the mean number of sensorimotor loop values (SLV), the mean distance traveled by the robot, in feet, for each controller to find goal locations over 12 runs in a simulated pen crowded with obstacles and the smoothness of travel. The baseline measures were taken from a run in the simulated world with only the boundaries and all of the obstacles removed. Standard deviations (Std) are also shown. The Baseline measures are the same as those shown in the previous Table.

	Innate	Two-MLPs	One-MLP
Mean SR	49.68	49.50	51.50
Std	0.21	0.17	1.51
Mean SLV	3208.42	2834.00	2409.25
Std	6.37	44.45	189.93
Mean Distance	137.78	137.49	158.38
Std	0.15	0.22	11.77
Mean Smoothness	0.045	0.052	0.065
Std	0.0028	0.0079	0.00052
Total Bumps	8	6	0

further in Experiment 2(c). However, it was clear from the results of the three experiments that the robot completed the task faster and smoother when it was driven by the Two-MLP controller than when it was driven by the Innate controller. Thus, it appears that the plasticity of MLP learning does adapt the general behavior of the Innate controller to the world in which it is active. The findings indicate that there is an underlying systematic aspect to the way in which the two innate control modules (goal-finding and avoid) interact with the world.

Study 3: Comparisons with a single MLP controller

In the previous two studies it was observed that MLP controllers appear to learn some of the regularities underlying the behavior of hardwired innate controllers in such a way as to improve upon their performance. In this study, the notion of *incremental bootstrapping* was tested. That is, the previously trained Two-MLP controller was used to bootstrap a new single MLP, the One-MLP controller. The reasoning was that the single MLP controller would extract the systematic aspect of both the robot's interaction with the word and the interaction of the two behavioral modules in the Two-MLP subsumption controller used to train it. In this way a new, perhaps more appropriate, function approximation can be trained with the possibility of filtering out some more of the environment specific behavioral noise.

A single MLP controller was trained on the complete task of finding goals and avoiding obstacles. The method of training is illustrated in Figure 5. The training data were collected in wake mode by driving the Nomad 200 around the laboratory pen under the control of the Two-MLP controller described in Study 2. Input/target vector pairs were collected for training the new network with 9 inputs (2 goal vector inputs and 7 distance sensor inputs), three hidden units, and two motor output units. The training data were gathered over a period of time with continued learning in a changing environment. The exact number of passes through the training data is not meaningful since training sets varied in size and some

of the learning was in wake mode. The length of the goal paths were varied and both wake and sleep training were used. The net to be evaluated here was trained to RMSE=0.04.

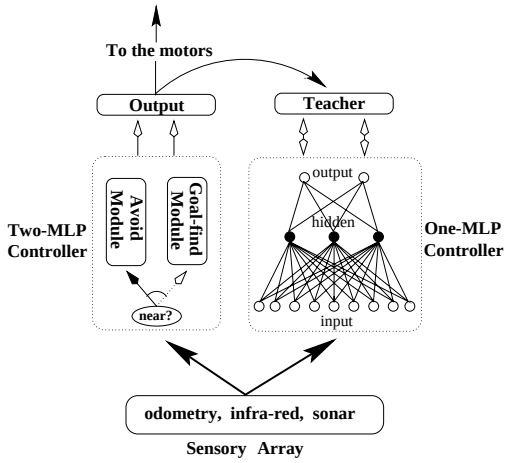

Figure 5. An illustration of the learning method for training a single neural network controller from a subsumption architecture with two neural net controllers. The subsumption controller drives the robot and outputs the teaching signal for the network.

To evaluate the performance of the One-MLP controller, it was tested in the same three worlds for the same number of experimental runs as the controllers in the previous study and with the same experimental conditions. Statistical analyses were then employed to compare the performance of the One-MLP controller with those of the Innate and Two-MLP controllers described in Study 2. The experimental hypothesis was that the One-MLP controller would improve upon the performance of the Two-MLP and Innate controllers. An example of trajectories for the three controllers is shown for the Sim1 world in Figure 6 for Experiment 3(b). Careful study is required to see the differences between the trajectories. Statistical analyses of the experimental quantities make these differences clear.

Experiment 3(a)

The experiment was run in the world shown in Figure 3, Lab2. As in Experiment 2(a), the Nomad was placed in turn in each of 5 chosen locations and the task was to visit the other 4 locations in a prespecified order (allocated randomly in advance). The locations are shown as filled circles in Figure 3, Lab 2. This procedure was repeated twice for each controller making 20 experimental trials altogether. The robot successfully reached all of the goals on every run.

The data means and standard deviations for the One-MLP controller in comparison with the other controllers are shown in Table 3. Statistical analyses revealed no significant differences between the distance traveled by the One-MLP controller compared to the Two-MLP controller, although the result marginally favored the one-MLP controller, $t = 2.11, df = 9, p > 0.06$, and the Innate controller, $t = 0.87, df = 9, p > 0.4$. Further

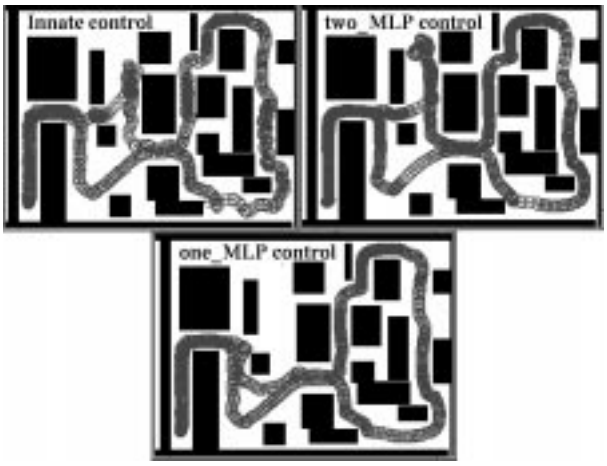

Figure 6. An illustration of robot trajectories for three controllers running in Sim1 world (see Figure 3). Careful study of the figures shows that the trajectory of the Innate controller is more erratic than either of the other trajectories. The One-MLP controller produces a smoother and shorter trajectory than the Two-MLP controller, but this is more difficult to see. The statistical analyses presented in studies 2(b) and 3(b) make the differences clearer

comparisons showed that the One-MLP controller carried out the task faster (less sensorimotor loops) than either the Innate controller, $t = 5.49, df = 9, p < 0.0004$, or the Two-MLP controller, $t = 3.45, df = 9, p < 0.007$; the One-MLP controller stayed further from the obstacles and walls than either the Innate controller, $t = 10.31, df = 9, p < 2.79 \times 10^{-6}$, or the Two-MLP controller, $t = 3.29, df = 9, p < 0.0094$; and the One-MLP controller was smoother than both the Two-MLP controller, $t = 2.65, df = 9, p < 0.027$ and the Innate controller, $t = 8.20, df = 9, p < 0.00002$. The comparisons with the baselines in Table 3 showed that in an empty world the One-MLP controller drove slower than the Innate controller but slightly faster than the Two-MLP controller. Overall, the results in Table 3 show that the One-MLP controller adapts well to its environment by improving on the performances of the other two controllers.

Experiment 3(b)

The experiment was run in the world shown in Figure 3, Sim1. As in Experiment 2(b), the controller was run for 20 trials from the same location (marked with an H). The robot successfully reached all four goals and returned home on every run. The data means and standard deviations for the One-MLP controller in comparison with the other controllers are shown in Table 4. Statistical analyses were not required for comparing the sensorimotor loop values or the mean sensor readings of the One-MLP controller with the Innate and Two-MLP controllers since the distribution of the scores did not overlap. The One-MLP traveled faster than the other controllers and stayed further away from walls and obstacles. The One-MLP controller traveled further than both the Innate Controller (no overlap between the distributions of their scores) and the Two-MLP controller, $t = 12.25, df = 38, p <$

9.33×10^{-15}. This time the One-MLP controller was not significantly smoother than the Two-MLP controller,$t = 0.70, df = 18, p > 0.4$. However, as before One-MLP controller was smoother than the Innate controller with no overlap in the distributions of the scores. The comparisons with the baselines in Table 4 showed that in an empty world the One-MLP controller drove faster than both the Innate controller and the Two-MLP controller. Overall, the results shown in Table 4, show that the One-MLP controller improved upon the performance to the other two controllers.

Experiment 3(c)

The experiment was conducted in the world shown in Figure 3, Sim2. As in Experiment 2(c), the robot was required to drive to 4 locations from its home (H) and then return. The robot successfully reached all of the goals and returned home on all 12 runs.

The data means and standard deviations for the One-MLP controller in comparison with the other controllers are shown in Table 5. Statistical analyses yielded a similar pattern of results to the previous two experiments. The mean overall distance traveled by the One-MLP controller was significantly further than for both the Two-MLP controller, $t = 6.14, df = 22, p < 3.49 \times 10^{-6}$, and the Innate controller, $t = 6.06, df = 22, p < 4.25 \times 10^{-6}$. Analysis of sensorimotor loop values revealed that the One-MLP controller completed the task faster than both the Two-MLP controller, $t = 7.54, df = 22, p < 1.55 \times 10^{-7}$, and the Innate controller $t = 14.57, df = 22, p < 8.84 \times 10^{-}13$. Note, however, the variability of the SLV for the One-MLP condition (5). This was because it sometimes found a shortcut. For the mean sensor readings, the One-MLP controller stayed further away from obstacles and walls than both the Innate controller, $t = 4.12, df = 22, p < 0.00046$ and the Two-MLP controller, $t = 4.56, df = 22, p < 0.00016$. As before, the One-MLP controller was significantly smoother than the Two-MLP controller, $t = 5.24, df = 18, p < 0.00006$, and the Innate controller, $t = 22.20, df = 18, p < 1.58 \times 10^{-14}$.

Summary of the results from Study 3

The findings reported in Study 3 clearly support the experimental hypothesis that the One-MLP controller would outperform both its teacher and its teacher's teacher. In all three experiments in worlds of increasing difficulty the One-MLP controller drove the robot to complete the tasks fastest, steered it furthest from the obstacles and was smoother. Although the extra distance from the obstacles enabled the One-MLP to move faster, it was at a cost. In all but the first experiment, when the robot was controlled by the One-MLP the distance it traveled was greater. The overal findings provide strong support for the notion that the single MLP controller can extract the systematic aspect of both the robot's interaction with the word and the interaction of its two behavioral modules in the MLP subsumption controller used to train it.

The bump data were too sparse for statistical analysis. In the 6 experiments reported in Studies 2 and 3, the Innate controller bumped 20 times, the Two-MLP controller bumped 8 times, and the One-MLP controller, twice. Not much can be made of these low figures except that they indicate that all of the controllers performed reasonably well. It should

be noted that the Innate and Two-MLP controllers had some difficulties in the Sim2 world shown in Figure 3. Their driving resulted in stuck bumpers in a number of cases.

5. Summary and Discussion

The primary aim of the studies was to investigate the incremental adaption of robot controllers given basic innate behaviors. The inspiration from biology was that predispositions may generate behaviors that can then be adapted to more specific environmental circumstances. Such behavioral flexibility would be particularly useful if the type of environment that the creature must live in, e.g., swamp, forest, desert, or office building, and the physiological embodiment (length of legs, etc.) are unknown *a priori*. But the question is, how can such general behavior be adapted? One suggestion (Johnson, 1992) was that the innate behavior bootstraps learning in plastic neural networks.

The objective of the current research was to evaluate a method by which such learning could take place. The idea was that the behavior of a simple hardwired reactive (innate) controller on a mobile robot could exhibit an underlying systematic aspect in its interaction with the world that could be learned and employed by MLP controllers. This would be manifest in improved performance of the MLP controllers. Furthermore, the possibility of incremental bootstrapping was investigated. A previously trained Two-MLP controller was used to bootstrap a new single MLP, the One-MLP controller. The reasoning was that the single MLP controller would extract the systematic aspect of both the robot's interaction with the world and the interaction of the two behavioral modules in the Two-MLP subsumption controller used to train it.

A secondary aim of the studies was to begin the development of a method for comparing controllers experimentally. This led to the use of four measurable quantities along four pertinent dimensions for the tasks (bumps were too infrequent to use): the mean distance from obstacles, the number of sensorimotor loops required for task completion, the distance traveled, and the smoothness of travel. Using these quantities, three series of experiments in the four different worlds shown in Figure 3 were carried out to test the general hypothesis that MLP controllers could outperform their 'teaching' controllers. In all of the studies the controllers performed well with only a total of 47 bumps recorded over 166 experimental runs. Most of these were repeated bumps on a single obstacle on a single run. In Study 1, comparisons were made between a MLP obstacle avoidance control module and an innate avoidance control module used to train it. The results from the two experiments suggested that the MLP improved upon the performance of the Innate controller by traveling further in a fixed time period in an obstacle laden environment. These results were compared to baseline measures that showed the opposite effect for controllers in an empty world.

Studies 2 and 3 were both concerned with goal-finding in an obstacle laden environment and they will be treated together. In Study 2, a simple subsumption architecture was used to combine the goal and avoidance modules into a single control structure. The reported comparisons were between a controller consisting of two MLP modules and its teacher, a controller consisting of the two innate control modules. In Study 3, the performance of the two subsumption controllers, innate and MLP, were compared with a single MLP controller that was trained on the combined task of avoiding obstacles whilst finding goals. The specific experimental hypothesis was that the single MLP controller would extract the

systematic aspect of both the robot's interaction with the word and the interaction of the two behavioral modules in the MLP subsumption controller used to train it.

To get an overall picture of the results of the comparison for Studies 2 and 3, each of the goal-find and avoid controllers was scored according to how well it fared in pairwise comparisons between the controllers along the measured dimensions. When a comparison showed a controller to be superior to one other controller along one of the measured dimensions, it scored 1 and when two controllers were not significantly different on a dimension, they scored 0.5 each, otherwise they scored 0. The maximum score that a controller could score on any dimension was 6 (beating both of the other controllers in all three experimental setups). The results, shown in Table 6, clearly favor the One-MLP controller with 18.5 out of 24. It had maximum scores of 6 on both the distance from obstacles and time to completion dimensions, and an almost maximum score, 5.5, for smoothness. Its only weakness was that it tended to travel further than the other two controllers in completing its tasks (it only scored 1 on Distance traveled). The extra distance traveled resulted, in part, from keeping a greater distance from obstacles. This was a small price to pay for smoother and faster performance. The One-MLP controller was less influenced by its local environment. The runner-up was the Two-MLP controller with 12 out of 24. It outperformed the Innate controller on three dimensions and equalized on distance traveled. The Innate controller came last with a score of 5.5 out of 24.

Table 6. The scores for the three goal-finding and avoid controllers based on the comparisons from all of experiments in Studies 2 and 3. See the text for an explanation.

	Innate	Two-MLPs	One-MLP
Distance from obstacles	1.5	1.5	6
Time to completion	0	3	6
Distance traveled	4	4	1
Smoothness	0	3.5	5.5
Totals	5.5	12	18.5

One of the most significant findings here was that a single MLP was shown to extract higher order regularities in the behavior of modules that were interacting with each other as well as with the world. This could lead to what has been termed "emergent behavior", but it would also be "emergent cognition" (Sharkey & Sharkey, 1994). For example, in the Two-MLP controller described in Study 2, the avoid module inhibited the goal-finding module when obstacles were too close and thus the direction of the goal location did not influence the direction of the obstacle avoidance. However, the One-MLP controller had no knowledge of which of the two modules was in operation at any time. They were in a black box. During training, the One-MLP observed only the sensory input to the Two-MLP controller and its output to the motors. It makes its generalizations on the basis of the combined behavior of the two modules and integrates them into a single controller.

In the environments used for training the MLPs, the goal behavior appears to dominate and so the integrated behavior is more goal directed and less influenced by the force of obstacles. Recall that the One-MLP controller had the maximum score of 6 for the number of winning comparisons for the distance that it maintained from obstacles and 5.5 for the

smoothness of travel. The behavior of the One-MLP controller showed a minor move up the developmental chain from its origins to begin the escape from purely reactive control. The robot is now in a position to detect and exploit more general trends in its behavior (Sharkey & Heemskerk, 1997). Of course this would be better if there were more than two modules to learn from.

Recently, Garry Trotter, a project student with our group conducted some studies in which a "dead-end escape" module was added to the subsumption architecture described above. This module was triggered if the goal was behind the robot within a certain angle; then the robot moved to the nearest starboard wall and continued wall following until it had turned 180° in one direction and there was free space. The controllers have not been submitted to a thorough experimental evaluation yet, but early indications are that the results are following a similar pattern to those reported here with a three module innate controller, a three module MLP controller and a single MLP controller. The single MLP integrated systematic aspects of all three behaviors collectively.

These findings show promise for the general methods proposed here. It has been demonstrated that innate behaviors can bootstrap learning in adaptive neural networks. Moreover, the performance of the controllers improved incrementally. It would also be interesting to enlist other learning methods such as GAs to develop the innate controllers and reinforcement learning to train more mature controllers. Additionally, control using the simple infra-red and sonar sensors could be useful for bootstrapping learning with other sensors such as lasers and cameras. Many of the ideas for developing the controllers described here were based on inspirations from studies of animal navigation. No attempt was made to model data from particular species. From this perspective, the robot controllers are too under-constrained to relate to biological models. However, the establishment of general principles of learning directed locomotion may feedback into animal research. Some ideas were presented here about how a neural system could adapt to a particular environment by extracting regularities from the amalgamated behavior of inflexible innate subsystems interacting with the world.

Acknowledgments

I would like to thank Tom Ziemke and Amanda Sharkey for commenting on several drafts of this paper, John Neary and Jan Heemskerk for helping to set up the preliminary research (Sharkey et al., 1996 (a), Sharkey et al., 1996 (b)), and Ali Zalzala and Gordon Manson for their help and collaboration in the project. I would also like to thank the Engineering and Physical Sciences Research Council (UK) for funding this research (project number GR/K18986).

Notes

1. Student was William Gosset's pseudonym because his employer, the Guinness Brewing Company in Dublin, would not allow him to publish under his real name.

References

Anderson, T.L. & Donath, M. (1990). Animal behavior as a paradigm for developing robot autonomy. *Robotics and Autonomous Systems, 6,* 145–168.

Bekey, G.A. & Goldberg K.Y. (Eds.). (1993). *Neural Networks in Robotics.* Boston, MA: Kluwer.

Benhamou, S., Sauve, J.P. & Bovet, P. (1990). Spatial memory in large scale movements: Efficiency and limitations of the egocentric coding process. *Journal of Theoretical Biology, 145*, 1–12.

Brooks, R.A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation, RA-2*, 14–23.

Carthy, J.D. (1963). *Animal Navigation*. London, U.K.: Unwin Books, 3rd edition.

Collett, T.S. (1996). Insect navigation *en route* to the goal: multiple strategies for the use of landmarks. *Journal of Experimental Biology, 199*, 227–235.

Donnart, J.I. & Meyer, J.A. (1996). Learning reactive and planning rules in a motivationally autonomous animat. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics. Special issue on Learning autonomous robots, 26(3)*, 381–395.

Dorigo, M. (1996). (Ed.), Special issue on learning autonomous robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 26(3).

Esch, H.E. & Burns, J.E. (1996). Distance estimation by foraging honeybees. *Journal of Experimental Biology, 199*, 155–162.

Etienne, A.S., Maurer, R. & Seguinot, V. (1996). Path integration in mammals and its interaction with visual landmarks. *Journal of Experimental Biology, 199*, 201–209

Floreano, D. & Mondada, F. (1996). Evolution of homing navigation in a real mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics. Special issue on Learning autonomous robots, 26(3)*, 396–407.

Gallistel, C.R. (1990). *The Organization of Learning.* Cambridge, MA: MIT Press.

Gould, J.L. (1986). The locale map of honey bees: do insects have cognitive maps? *Science, 232*, 861–863.

Johnson, M.H. (1992). Imprinting and the development of face recognition: From chick to man. *Current Directions in Psychological Science, 1*, 52–55.

Johnson, M.H. & Bolhuis, J.J. (1991). Imprinting, predispositions and filial preference in the chick. In R.J. Andrew, (Ed.), *Neural and Behavioural Plasticity*. Oxford, UK: Oxford University Press.

Kassim, A.A. & Kumar, B.V.K.V. (1995). Potential fields and neural networks. In M.A. Arbib, (Ed.), *The handbook of brain theory and neural networks*. Cambridge, MA: MIT Press.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research, 5*, 90–98.

Krose, B. (1995). (Ed.), Special issue on reinforcement learning and robotics. *Robotics and Autonomous Systems, 15*.

McCulloch, W.A. & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics, 5*, 115–133.

Meeden, L.A. (1996). An incremental approach to developing intelligent neural network controllers for robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 26(3)*, 474–485.

del Millan, J. (1996). Rapid, safe, and incremental learning of navigation strategies. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 26(3)*, 408-420.

Nolfi, S. (1997). Evolving non-trivial behaviors on a real robot: a garbage collecting robot. *Robotics and Autonomous Systems, 22*.

Nolfi, S. & Parisi, D. (1997). Learning to adapt to changing environments in evolving neural networks. *Adaptive Behavior, 5*, 75–98.

Owen, C. & Nehmzow, U. (1997). Middle scale robot navigation - a case study. In N.E. Sharkey and U. Nehmzow, (Eds.), *Proceedings of the AISB workshop on Spatial Reasoning in Mobile Robots and Animals*, (pp. 104–111), Manchester University, Dept. Computer Science Technical report UMCS-97-4-1.

Salomon, R. (1997). The evolution of different neuronal control structures fo autonomous agents. *Robotics and Autonomous Systems, 22*.

Sharkey, N.E. (1997). The new wave in robot learning. *Robotics and Autonomous Systems, 22*.

Sharkey, N.E. (1997 (b)). Artificial neural networks for coodination and control: The portability of experiential representations. *Robotics and Autonomous Systems, 22*.

Sharkey, N.E., Heemskerk, J.N.H. & Neary, J. (1996 (a)). Subsuming behaviors in neural network controllers. In H. Hexmoor and L. Meeden, (Eds.), *Proceedings of Robolearn-96: An international workshop on learning for autonomous robots*, (pp 98–104), Key West, Florida.

Sharkey, N.E., Heemskerk, J.N.H. & Neary, J. (1996 (b)). Training artificial neural networks for robot control. In A.B. Bulsari, S. Kallio, and D. Tsaptsinos, (Eds.), *Solving engineering problems with neural networks.* London: Systems Engineering Association.

Sharkey, N.E. & Heemskerk, J.N.H. (1997). The neural mind and the robot. In A.J. Browne, (Ed.), *Neural Network Perspective on Cognition and Adaptive Robotics*. London: Institute of Physics Press.

Sharkey, N.E. & Sharkey, A.J.C. (1994). Emergent cognition. *Handbook of Neuropsychology, 9*, 347–360.

Srinivasan, M.C., Zhang, S.W., Lehrer, M. & Collett, T.S. (1996). Honeybee navigation *en route* to the goal: visual flight control and odometry. *Journal of Experimental Biology, 199*, 127–244.

Touzet, C. (1997). Neural reinforcement learning for behavior synthesis. *Robotics and Autonomous Systems, 22*.

von Frisch, K. (1967). Honeybees: Do they use direction and distance information provided by their dancers? *Science, 158*, 1076–1077.

Walter, G.W. (1950). An imitation of life. *Scientific American, 182*, 42–54.

Walter, G.W. (1953). *The living brain*. New York: Norton.

Wehner, R. (1992). Arthropods. In F. Papi (Ed.), *Animal Homing*. London, U.K.: Chapman and Hall.

Wehner, R., Michel, B. & Antonsen, P. (1996). Visual navigation in insects: coupling of egocentric and geocentric information. *Journal of Experimental Biology, 199*, 129–140.

Received September 1, 1997
Accepted December 30, 1997
Final Manuscript February 1, 1998