Rapid Unsupervised Connectionist Learning for Backing a Robot with Two Trailers*

Dean F. Hougen, Maria Gini, and James Slagle
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

Abstract

This paper presents an application of a connectionist control-learning system designed for use on an autonomous mini-robot. This system was formerly shown to form useful two-dimensional mappings rapidly when applied to backing a car with a single trailer. In the current paper the learning system is extended to three dimensions and applied to a similar but significantly more difficult problem. The system is shown to be capable of rapid unsupervised learning of output responses in temporal domains through the use of eligibility traces and inter-neural cooperation within topologically defined neighborhoods.

1 Introduction

Connectionist control-learning systems have recently received much attention; numerous papers and several books have been published on this topic in the last few years (e.g. [13, 17]). An overview of many such systems as they have been applied to robot control is given by Prabhu and Garg [16]. Most of these works, however, have concentrated on simulated systems and therefore have not had to deal with the ambiguities and constraints of the real world. The primary exception to this has been in the area of navigation (e.g. [1, 2, 3]). In 1996, Hougen et al [5] presented a new connectionist system designed for task learning on a real robot. In the current paper, we present an extension of this learning system to a significantly more difficult task.

Learning responses is generally classified into supervised and unsupervised learning. In supervised learning an agent or function, often called the teacher, provides the desired output response for each input vector. Systems that do not make use of a teacher, then, are known as unsupervised learning systems.

*This work was funded in part by the NSF under grant NSF/DUE-9351513 and by the AT&T Foundation.

Within unsupervised learning different levels of feedback may be available. Often an evaluation of system output is immediately available. This allows learning to occur for each input vector and output response pair. We are concerned with learning in situations in which a less immediate response is available.

In this paper we examine the problem of backing a rig consisting of a car and two trailers to a target location by steering only the front wheels of the car. This task is significantly more difficult than the problem examined previously (in [5]), in which the rig consisted of a car with a single trailer. No feedback is available until the task is completed (called *terminal feedback*) and the feedback is simply a boolean value (a success or failure signal).

Control-learning in autonomous robotic systems provides many challenges. The learning system must be robust enough to overcome the problems of noisy input data and uncertain interactions between motor commands and effects in the world, be compact enough to fit into available on-board memory, and be able to give responses in real time. The system we present here meets these specifications, yet is capable of unsupervised learning of a difficult credit-assignment problem.

2 ROLNNET

For rapid output learning in terminal feedback problems with boolean evaluation functions, Hougen et al [5] introduced the Rapid Output Learning Neural Network with Eligibility Traces (ROLNNET). In order to form mappings from input parameters to output responses, the input space is partitioned by the system designer into discrete regions and a single artificial neural element is assigned to each region. Neurons are provided with temporal sensitivity through the use of eligibility traces and adjust their response values on task completion. Response learning can take place rapidly through the use of inter-neural cooperation. This cooperation is based on the neighborhood concept inspired by Kohonen's Self-Organizing Topological Feature Maps [8]. ROLNNET systems can be seen as a simplification of the more general learning system introduced by Hougen [4], in which the system also learns the input space partitioning.

2.1 Neighborhood function

The internal structure of a ROLNNET map is defined by a topological ordering of the neurons that remains unaltered as the network learns. We use an $8 \times 8 \times 8$ cubic topology. Each neuron is uniquely numbered with a trio of integers that can be thought of as its coordinates in topology space. The existence of a network topology allows for the definition of a *distance* function for the neurons. Typically, this is defined as the Cartesian distance between coordinates in topology space. For the sake of computational efficiency, however, we use the maximum difference between coordinates of the neurons. E.g. for neurons (1,1,5) and (2,3,4) the distance between them would be $max(|1-2|, |1-3|, |5-4|)$ or 2.

The distance function is used indirectly through the definition of *neighborhoods*. A neighborhood may have any width from zero (the neighborhood is restricted to the unit itself) to the maximum separation between units in topology space (the entire network is in the neighborhood) and may vary with time, typically starting large and subsequently decreasing.

Formally, if U is the set of units u in the network, d the distance function defined on topology space, and W a time dependent function specifying the width of the neighborhood, then the neighborhood N of neuron n at time t may be defined as

$$N_n(t) = \{u \in U \mid d(n,u) \leq W(t)\} \tag{1}$$

All units within a neighborhood may be treated as belonging to a single class for a particular computation, and those outside as belonging to a separate class, giving a discretization which improves the computational efficiency of the method.

The concept of the neighborhood relationship is borrowed from Kohonen's Self-Organizing Topological Feature Maps [8] where the neighborhoods are used for self-organization of the maps.

2.2 Competition

Each neural element in the network is sensitive to a particular region in the input space of the problem. For the present application, in which we have an $8 \times 8 \times 8$ cubic topology, the three input dimensions are evenly partitioned into eight regions along each axis. A one-to-one correspondence between neurons and input space regions is defined such that nearest neighbors (according to the neighborhood relation described above) are assigned adjacent regions of the input space.

Each time a new input vector is given to the network the neuron sensitive to the input region into which the vector falls is declared the "winner". The winning neuron gives an output response based on its response weight value (see 2.4) and has its eligibility for adaptation increased (see 2.3).

2.3 The eligibility trace

One function of biological neurons which has not been approximated in the more standard connectionist systems is what we refer to as the eligibility trace. It is known that many neurons become more amenable to change when they fire (see, e.g. [7]). This plasticity reduces with time, but provides an opportunity for learning based on feedback received by the neuron after its activity.

All neural elements in a ROLNNET system have an eligibility value associated with them. Initially, all neurons are given an eligibility value of zero. Each time a neural element fires (wins and gives an output response), its eligibility is increased by a preset amount which is uniform for all neurons in the network. The eligibility value for each neuron decays exponentially regardless of whether or not it fires on any given time step.

2.4 Output weights

Each neural element has a single output weight which is initially given a random value. Together with the input region sensitivities described above (2.2), the weights can be understood as describing a mapping from car-trailer states to output responses.

The output weights are used to determine the system's response to an input vector. For the present application, the weight value of the winning neuron (see 2.2) is examined for its sign alone. If the weight is positive, the wheels of the car are turned to the right. Otherwise, the wheels are turned to the left.

When success or failure is signaled, the weights are updated according to

$$w^{new} = \text{sign}(w^{old})(|w^{old}| + e\, s(T)\, f) \tag{2}$$

where w is the weight, e is the eligibility for adaptation, s is a scaling function that changes with the trial number T, and f is the feedback signal (+1 for success, -1 for failure). The scaling function s is used to allow for large changes to the weights in early training trials and smaller changes in subsequent trials. In this application, s is defined to be

$$s(T) = \frac{1}{1 + (T-1) \bmod 100} \tag{3}$$

where T is the trial number. (I.e. For the first 100 trials, s is 1, for the second 100 it is $1/2$, for the third 100 it is $1/3$, etc.)

2.5 Inter-neural cooperation

After the completion of a trial and the subsequent updating of each neuron's weight according to its eligibility, inter-neural cooperation takes place. This consists of each neuron updating its weight a second time, this time based on the weight values of the other neurons in its neighborhood. For the present application, the neighborhood size is initially 2 and shrinks to 1 after the first 100 trials.

Each individual neuron i is influenced by its neighbors according to the following equation:

$$w_i = (1 - \alpha(T))w_i + \alpha(T) \sum_{n \in N_i} \frac{w_n}{m} \tag{4}$$

where each w is a weight, N_i is the neighborhood of neuron i, m is the number of neurons in that neighborhood, and α determines the degree to which a neuron's value is "smoothed" with that of its neighbors. In general, α decreases with time. This means that each neuron's value becomes more independent from those of its neighbors as time passes. In this particular application, α is defined to be

$$\alpha(T) = \frac{1}{2 + (T-1) \bmod 100} \tag{5}$$

where T is the trial number. (I.e. For the first 100 trials, α is $1/2$, for the second 100 it is $1/3$, and so on.)

3 Application

A ROLNNET system has previously been applied to the task of backing a car and trailer to a goal, both in simulation and using a real robot [5]. In the present paper, the task is complicated by adding a second trailer behind the first. Now the control system has three inputs: (1) the angle of hitch 1 between the car and the first trailer, (2) the angle of hitch 2 between the first and second trailers, and (3) the angle between the spine of the second trailer and the goal, as shown in Figure 1. The addition of the second trailer adds a dimension to the input space of the problem when compared with [5]. This is reflected in an additional dimension to the ROLNNET system used.

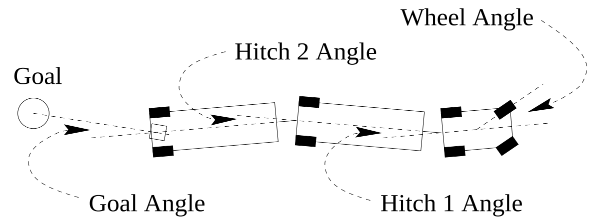


Figure 1: The two-trailer-backing problem

The feedback to the learning system is a simple boolean value. If the rear of the second trailer reaches the goal, success is signaled. If the absolute value of any of the angles exceeds 90°, failure is signaled. The system is clocked to operate in discrete time units. No other sensory data or feedback is available.

Whereas many simulated mobile robot systems include such variables as the x and y position of the robot in Cartesian space, we do not. This is primarily because our system is designed to operate in the real world where it would be difficult for the robot to acquire its own x and y coordinates.

3.1 Experimental design

We studied two cases that varied in the range of possible initial car-trailer-trailer positions. For both cases, the system was trained in a series of runs of 1000 trials each.

Case 1: the rig was started with the back of the second trailer from two to five feet from the target, and with each of the three angles between $-6°$ and $+6°$. New initial conditions were chosen randomly at the start of each trial with a uniform distribution over the entire range.

Case 2: this is identical to case 1, except that the rig was started with the back of the second trailer from three to six feet from the target and all angles between $-5°$ and $+5°$ each.

While the maximum initial angles specified in the case descriptions may seem small, it should be noted that the inherent instability in backing a rig with passive trailers means that, for a two trailer system like the one described, large differences in angles will inevitably lead to a failure state, regardless of the control signal given. This is because if the signs of the angles between adjacent units in the rig are opposite the active car must swing out in arcs wider than those traced by the first trailer in order to move it around into position to correct the trajectory of the second trailer. However, in trying to bring the car into such a position, the angle between the trailers and that between the second trailer and the goal are increased in magnitude as the entire rig backs up. In order for the system to be able to physically maneuver to the goal with larger possible initial angles, it would be necessary for the system to utilize forward motion, as well as backing. In fact, a simple forward move of even a short distance with the wheels straight would serve to bring even a doubly jackknifed rig to within the bounds given for our initial angles in these cases, and so our limits are sufficient to also cover those cases where such an initial forward motion is possible.

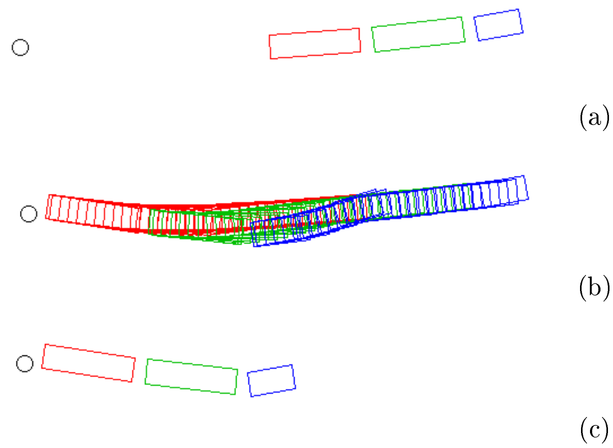


Figure 2: Example trial. (a) start, (b) progress (shown every fifteenth time step), (c) end.

A total of 100 runs (of 1000 trials each) were made in simulation for each of the cases, using different random seeds for both the initial starting positions for each trial and for the random initial values of the output weights. At the start of each trial, the eligibility for adaptation for each neuron was set to zero. On each time step the learning system was given the current values of the trailer and hitch angles and, if applicable, a failure or success signal. The network response was thresholded and values less than zero were used as *hard left* control signals, while values equal to or greater than zero are used as *hard rights*. In this way, the system was given *bang-bang* control. An example trial is shown in Figure 2 and the averages over all runs are plotted in the graphs in Figure 3. The horizontal axis on each graph gives the trial number while the vertical axis gives the success rate.

3.2 Experimental results

The system described in this paper must meet many varied and sometimes conflicting challenges. Some of these are inherent in the task to be learned, some are imposed by our desire to use small yet autonomous mini-robots, and some we have imposed on ourselves to study learning in completely unsupervised domains. The results in this section show that the system is able to function in the demanding environment in which we have placed it.

The three most prominent features of the results graphs are the two relatively flat regions, one at the beginning of each graph and the other at the end, and the steep slope in between. All of these features are significant.

The first flat region, running from trial zero to approximately trial 100 reflects the neighborhood size of two that is present during that time period. After trial 100, the neighborhood size shrinks to 1. Also, during the first 100 trials the degree of “smoothing,” as determined by α, is at its greatest (see 2.5).

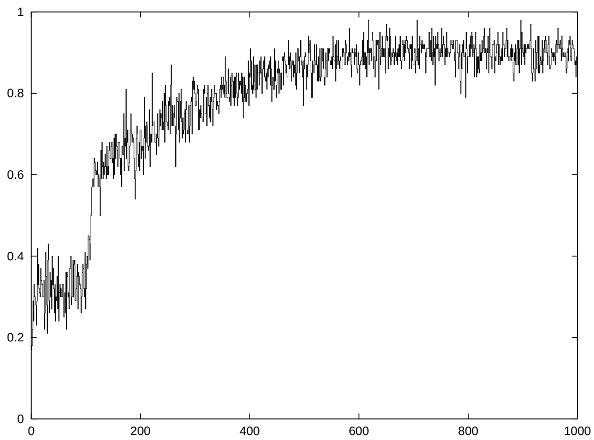
These factors appear to prevent the network from learning the distinctions necessary to succeed regularly in backing the trailers to the goal. What the graph does not show, however, is that if the period during which the neighborhood size is 2 is greatly reduced, the system never approaches the level of success given here. The larger initial neighborhood plays an important role in forming a generalized representation of the mapping that can be built upon by later specific changes.

The steep slope of both graphs starting after trial 100 shows that ROLNNET does in three dimensions exactly what it was created to do in two. The system was designed to rapidly acquire proficiency and this slope indicates that significant learning occurs on failure as well as success.

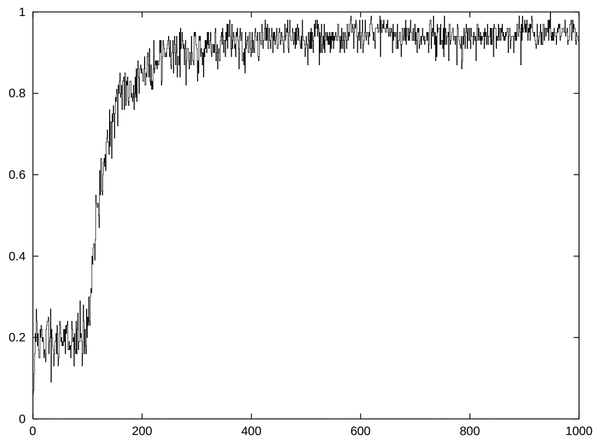
The final region of the graph – the second flat region – simply shows the ROLNNET system approaching its maximum proficiency and settling into a stable state.

4 Related work

The trailer-backing problem is gaining attention as a simple to understand yet difficult to solve learning-control problem. Approaches such as the Cerebellar Model Articulated Controller (CMAC) [18], adaptive fuzzy systems [9, 6], backpropagation through time [14], “fuzzy BOXES” [20], genetic algorithms [10], and our own ROLNNET [5] have all been applied to versions of this problem. It is difficult to directly compare



Graph of case 1. Average performance for 100 runs of 1000 trials each. Initial position: Rear of second trailer 2 to 5 feet from target, all three input angles $+/-6^\circ$ each.



Graph of case 2. Average performance for 100 runs of 1000 trials each. Initial position: Rear of second trailer 3 to 6 feet from target, all three input angles $+/-5^\circ$ each.

Figure 3: Results

results across many of these systems and with the results presented here.

The most obvious difference between this study and those of other authors writing on this problem, is that our system is here applied to a rig with a car and two trailers, whereas all previous work of which we are aware applies learning systems only to a car with a single trailer. Other authors have explored the control of rigs with multiple trailers (e.g. [12]) but these have not been learning systems.

While backing a rig with a single trailer is still a nonlinear and unstable problem, it is significantly less difficult than backing a rig with two trailers. As an indication of this, note that human truck drivers do not even attempt to back two-trailer rigs to loading docks – instead they separate the trailers and back them each individually as single-trailer rigs.

The additional difficulty in backing a rig with a second trailer is reflected in differences between the present work and its immediate predecessor [5] in two ways. First, the ROLNNET system itself has been expanded. For backing a rig with a single trailer, a two-dimensional topology was sufficient. For a rig with two trailers, a three dimensional topology is needed. With this additional dimension comes an increase in the number of neural elements needed to maintain the same grain of partitioning of the input space; for two dimensions 64 neurons were used but for three dimensions this number jumps to 512. Second, the number of trials needed to achieve good performance is increased. In the one-trailer system, maximum performance was reached in 40 to 100 trials, depending on the case examined. In the two-trailer system, the cases examined required 400 to 600 trials to approach their apparent maximum proficiency.

The extension of the ROLNNET topology to three dimensions is significant from both biological and practical viewpoints. It is well known that of the sensory and/or motor maps found in biological neural networks (brains), many are two-dimensional while many others are three-dimensional. Any artificial neural network, then, needs to be extensible to three dimensions to retain biological plausibility. Further, since the three-dimensional mappings in the biological systems are presumably performing useful functions, this implies that there are many useful functions that can be captured by similar three-dimensional artificial neural networks.

Another difference between our work and that of other authors studying the application of learning systems to trailer-backing is that ROLNNET was designed for use on a real robot with very limited computing power and memory. While there have in the last couple of years been an encouraging number of researchers looking at trailer-backing using real robots (e.g. [15, 19, 11]), these have not been learning systems. Our decision to apply learning systems to real robots has ensured that ROLNNET systems are capable of handling noise and of rapid learning.

Another difference is the formulation of the problem with regard to information provided to the learning systems, either by the feedback system or in the choice of input vectors. Our system is completely unsupervised and receives only terminal feedback signals (see Section 1) and, as mentioned above (Section 3), we provide only the angles of the hitches and to the goal, not the x and y coordinates, as input.

All of the differences taken in combination make it clear that simple comparison of success rates between ROLNNET and other learning systems for trailer-backing are of no value. Nonetheless, ROLNNET is seen to learn remarkably quickly even on a very difficult problem.

5 Future work

While ROLNNET was designed for use on a real robot, and its learning is extraordinarily fast, we are still working to decrease the amount of training time necessary to achieve maximum performance. Our experience with a ROLNNET system for a rig with a car and a single trailer indicated that learning on-board a real robot took approximately the same number of trials as it did in simulation to achieve a similar success rate. Given the hardware used in the single trailer experiments, each run of 400 to 600 trials necessary for the two trailer experiments would take from 1 to 1.5 hours to complete. The time needed to collect a meaningful number of runs, therefore, is not insignificant. We are exploring hardware and software options for decreasing the time needed for each trial as well as additions to the learning system to reduce the total number of trials needed.

6 Conclusions

We have described ROLNNET, a paradigm for learning simple tasks for real robots and we have presented experimental evidence to support our approach. The extension of a ROLNNET system to three dimensions is seen to be significant. Its application to a very difficult control-learning problem and a design that allows it to operate with limited computing power and feedback set ROLNNET systems apart from other control-learning systems. The paradigm is rich with possibilities for further study, including novel network architectures and hybridization with other systems (such as self-learning critics).

References

[1] S. Baluja. Evolution of an artificial neural network based autonomous land vehicle controller. *IEEE Trans. on Systems, Man, and Cybernetics*, 26, Part B(3), 1996.

[2] J. del R. Millán. Rapid, safe, and incremental learning of navigation strategies. *IEEE Trans. on Systems, Man, and Cybernetics*, 26, Part B(3), 1996.

[3] D. Floreano and F. Mondada. Evolution of homing navigation in a real mobile robot. *IEEE Trans. on Systems, Man, and Cybernetics*, 26, Part B(3), 1996.

[4] D. F. Hougen. Use of an eligibility trace to self-organize output. In *Science of Artificial Neural Networks II, Proc. SPIE*, vol. 1966, pp. 436–447, 1993.

[5] D. F. Hougen, J. Fischer, M. Gini, and J. Slagle. Fast connectionist learning for trailer backing using a real robot. In *Proc. of the Int'l Conf. on Robotics and Automation*, pp. 1917–1922, 1996.

[6] H. Ichihashi and M. Tokunaga. Neuro-fuzzy optimal control of backing up a trailer truck. In *Proc. of the IEEE Int'l Conf. on Neural Networks*, pp. 306–311, 1993.

[7] A. Klopf. Brain function and adaptive systems – a heterostatic theory. In *Proc. of the Int'l Conf. on Systems, Man, and Cybernetics*, 1974.

[8] T. K. Kohonen. *Self-organizing and associative memory.* Springer-Verlag, Berlin, 3rd ed., 1989.

[9] S.-G. Kong and B. Kosko. Adaptive fuzzy systems for backing up a truck-and-trailer. *IEEE Trans. on Neural Networks*, 3(2):211–223, 1992.

[10] J. R. Koza. A genetic approach to finding a controller to back up a tractor-trailer truck. In *Proc. of the American Control Conference*, pp. 2307–2311, 1992.

[11] U. Larsson, C. Zeli, K. Hyyppä, and Å. Wernersson. Navigating an articulated vehicle and reversing with a trailer. In *Proc. of the Int'l Conf. on Robotics and Automation*, pp. 2398–2404, 1994.

[12] J.-P. Laumond. Controllability of a multibody mobile robot. *IEEE Trans. on Robotics and Automation*, 9(6):755–763, December 1993.

[13] W. T. Miller, III, R. S. Sutton, and P. J. Werbos. *Neural Networks for Control.* MIT Press, Cambridge, MA, 1990.

[14] D. Nguyen and B. Widrow. Neural networks for self-learning control systems. *IEEE Control Systems Magazine*, 10(3):18–23, 1990.

[15] R. Parra-Loera and D. J. Corelis. Expert system controller for backing-up a truck-trailer system in constrained space. In *Proc. of the 37th Midwest Symposium on Circuits and Systems*, pp. 1357–1361, 1995.

[16] S. M. Prabhu and D. P. Garg. Artificial neural network based robot control: An overview. *Journal of Intelligent and Robotic Systems*, 15:333–365, 1996.

[17] H. Ritter, T. Martinetz, and K. Schulten. *Neural computation and self-organizing maps: an introduction.* Addison-Wesley, Reading, MA, 1992.

[18] R. O. Shelton and J. K. Peterson. Controlling a truck with an adaptive critic CMAC design. *Simulation*, 58(5):319–326, 1992.

[19] K. Tanaka. Model-based fuzzy control of a trailer type mobile robot. In *Proc. of the IEEE Int'l Conf. on Fuzzy Systems*, pp. 65–70, 1995.

[20] N. Woodcock, N. J. Hallam, and P. D. Picton. Fuzzy BOXES as an alternative to neural networks for difficult control problems. *Artificial Inteligence in Engineering*, pp. 903–919, 1991.