

# Neural Networks for Control

Edited by W. Thomas Miller, III, Richard S. Sutton, and Paul J. Werbos

[1991]

A Bradford Book

The MIT Press  
Cambridge, Massachusetts  
London, England

## 12 The Truck Backer-Upper: An Example of Self-Learning in Neural Networks

Derrick Nguyen and Bernard Widrow

### 12.1 Introduction

The control of severely nonlinear systems has for the most part escaped the attention of control theorists and practitioners. This chapter addresses the issue from the point of view of utilizing self-learning techniques to achieve nonlinear controller design. The methodology shows promise for applications to control problems that are so complex that analytical design techniques either do not exist or will not exist for some time to come. Neural networks can be used to implement highly nonlinear controllers whose weights or internal parameters can be chosen or determined by a self-learning process.

Backing a trailer truck to a loading dock is a difficult exercise for all but the most skilled truck drivers. Anyone who has tried to back up a house trailer or a boat trailer will realize this. Normal driving instincts lead to erroneous movements. A great deal of practice is required to develop the requisite skills.

When watching a truck driver backing toward a loading dock, one often observes the driver backing, going forward, backing again, going forward, etc., and finally backing to the desired position along the dock. The forward and backward movements help to position the trailer for successful backing up to the dock. A more difficult backing up sequence would only allow backing, with no forward movements permitted. The specific problem treated in this paper is that of the design by self-learning of a nonlinear controller to control the steering of a trailer truck while backing up to a loading dock from an arbitrary initial position. Only backing up is allowed. Computer simulation of the truck and its controller has demonstrated workability, although no mathematical proof yet exists. The experimental controller contains twenty six adaptive ADALINE units (Widrow and Hoff 1960) and exhibits exquisite backing up control. The trailer truck can be initially "jackknifed" and aimed in many different directions, toward and away from the dock, but as long as there is sufficient clearance, the controller appears to be capable of finding a solution.

Figure 12.1 shows a computer-screen image of the truck, the trailer, and the loading dock. The critical state variables representing the position of the truck and that of the loading dock are  $\theta_{cab}$ , the angle of

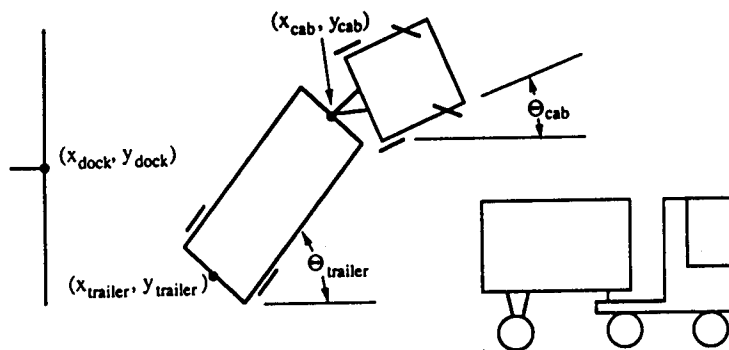


Figure 12.1  
The truck, the trailer, and the loading dock.

the truck,  $x_{cab}$  and  $y_{cab}$ , the Cartesian position of the yoke,  $x_{trailer}$  and  $y_{trailer}$ , the Cartesian position of the rear of the center of the trailer, and  $x_{dock}$  and  $y_{dock}$ , the Cartesian position of the center of the loading dock. Definition of the state variables is illustrated in figure 12.1.

The truck backs up until it hits the dock, then stops. The goal is to cause the back of the trailer to be parallel to the loading dock, and to have the point  $(x_{trailer}, y_{trailer})$  be aligned as closely as possible with point  $(x_{dock}, y_{dock})$ . The controller will learn to achieve this objective.

## 12.2 Training

The approach to self-learning control that has been successfully used with the truck backer-upper involves a two-stage learning process. The first stage involves the training of a neural network to be an emulator of the truck and trailer kinematics. The second stage involves the training of a neural-network controller to control the emulator. A similar approach has been used by Widrow (1986, Widrow and Stearns 1985) and by Jordan (1988). Once the controller knows how to control the emulator, it is then able to control the actual trailer truck. Figure 12.2 gives an overview, showing how the present state vector  $state_k$  is fed

to the controller which in turn provides a *steering signal*,  $k$  between  $-1$  (hard right) and  $+1$  (hard left) to the truck. The time index is  $k$ . Each time cycle, the truck backs up by a fixed small distance. The next state is determined by the present state and the steering signal, which is fixed during the cycle.

Figure 12.3 shows a block diagram of the process used to train the emulator. The truck backs up randomly, going through many cycles with randomly selected steering signals. By this process, the emulator "gets the feel" of how the trailer and truck behave. The emulator, chosen as a two-layer neural network, learns to generate the next positional state vector when given the present-state vector and the steering signal. This is done for a wide variety of positional states and steering angles. The two-layer emulator is adapted by means of the backpropagation algorithm (Werbos 1984, Parker 1985, Rumelhart and McClelland 1986). The first layer had six present-state inputs plus the present steering signal input. This layer contained forty-five hidden adaptive ADALINE units producing six next-state predictions. Once the emulator is trained, it can then be used to train the controller.

Refer to figure 12.4. The identical blocks labeled C represent the controller. The identical blocks labeled T represent the truck and trailer emulator. Suppose that the truck is engaged in backing up. Let C be chosen randomly and be initially fixed. The initial state vector  $s_0$  is fed to C, which produces the steering signal output which sets the steering angle of the truck. The backing up cycle proceeds with the truck and trailer soon arriving at the next state  $s_1$ . With C remaining fixed, the backing up process continues from cycle to cycle until the truck hits something and stops. The final state  $s_K$  is compared with the desired final-state (the rear of the trailer parallel to the dock with proper positional alignment) to obtain the final state error vector  $\epsilon_K$ . This error vector contains three elements (which are the errors of interest),  $x_{trailer}$ ,  $y_{trailer}$  and  $\theta_{trailer}$ , and is used to adapt the controller C.

The method of adapting the controller C is illustrated in figure 12.5. The final-state error vector  $\epsilon_K$  is used to adapt the blocks labeled C, which are maintained identical to each other throughout the adaptive process. The controller C is a two-layer neural network. The first layer has the six state variables as inputs, and this layer contains twenty five adaptive ADALINE units. The second or output layer has one adaptive ADALINE unit and produces the steering signal as its output.

The procedure for adapting C goes as follows. The weights of C are chosen initially at random. The initial position of the truck is chosen at random. The truck backs up, undergoing many individual back up

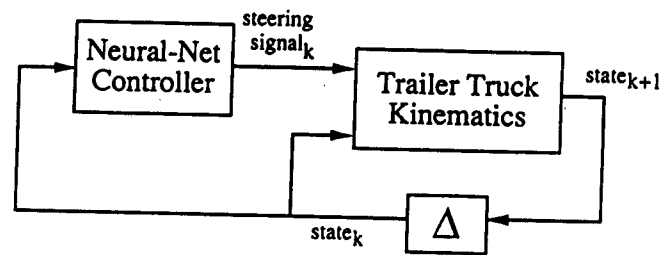


Figure 12.2  
Overview diagram.

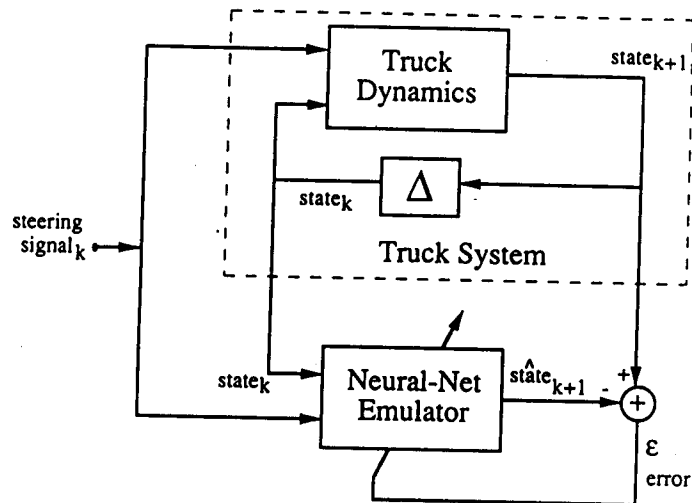


Figure 12.3  
Training the neural-net truck emulator.

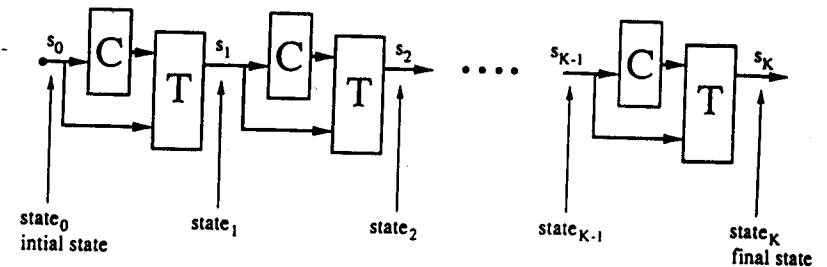


Figure 12.4  
State transition flow diagram.

cycles, until it stops. The final error is used by backpropagation to adapt the controller. Each of the C blocks could be tentatively adapted by backpropagation if they were independent of each other, but the actual weight changes in C are taken as the sum of the tentative changes. In this way, the C blocks are maintained identical to each other. The weights are changed by this constrained backpropagation algorithm to reduce the sum of the squares of the components of the final-state error  $\epsilon_K$  by following the negative of the gradient, using the method of steepest descent. The entire process is repeated by placing the truck and trailer in another initial position, and allowing it to back up until it stops. Once again, the controller weights are adapted. And so on.

Figure 12.6 and figure 12.7 show details of one of the state transition stages of figure 12.5. One can see the structure of controller C and of emulator T, and how they are interconnected. Each stage of figure 12.5 amounts to a four-layer neural network. The entire process of going from an initial state to the final state can be seen from figures 12.4 and 12.5 to be analogous to a neural network having a number of layers equal to four times the number of backing up steps when going from the initial state to the final state. The number of steps varies of course with initial position of the truck and trailer.

The diagram of figure 12.5 was simplified for clarity of presentation. The output error does not go directly to the C-blocks as shown, but backpropagates through the T-blocks and C-blocks. Thus, the error used to adapt each of the C-blocks does originate from the output error  $\epsilon_K$ , but travels through the proper backpropagation paths. For purposes

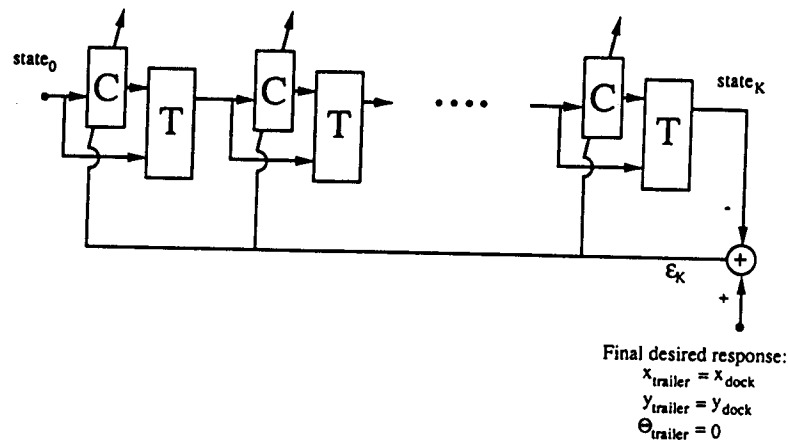


Figure 12.5  
 Training the controller with backpropagation.

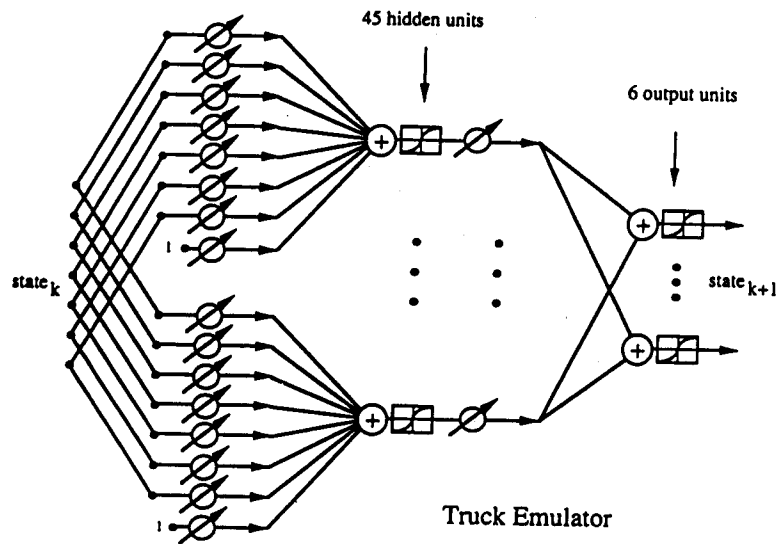


Figure 12.6  
 Details of emulator and controller.

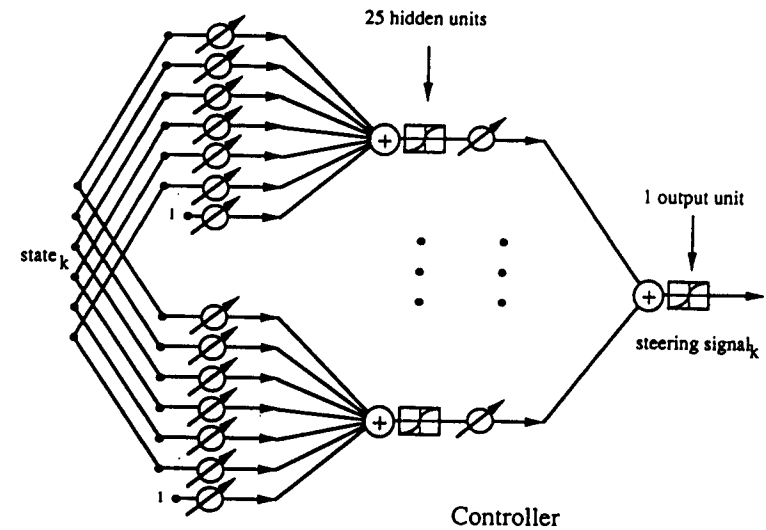


Figure 12.7  
 Details of emulator and controller.

of back-propagation of the error, the T-blocks are the truck emulator. But the actual truck kinematics are used when sensing the error  $\epsilon_k$  itself.

### 12.3 Summary and Results

The truck emulator was able to represent the trailer and truck when jackknifed, in line, or in any condition in between. Nonlinearity in the emulator was essential to represent the truck and trailer. The angle between truck and trailer were not small.  $\sin \theta$  could not be represented approximately as  $\theta$ . Nonlinearity in the controller was also essential. Self-learning processes were used to determine the parameters of both the emulator and the controller. Thousands of backups were required to train these networks. Without the learning process however, substantial

amounts of human effort and design time would have been required to devise the controller.

The training of the controller was divided into several "lessons." In the beginning, the controller was trained with the truck initially set to points very near the dock and the trailer pointing at the dock. Once the controller was proficient at working with these initial positions, the problem was made harder by starting the truck farther away from the dock and at increasingly difficult angles. This way, the controller learned to do easy problems first and more difficult problems after it mastered the easy ones. There were sixteen lessons in all. In the easiest lesson the trailer was set about half a truck length from the dock in the  $x$  direction pointing at the dock, and the cab at a random angle between  $\pm 30$  degrees. In the last and most difficult lesson the rear of the trailer was set randomly between 1 and 2 truck lengths from the dock in the  $x$  direction and  $\pm 1$  truck length from the dock in the  $y$  direction. The cab and trailer angle was set to be the same, at a random angle between  $\pm 90$  degrees. The controller was trained for about 1000 truck backups per lesson during the early lessons, and 2000 truck backups per lesson during the last few. It took altogether about 20000 backups to train the controller.

The controller learned to control the truck very well with the above training process. Near the end of the last lesson, the root mean square error of  $y_{trailer}$  was about 3 percent of a truck length. The root mean square error of  $\theta_{trailer}$  was about 7 degrees. There is no error in  $x_{trailer}$  since a truck backup is stopped when  $x_{trailer} = x_{dock}$ . One may, of course, trade off the error in  $y_{trailer}$  with the error in  $\theta_{trailer}$  by giving them different weightings during training.

Results with the adaptive controller are illustrated in figures 12.8 through 12.10. The controller has already been trained and its weights remained fixed for all the experiments. The truck and trailer were placed in a variety of initial conditions, and backing up was effected in each case. Initial and final states are shown in the computer screen displays, and the dynamics of backing up is illustrated by the time-lapse plots.

The truck backer-upper learns to solve sequential decision problems. The control decisions made early in the backing up process have substantial effects upon final results. Early moves may not always be in a direction to reduce error, but they position the truck and trailer for ultimate success. In many respects, the truck backer-upper learns a control strategy that is like a dynamic programming problem solution. The learning is done in a layered neural network. Connecting signals from

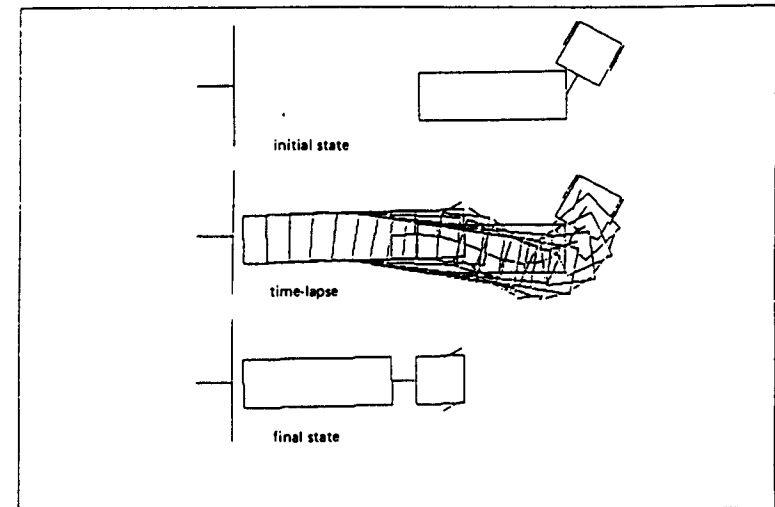


Figure 12.8  
Results of training.

one layer to another corresponds to idea that the final state of a backing up cycle is the same as the initial state of the next backing up cycle.

Future research will be concerned with:

- Determination of complexity of emulator as related to complexity of the system being controlled.
- Determination of complexity of controller as related to complexity of emulator.
- Determination of convergence and rate of learning for emulator and controller.
- Proof of robustness of control scheme.
- Analytic derivation of nonlinear controller for truck backer-upper, and comparison with self-learned controller.
- Relearning in the presence of movable obstacles.

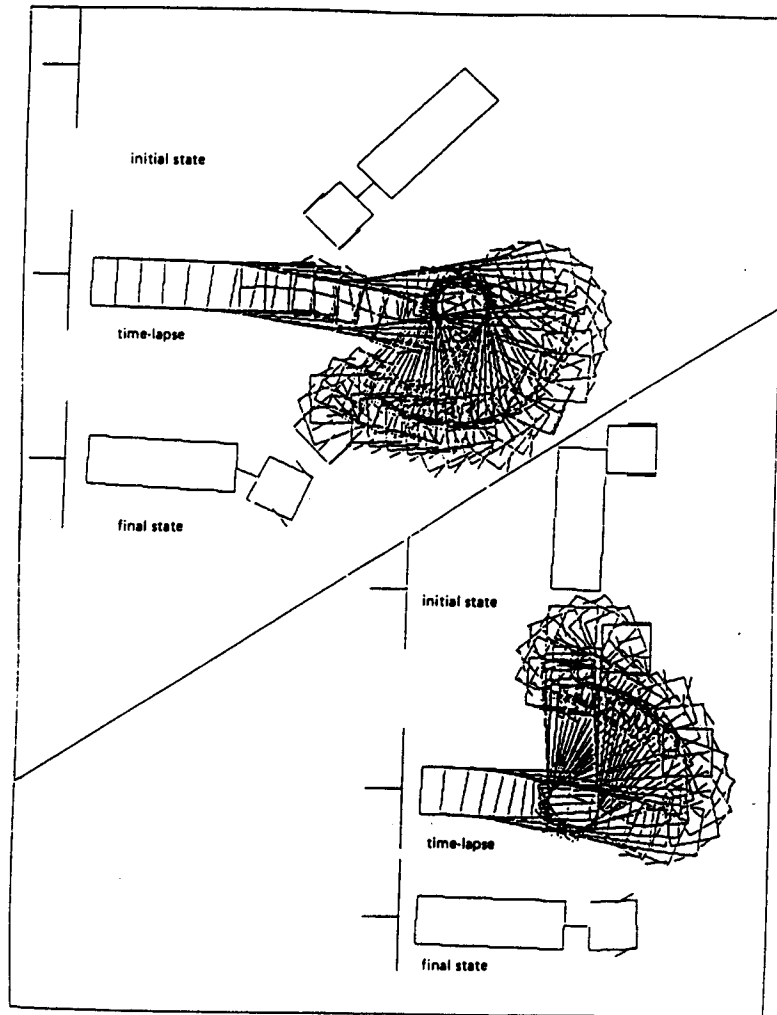


Figure 12.9  
Results of training.

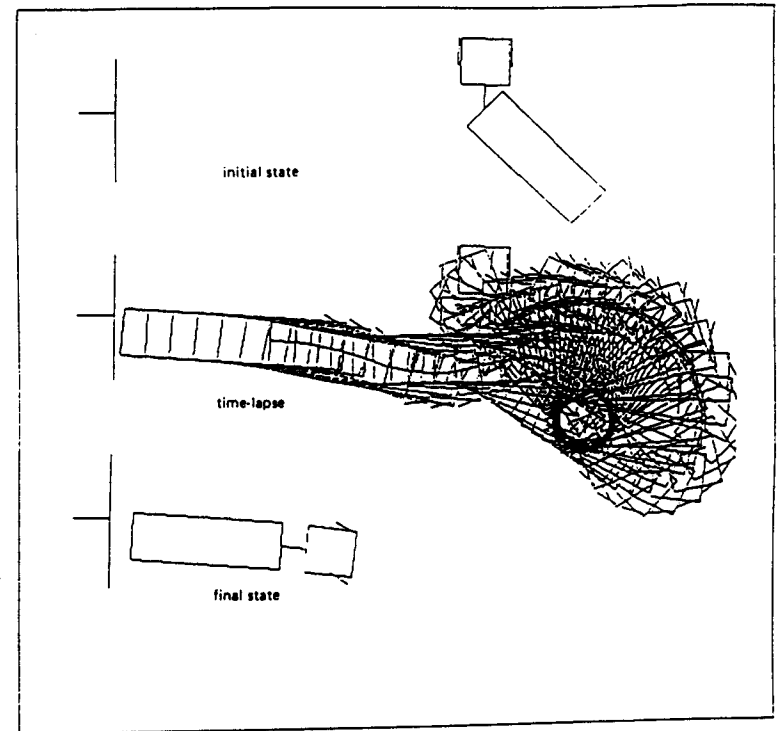


Figure 12.10  
Results of training.

- Exploration of other areas of application for self-learning neural networks.

### Acknowledgments

This research was sponsored by SDIO Innovative Science and Technology Office and managed by ONR under contract #N00014-86-K-0718, by the Department of the Army Belvoir Research, Development, and Engineering Center under Contract #DAAK70-89-K-0001, and by a grant from the Thomson CSF Company.

This material is based on work supported under a National Science Foundation Graduate Fellowship. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

### References

- Jordan, M. I. (1988). Supervised learning and systems with excess degrees of freedom. COINS Technical Report 88-27, Massachusetts Institute of Technology, Cambridge, MA.
- Parker, D. B. (1985). Learning logic. Technical Report TR-47, Center for Computational Research in Economics and Management Science, Massachusetts Institute of Technology, Cambridge, MA.
- Rumelhart, D. E., and McClelland, J. L., eds. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition*. Vol. 1, Foundations, chap. 8. Cambridge, MA: MIT Press.
- Werbos, P. (1984). Beyond regression: New tools for prediction and analysis in the behavioral sciences. Ph.D. diss., Harvard University, Cambridge, MA.
- Widrow, B. (1986). Adaptive inverse control. In *Adaptive systems in control and signal processing*, International Federation of Automatic Control, July. Pergaman Press, Lund, Sweden.
- Widrow, B., and Hoff, M. E., Jr. (1960). Adaptive switching circuits. *IRE WESCON Convention Record*, pt. 4:96-104.

- Widrow, B. and Stearns, S. D. (1985). *Adaptive signal processing*. Englewood Cliffs, NJ: Prentice Hall.