Comparison of Fuzzy and Neural Truck Backer-Upper Control Systems

Seong-Gon Kong and Bart Kosko
Department of Electrical Engineering
Signal and Image Processing Institute
University of Southern California
Los Angeles, California 90089-0272

Fuzzy and Neural Control Systems

Fuzzy and neural control systems are constructed directly from control data, but from different types of control data. Fuzzy systems use a small number of structured *linguistic* input-output samples from an expert or from some other adaptive estimator. Neural systems use a large number of *numeric* input-output samples from the control process or from some other database.

Figure 1 illustrates this difference. The neural system estimates function $f : X \longrightarrow Y$ from several numerical *point* samples (x_i, y_i). The fuzzy system estimates f from a few fuzzy *set* samples or fuzzy associations (A_i, B_i).

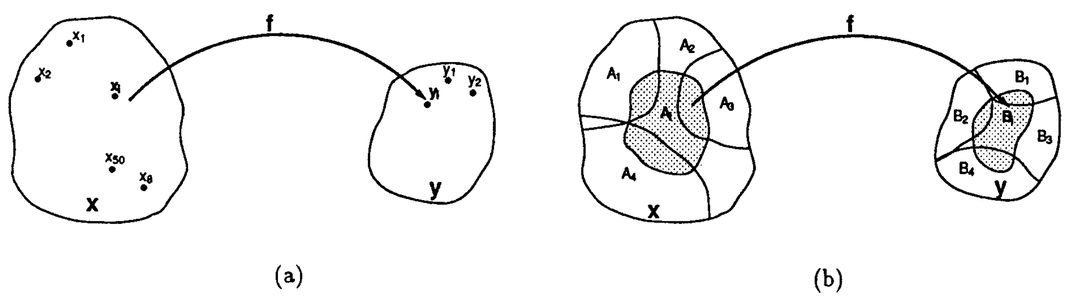


FIGURE 1 Geometry of neural and fuzzy function estimation. The neural approach (a) uses several numerical point samples. The fuzzy approach (b) uses a few fuzzy set samples.

The advantage of fuzzy and neural systems over traditional control approaches is *model-free estimation.* The user need not specify how the controller's output mathematically depends on its input. Instead the user provides a few common-sense associations of how the control variables behave. Or the user provides a statistically representative set of numerical training samples. Even if a math-model controller is available, fuzzy or neural controllers may prove more robust and easier to modify.

Which approach, fuzzy or neural, is best for which type of control problem depends on the type and availability of sample data. If structured knowledge of the control process is available, or if sufficient numerical training samples are unavailable, the fuzzy approach is preferable. A fuzzy control system is comparatively simple to construct and use when structured knowledge is available. A fuzzy control system seems a reasonable benchmark in such cases, even if a neural controller or math-model controller is developed.

If representative numerical data is available and structured expertise is not, the neural approach is preferable. Or a statistical regression approach may be more approapriate. The data simply tell their own story — if there is a story to tell. Yet even here a hybrid fuzzy-neural system may be preferable. The numerical data can always be used, perhaps by neural systems, to generate *fuzzy associative memory* (FAM) rules. The FAM rules can then form the skeleton of a fuzzy control *architecture*. In other words, if structured knowledge is unavailable, estimate it. This may be more practical than it would appear because of the small number of control FAM rules needed to reliably control many realworld processes.

How can fuzzy and neural controllers be compared? Abstract comparison is difficult because both approaches build a control black box in different ways. That they build black boxes distinguishes them from math-model controllers. It also suggests they should be compared by their black-box control performance.

Each control system generates an output *control surface* as it ranges over the common input space of parameter values. Figure 5 below shows a three-dimensional control surface for a fuzzy controller. For control systems with few input parameters with moderately quantized ranges, both fuzzy and neural controllers — or rather their quantized control surfaces — can be stored as decision look-up tables. The two controllers are, after all, deterministic algorithms. Then once a system performance criterion is specified, the controllers can in principle be quantitatively compared.

Comparing system trajectories is more complicated. In the case at hand, we are interested in backing up a truck to a loading dock. The quality and quantity of the truck trajectory can be measured and compared, perhaps with mean-squared error criteria. Intuitively, smooth short trajectories are preferred to jagged long trajectories. Reaching the loading-dock goal is also important. In practice it is the most important performance requirement. It is unclear how to ideally balance the trajectory type with the trajectory destination. For this reduces to the pragmatic issue of balancing means and ends.

Below we develop a simple fuzzy control system and a simple neural control system for backing up a truck in an open parking lot. Our choice of control problem was motivated by the recent, and successful, neural network truck backer-upper simulation of Nguyen and Widrow [1989]. We were unable to exactly replicate the neural network they used. Instead we built the best backpropagation network we could with essentially the same kinematics and compared it to the best fuzzy controller we could develop.

The fuzzy controller compares favorably with the neural controller in terms of black-box development effort, black-box computational load, smoothness of truck trajectories, and robustness.

We studied robustness of the fuzzy controller by both deliberately adding confusing FAM rules — "sabotage" rules — to the system and by randomly removing different subsets of FAM rules. We studied robustness of the neural controller by randomly removing different portions of the training data.

These simulations suggest fuzzy controllers may be a practical alternative to neural-architecture controllers in many cases.

Backing up a truck

The simulated truck and loading zone are shown in Figure 2. The truck corresponds to the cab part of the neural truck in the Nguyen-Widrow neural truck backer-upper system. The truck position is exactly determined by the three state variables ϕ, $\mathbf{x}$, and $\mathbf{y}$. ϕ is the angle of the truck with the horizontal. The coordinate pair $(\mathbf{x},\mathbf{y})$ specifies the position of the rear center of the truck in the plane.

The goal is to make the truck arrive at the loading dock at right angle ($\phi_f = 90^o$) and to have the position $(\mathbf{x},\mathbf{y})$ of the truck be aligned with the desired loading dock $(\mathbf{x}_f, \mathbf{y}_f)$. Only backing up is considered. The truck moves backward by some fixed distance every stage. There are K-many such stages until the truck hits the border of the loading zone. The loading zone is the plane $[0,100] \times [0,100]$, and $(\mathbf{x}_f, \mathbf{y}_f)$ is $(50,100)$.

The fuzzy and neural controllers should produce the appropriate steering angle θ at every stage to make the truck back up to the loading dock from any initial position and from any angle in the loading

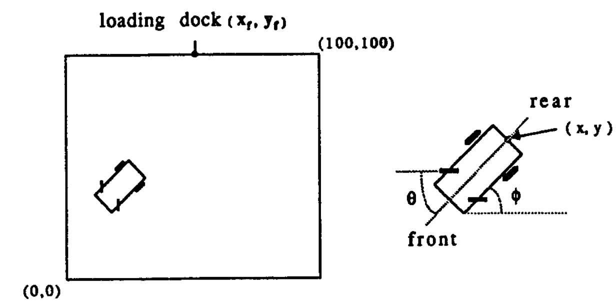


FIGURE 2 Diagram of simulated truck and loading zone.

zone.

Fuzzy Truck Backer-Upper System

The input and output parameters of the controller must be specified. The input parameters are the truck angle $\boldsymbol{\phi}$ and the x-position $\mathbf{x}$. The output parameter is the steering signal $\boldsymbol{\theta}$. If we assume enough clearance between the truck and the loading dock, y-position need not be considered as an input. The parameter ranges are as follows:

$$\begin{array}{rl} \boldsymbol{\phi}: & [\ 0,\ 360\ ] \\ \mathbf{x}: & [\ 0,\ 100\ ] \\ \boldsymbol{\theta}: & [\ -30,\ 30\ ] \end{array}$$

Positive values of $\boldsymbol{\theta}$ are clockwise rotations of the steering wheel. Negative values are counterclockwise rotations. All values are discretized to reduce computation. The resolution of $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ is one degree each. The resolution of $\mathbf{x}$ is 0.1.

Fuzzy subsets of the input and output parameters must be specified. The fuzzy sets numerically represent linguistic terms, the sort of linguistic terms an expert might give to describe the control system's behavior. Fuzzy subsets of parameter values are chosen to fit the problem at hand. The fuzzy subsets for the truck backer-upper controller are chosen as follows:

Angle $\boldsymbol{\phi}$	x-position $\mathbf{x}$	Steering signal $\boldsymbol{\theta}$
RB: Right Below	LE: Left	NB: Negative Big
RU: Right Upper	LC: Left Center	NM: Negative Medium
RV: Right Vertical	CE: Center	NS: Negative Small
VE: Vertical	RC: Right Center	ZE: Zero
LV: Left Vertical	RI: Right	PS: Positive Small
LU: Left Upper		PM: Positive Medium
LB: Left Below		PB: Positive Big

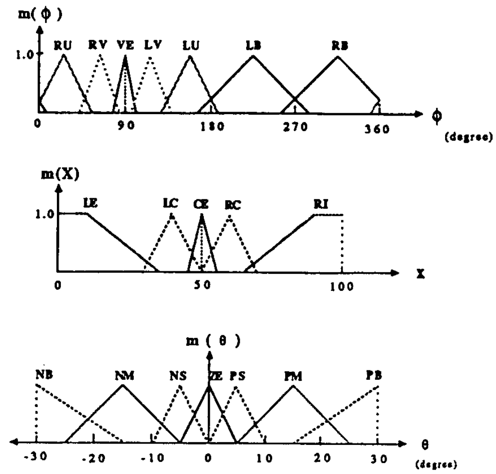


FIGURE 3 Fuzzy membership functions for each linguistic fuzzy set. To allow finer control, the fuzzy sets that correspond to near the loading dock are narrower than the fuzzy sets that correspond to far from the loading dock.

Fuzzy subsets have degrees of membership. A fuzzy membership function $m_A : X \longrightarrow [0,1]$ assigns a real number between 0 and 1 to any object in the universe of discourse X. This number $m_A(x)$ is the degree to which the object or data x belongs to the fuzzy set A.

Fuzzy membership functions can have different shapes depending on the designer's preference or experience. In practice triangular and trapezoidal shapes have proven useful both in capturing the modeler's sense of fuzzy parameter numbers and in computational simplicity. Membership function graphs of the fuzzy subsets above are shown in Figure 3. In the third graph, for example, 20^o is Positive Medium to degree 0.5 but is only Positive Big to degree 0.3.

In Figure 3 the vertical, center, and zero fuzzy sets are narrower than the other fuzzy sets. These narrow fuzzy sets permit fine control near the loading dock. Wider fuzzy sets are used for describing the endpoints of the range of the variables ϕ, $\mathbf{x}$, and θ. The wider fuzzy sets permit rough control far from the loading dock.

The next step is to specify the fuzzy "rulebase" or bank of *fuzzy associative memories* (FAMs). Fuzzy associations or "rules" (A, B) associate output fuzzy sets B of control values with input fuzzy sets A of parameter values. The fuzzy associations can be written as antecedent-consequent pairs or IF-THEN statements.

In the truck backer-upper case, the FAM bank is composed of 35 FAM rules as in Figure 4. For example, the FAM rule of the left upper block is the following fuzzy association:

IF x-position $\mathbf{x}$ is Left *AND* angle ϕ is Right Below,
THEN steering signal θ is Positive Big.

FAM rule 17 denotes that if the truck is in the right position, then the controller should not produce positive or negative steering signal. The FAM rules in the FAM-bank matrix reflect the symmetry of the controlled system.

The three-dimensional control surface in Figure 5 shows steering signal outputs corresponding to all combinations of the two input state values ϕ and $\mathbf{x}$. The control surface defines the fuzzy controller. In this simulation it is determined by the min-max FAM inference procedure discussed in Kosko [1990]. If the control surface changes with sampled parameter values, the system behaves as an *adaptive* fuzzy controller.

The final step is to determine the output action given the input conditions. One possibility is the min-max fuzzy combination method. Each FAM rule produces the output fuzzy set clipped at the degree x-position (X)

		LE	LC	CE	RC	RI
	RB	0 PB	1 PB	2 PM	3 NB	4 NB
	RU	5 ZE	6 PS	7 PM	8 PB	9 PB
	RV	10 NB	11 NM	12 PS	13 PM	14 PB
angle (φ)	VE	15 NB	16 NM	17 ZE	18 PM	19 PB
	LV	20 NB	21 NM	22 NS	23 PM	24 PB
	LU	25 NB	26 NB	27 NM	28 NS	29 ZE
	LB	30 PB	31 PB	32 NM	33 NB	34 NB

FIGURE 4 FAM-bank matrix for the fuzzy truck backer-upper controller.

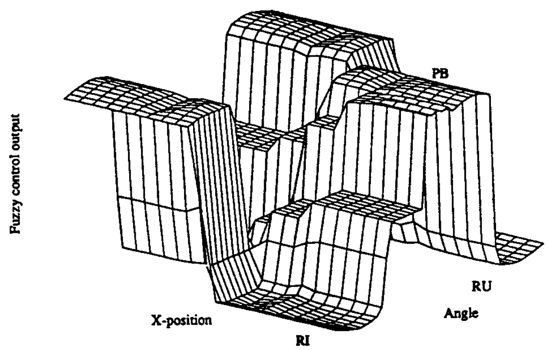


FIGURE 5 Control surface of the fuzzy controller. The output and input combination corresponding to FAM rule 9 (If x is RI and ϕ is RU, then θ is PB.) is illustrated.

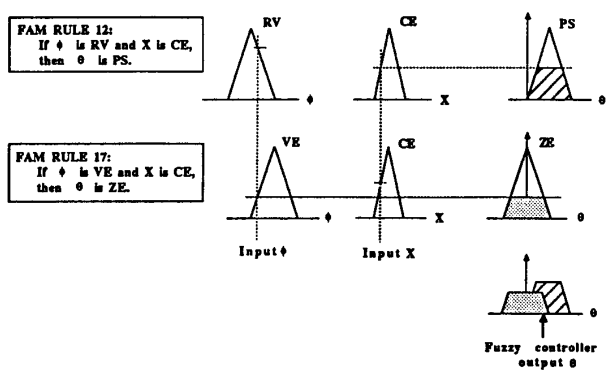


FIGURE 6 Min-max fuzzy inference with centroid defuzzification method. FAM rule antecedents combined with AND use the *minimum* fit value to activate consequents. Those combined with OR use the *maximum* fit value.

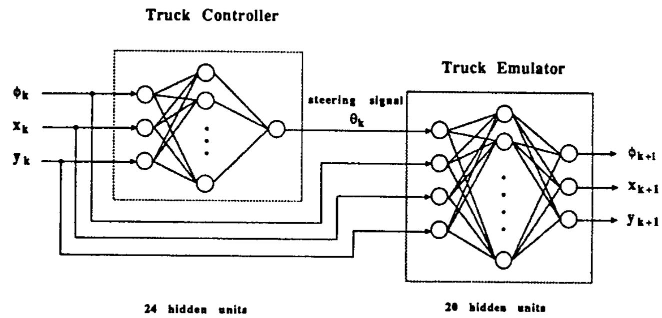


FIGURE 8 Topology of the neural controller.

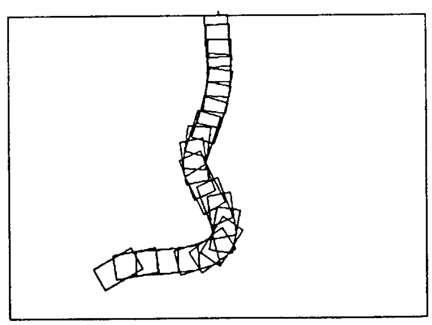


FIGURE 9 Truck trajectory of the neural controller.

and y coordinate and any angle. The *emulator network* computes the next position of the truck. The emulator network takes as input the previous truck position and the current steering output computed by the controller network. The controller network uses 24 "hidden" neural units. The emulator network uses 20 hidden units.

There are K-many back-ups until the truck hits the loading zone. So K-many controller-emulator blocks must be concatenated when the neural system backs up the truck. K depends on the initial truck position and the backing distance of the truck. For example, if the backing distance at every iteration is 5.0 then K is 20 starting from $\phi_0=0°$, $x_0=20$, and $y_0=40$.

The emulator network must be trained before the control network is trained. "Universal" synaptic connection weights of the truck-emulator network were unobtainable since backpropagation learning does not converge for many training patterns. The number of training samples was more than 3000. For example, the combinations of training patterns of a given angle, x-position, y-position, and steering signal might respectively correspond to $18 \times 5 \times 5 \times 7 = 3150$ samples.

Moreover, training patterns are numerically similar since the neuron signal values are restricted to $[0, 1]$ or $[-1, 1]$. Close values such as 0.40 and 0.41 need to be considered as two distinct patterns.

Unlike the Nguyen-Widrow neural controller, our truck emulator network was repeatedly trained with one training sample at every stage to properly back-propagate the error. This did not affect the post-training performance of the neural truck backer-upper since the purpose of the truck-emulator network is

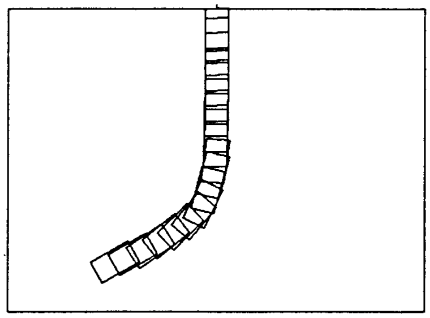


FIGURE 7 Truck trajectory of the fuzzy controller.

of membership determined by the input conditions and the rule. Another possibility combines FAM rules multiplicatively. We use the minimum combination method.

The controller outputs obtained by the FAM inference method are fuzzy sets. These sets must be "defuzzified" to produce numerical (point-estimate) control outputs.

The simplest defuzzification scheme selects the value corresponding to the *maximum* fuzzy unit (*fit*) value [Kosko, 1986] in the fuzzy set. This mode-selection approach ignores most of the information in the output fuzzy set.

A more effective procedure is *centroid* defuzzification. This method, used for example in servo-motor control by Li and Lau [1989], uses as output the *fuzzy centroid*:

$$\text{centroid} = \frac{\sum_{j=1}^{p} m_O(y_j)\, y_j}{\sum_{j=1}^{p} m_O(y_j)},$$

where O is the output fuzzy set defined on the output-parameter universe of discourse $Y = \{y_1, \ldots, y_p\}$. Figure 6 shows the min-max combination and the centroid defuzzification method of the FAM rules 12 and 17. We used centroid defuzzification for all simulations.

With 35 FAM rules, the fuzzy truck controller produces successful truck backing-up trajectories starting from any initial position. Figure 7 shows the truck trajectory by the fuzzy controller starting from the initial position $(\mathbf{x}_0, \mathbf{y}_0)$ of (30,20) and the initial angle ϕ_0 of 30 degrees. All FAM rules are not used at the decision at every stage. In most cases one or two FAM rules were used. At most 4 FAM rules were used at any stage.

Neural Truck Backer-Upper System

The neural truck backer-upper of Nguyen and Widrow [1989] is composed of feedforward multilayer neural networks trained with the backpropagation gradient-descent(stochastic-approximation) algorithm. Figure 8 shows the network connection topology for the neural truck backer-upper considered.

The *neural controller system* consists of two neural networks: the controller network and the truck emulator network. The *controller network* produces an appropriate steering signal output given any x

to back-propagate errors.

The error was computed with simple kinematics. If the truck moves backward by the distance (dx, dy) each iteration, then

$$(x, y)_{new} = (x, y)_{old} + (dx, dy)$$

where $dx = r\cos(\phi + \theta)$, $dy = r\sin(\phi + \theta)$, and r is the fixed driving distance of the truck for all backing movements.

In the training of the truck-controller, ideal steering signal was estimated at each stage before training the controller network. In the simulation, the arc-shaped truck trajectory produced by the fuzzy controller was taken as the ideal trajectory.

Figure 9 shows a typical neural-controlled truck trajectory starting from the initial position (30,20) and the angle 30 degree. Even though the neural network was trained to follow the smooth arc-shaped path, the actual truck trajectory is clearly non-optimal.

Comparison of Fuzzy and Neural Systems

As shown in Figure 7, the fuzzy controller always succeeds at smoothly backing up the truck. The neural controller does not. When it succeeds, the neural-controlled truck often follows an irregular path.

The central problem of the neural truck backer-upper system is its time-consuming training. Thousands of back-ups were required to train both the controller and the emulator networks with the backpropagation algorithm. In some cases, the training procedure did not converge.

The truck emulator network is redundant. It only propagates performance errors backward. As in the Nguyen-Widrow neural truck backer-upper, the kinematic equation $(x, y)_{new} = (x, y)_{old} + (dx, dy)$ is used to determine the performance error.

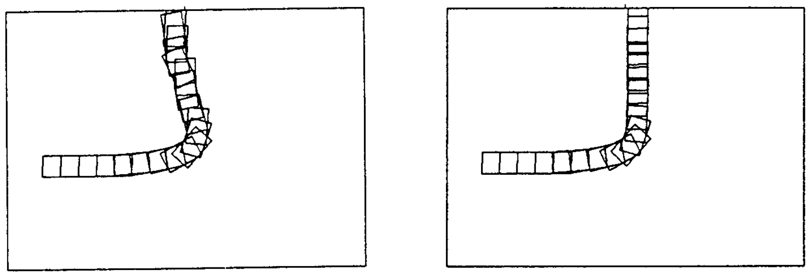
The fuzzy controller was "trained" by inspection and consistently performed well. Once the FAM rule bank is established, control outputs can be obtained from the resulting FAM-bank matrix or control surface. The fuzzy controller did not need a truck emulator. No math model of how outputs depend on inputs was used.

The fuzzy controller is computationally lighter than the neural controller. Most computation operations in the neural controller are the multiplication, addition, or logarithm of two real numbers. In the fuzzy controller, most computational operations were comparing and adding two real numbers.

Sensitivity Analysis

We studied the sensitivity of the fuzzy controller by replacing the FAM rules with "bad" or sabotage FAM rules and by randomly removing FAM rules. Sabotage FAM rules were selected to deliberately confound the system. Figure 10 shows the trajectory when the most important FAM rule — FAM rule 17: the fuzzy controller should produce zero output when the truck is nearly in the correct parking position — is replaced by two extreme bad rules. Figure 11 shows the truck trajectory when four randomly chosen FAM rules (6, 12, 18, and 22) are removed. Such perturbations have little effect on the fuzzy controller's performance.

We studied robustness of each controller by examining failure rates. In the fuzzy control case, fixed percentages of randomly selected FAM rules were removed from the system. In the neural control case, training data was removed. Figure 12 shows performance errors averaged over ten back-ups with missing FAM rules for the fuzzy controller and missing training data for the neural controller. The missing FAM rules and training data ranged from 0 % to 100 % of the total. In Figure 12 (a), the error is Euclidean distance from the actual final position (ϕ, x, y) to the desired final position (ϕ_f, x_f, y_f). In part (b),



(a) (b)

FIGURE 10 The fuzzy truck trajectory when the key FAM rule 17 (ZE) was replaced by the two worst rules: (a) Positive extreme(PB) and (b) Negative extreme(NB).

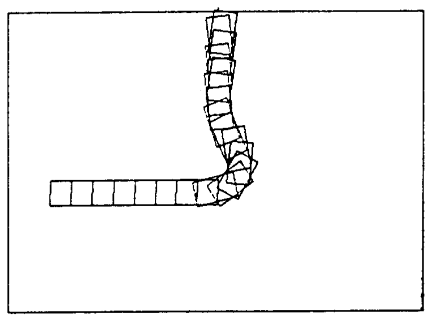


FIGURE 11 Fuzzy truck trajectory when FAM rules 6, 12, 18 and 22 are removed.

the trajectory error is the ratio of the actual trajectory length of the truck divided by the straight line distance to the loading dock:

$$\text{Trajectory error} = \frac{\text{length of truck trajectory}}{\text{distance(initial position, desired final position)}} .$$

Conclusion

Fuzzy control shows optimal truck backing-up performance despite rough definition of fuzzy membership functions and a small number of articulated FAM rules. Once the linguistic FAM rules are specified, the control is efficient and robust. Structured knowledge of the control process is simply and immediately represented as entries in a FAM-bank matrix. The user need not specify a mathematical transfer function or provide a statistically representative set of numerical training data. If such data is provided, various neural learning algorithms can be used to adaptively estimate, and re-estimate, the FAM rules.

The neural truck backer-upper requires extensive training. In large-scale systems, such training may well be computationally prohibitive. Moreover sufficient numerical training data may not be available.

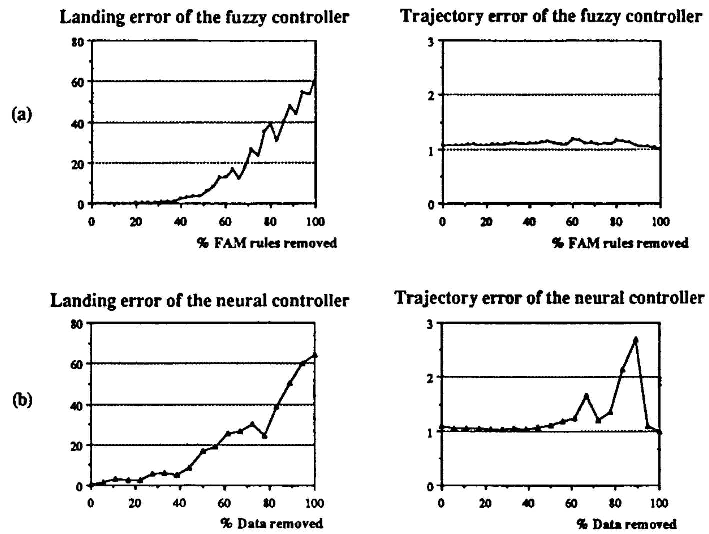


FIGURE 12 Comparison of robustness of the controllers: (a) Landing and Trajectory error of the fuzzy controller, (b) Landing and Trajectory error of the neural controller.

Our neural truck backer-upper, with its comparatively fast failure rate, required hours of training time on a SUN 3 workstation. Training was complicated by the many training samples with numerically close values.

Structured knowledge of a control system seems difficult to directly encode in a neural controller. A neural network simply may not be the appropriate *architecture* for many control problems. A more appropriate architecture may be a fuzzy system or FAM-bank architecture, with embedded neural networks to adaptively estimate individual FAM rules.

Acknowledgement

This research was supported by the Air Force Office of Scientific Research (AFOSR-88-0236) and by a grant from the Rockwell Science Center.

References

Kosko, B., "Fuzzy Entropy and Conditioning," *Information Sciences*, vol.40, 165-174, 1986.

Kosko, B., Neural Networks: A Dynamical Systems Approach to Machine Intelligence, *Prentice-Hall*, 1990.

Li, Y.F. and Lau, C.C., "Development of Fuzzy Algorithms for Servo Systems," *IEEE Control Systems Magazine*, 65-72, April 1989.

Nguyen, D. and Widrow, B., "The Truck Backer-Upper: An Example of Self-Learning in Neural Networks," *Proceedings International Joint Conference on Neural Networks(IJCNN-89)*, vol.2, 357-363, June 1989.