A comparison of neural net and conventional techniques for lighting control

Robert Dodier [*], Diane Lukianow[†], James Ries[‡], and Michael C. Mozer[§]

August 25, 1994

Abstract

We compare two techniques for lighting control in an actual room equipped with seven banks of lights and photoresistors to detect the lighting level at four sensing points. Each bank of lights can be independently set to one of sixteen intensity levels. The task is to determine the device intensity levels that achieve a particular configuration of sensor readings. One technique we explored uses a neural network to approximate the mapping between sensor readings and device intensity levels. The other technique we examined uses a conventional feedback control loop. The neural network approach appears superior both in that it does not require experimentation on the fly (and hence fluctuating light intensity levels during settling, and lengthy settling times) and in that it can deal with complex interactions that conventional control techniques do not handle well. This comparison was performed as part of the "Adaptive House" project, which is described briefly. Further directions for control in the Adaptive House are also discussed.

1 Introduction

1.1 The lighting control problem

The problem of lighting control is more subtle than it appears at first glance. The goal is to set light devices so that their light output matches the vision requirements of an inhabitant of a building. The hardware side of the problem

[*]Dept. of Civil Engineering, U. Colorado at Boulder, Boulder, CO 80309. Please address email to `dodier@cs.colorado.edu`.

[†]Dept. of Electrical Engineering, U. Colorado at Boulder.

[‡]Dept. of Electrical Engineering, U. Colorado at Boulder.

[§]Dept. of Computer Science, U. Colorado at Boulder.

is well-defined. Light devices, incandescent lamps in this study, can be set to various power levels, and their light output varies in a predictable way with the power input; the light output can be reliably measured using photoresistor sensors. Thus, if we assume the problem consists entirely of matching hardware device levels to hardware sensor levels, the solution is at least feasible.

The picture becomes more complicated when we consider a human inhabitant. The human response to light has at least two aspects, one of which is subjective and the other of which is extremely subjective. First, the brightness of the object at which the inhabitant is gazing (technically speaking, the *luminance* of the *visual task*) is not always perceived the same. Rather, the perceived brightness depends on the brightness of the surroundings, the brightness of objects viewed within the last few minutes, and the intrinsic brightness of the object being viewed. The response of the eye to luminance is roughly logarithmic. So at a high luminance level, a change of a certain magnitude is perceived as smaller than the same change at low luminance levels. In particular, a bright daylight scene outside the windows of the building will cause the interior to appear relatively dark, and also a person walking from the outside to the inside on a sunny day will perceive light differently immediately upon entering than after a few minutes within the building. These physiological phenomena make it difficult to compute the performance index of the lighting control system, for the same objective brightness (as measured by light sensors) can be perceived in different ways at different times.

The mental state of the inhabitant is even more elusive than visual physiology. One of the goals of a control system is to meet the desires of the inhabitant, yet these desires are subject to whim and caprice, and are never completely predictable. The best a control system can do is to attempt to detect the gross patterns or regularities in the desires of the inhabitant, as reflected by the device and sensor states that are effected directly by him or her. In a sense the control problem is more difficult in a residence which is chiefly used by only one person than in a large building which is used by many people, because in a large building, use of lights, hot water, cold water, and so on, are averaged over many inhabitants, and only gross patterns of use remain. But in a residence, use patterns may vary greatly from day to day as there is but one or a few inhabitants.

In the design of the lighting control systems discussed in this paper, the vagaries of physiology and mental state are ignored. The desired state is specified by light sensor levels, and the performance index is computed as the sum of the squared differences between the specified levels and the levels that are actually achieved by the control system.

2 The Adaptive House Project

The Adaptive House project is an experiment in adaptive control of an actual, inhabited residence being carried out at the University of Colorado. The residence was recently remodeled, and at the time of remodeling, hardware for sensors and control devices was installed. Several kinds of sensors have been installed within the house, including photoresistor, thermistor, and sound detector sensors installed in sets of three (11 sets have been installed), 10 infrared motion detectors, magnetic door and window closure detectors (11 door units and 2 groups of window units), and a positive-displacement flow meter on the outlet pipe of the hot-water heater. There is also one set of thermistor, photoresistor, and photovoltaic sensors installed on the exterior of the house. The analog sensor outputs (including thermistor, photoresistor, photovoltaic, and sound sensors) are converted from voltages to 8-bit unsigned numbers by an analog-to-digital converter. Thus the minimum and maximum readings of these sensors are 0 and 255, respectively. In the case of the photoresistor sensors, the voltage range is 0 to 10 V, giving a resolution of 39 mV per unit output. The controllable devices include 23 banks of incandescent lights (mostly within the house but also including porch and garage lights), six ceiling fans, the water heater, a stereo system, and an electric resistance heater in one of the bedrooms. The furnace will also be controllable, but it is not now.

Devices in the Adaptive House are controlled by a commercially-available residential control system which follows the X-10 protocol. The X-10 controller transmits signals over the household electrical system to controlled devices. A device can be supplied with 16 power levels, from 0 (no power) to 15 (full power). The power supplied is proportional to the power level. The device levels can be set by hand using wall switches, as well as by using the computer interface. X-10 device levels are issued by a custom microcontroller based on the Intel 8086 chip. An IBM PC-clone collects data from sensors and relays the data to a Unix workstation over a serial line; device levels are specified by applications running on the workstation and are relayed by the PC to the microcontroller. Figure 1 shows the flow of data in the Adaptive House.

The server and its clients on the workstation communicate data and device commands by means of sockets. Specific statistics can be requested by a client, which are then collected on its behalf by the server. The server can also notify clients when events occur, such as a door opening or a device command issued by the inhabitant. The server has the important task of arbitrating clients' requests to send device commands to the PC, and can broadcast updates of the states of devices to all clients. Thus much of the data collection and processing can be localized to the server, making each client a simpler program, and making it possible for the clients to run without worrying about synchronizing device requests with the other clients.

To turn a light device on or off delays the client about 700 ms. Of this time delay, about 275 ms is required to send the X-10 command from the client,

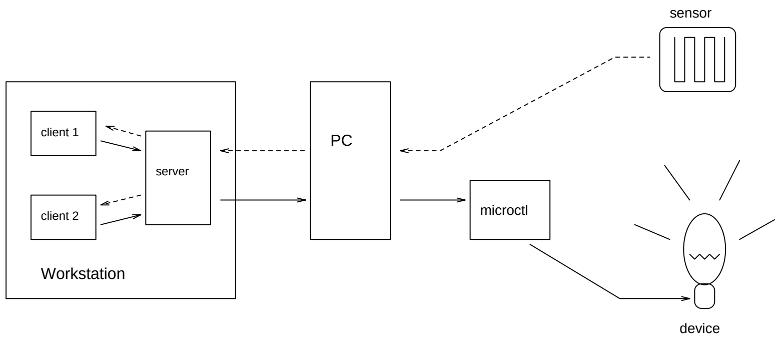

Figure 1: Flow of data in the Adaptive House. Solid lines indicate device commands going from client programs to the X-10 devices, and dashed lines indicate sensor data going from sensors to client programs.

through the server, to the PC, and to return an acknowledgement of the command to the client. The remainder of the delay is spent transmitting the X-10 command to the controlled device. Increasing or decreasing the device level requires a delay of about 200 ms per increment or decrement. Thus increasing the device level from 5 to 10 requires a delay of about $5 \cdot 200$ ms + 275 ms or 1275 ms. Only one device command at a time can be processed by the microcontroller. While the client can continue computing as soon as it has transmitted its device command to the server, the command is only completed when the acknowledgement has been returned.

3 Conventional Feedback Control in the Adaptive House

A feedback control system is a control system that tends to maintain a prescribed relationship of one system variable to another by comparing these variables and using the difference as means of control. Feedback systems, often referred to as closed-loop systems, have increased accuracy over open-loop systems and are capable of reducing the effects of nonlinearities as well as external disturbances.

The primary goal of this project was to demonstrate the effectiveness of a neural network in controlling room-light intensity. The classical control approach was needed to serve as a benchmark against which the performance of the neural network could be evaluated.

The component or process to be controlled can be represented by a block diagram. A block diagram is a pictorial representation of the cause-and-effect relationship between the input and output of a physical system. A single input, single output block diagram representing the various components of the lighting control system used in this project is illustrated in Figure 2.

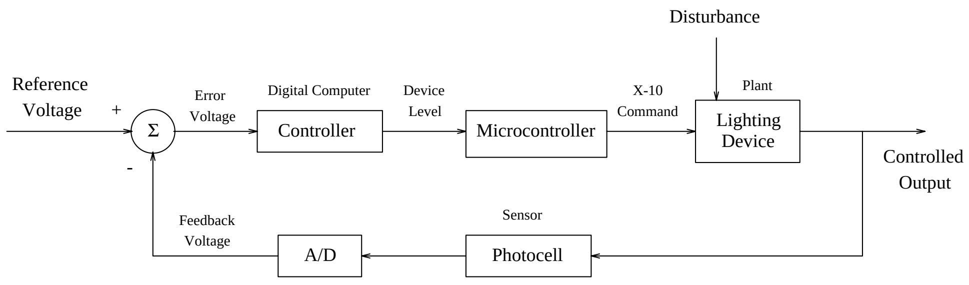

Figure 2: Block diagram of the lighting control system in the Great Room.

In the application of feedback control to the lighting system of the Adaptive House, the following system components are identified. Here the plant is the group of lighting devices operating in the Great Room. The reference input is specified by a text file that is an input to the control program; our objective is to make the plant achieve the reference input. Disturbances, for example, sunlight coming through the windows and random variations in the outputs of the photoresistor sensors, cause the system to drift away from the state predicted by our system model, and must be compensated for. Our control signal is the set of X-10 device levels which we specify to the plant. The photoresistor sensor circuit, which transduces radiation to a voltage signal, is our feedback device. The difference between the reference input and the feedback signal is the error signal, which we try to drive to zero. The controlled output here is the photoresistor sensor output. As noted above, this is observable, but it is only a stand-in for the unknowable physiological visual response of the inhabitant.

3.1 Design of the Conventional Controller

This experiment was limited to a single room in the Adaptive House referred to as the Great Room. A layout of this room is shown in Figure 3. This room contains four photoresistors, three banks of wall lights, four banks of ceiling lights. The ceiling is high and there are several windows.

To simplify the controller design, the Great Room is divided into four zones, namely, the South, West, North, and Kitchen zones. Each zone is treated as a single input, single output control loop. That is, the cross-coupling from one zone to the other is not taken into consideration when developing the system model. Interactions between zones do occur, because light from devices in one zone spills over into adjacent zones and can be detected by the sensors there. The coupling is strongest between adjacent zones; the North and South zones are only weakly coupled.

The location and type of system components had been selected and fixed before design of the conventional controller began. Properties of the inexpensive photoresistors chosen for the sensors contribute to system inaccuracies. The

Figure 3: Floor plan of the Great Room. There are other rooms to the southest, east, and northeast.

photoresistors have an exponential characteristic curve with a limited linear region. They are directionally sensitive, sustain some amount of hysteresis, and there is substantial variation in response from photoresistor to photoresistor. Mounting locations were not chosen for optimal performance, making light intensities and incident angles inconsistent from sensor to sensor. In particular, the Kitchen sensor is mounted on the wall below cabinets which partially shade the sensor from the light devices in the Great Room. As a result, the output range of the Kitchen sensor is only about half that of the other sensors.

Because the overall design called for a controller which accepted a desired light sensor level as the reference input and output a corresponding X-10 level, our first task was to determine a mapping function that would convert light sensor levels to X-10 levels. We knew the response would be inconsistent from one sensor to the other because of differences in mounting orientation and response. Nevertheless, we felt we could obtain a curve that would provide a reasonable model on which the controller could be based.

We chose to use the South sensor on which to base our model since it received the widest range of light. All light devices in the Great Room were set to a particular X-10 level, and then the sensor reading was recorded. This was repeated for each of the 16 X-10 levels.

In order to turn the data into a function which could be used in the controller, we applied a third order least squares curve fit. The regression curve did not fit the endpoints well, so the curve was adjusted by hand to compensate for the inaccuracy. It is found that the regression curve has a strong linear component. This is somewhat surprising, considering the roughly logarithmic response of the photoresistor sensors. Probably the incandescent lamps have an output which increases faster than linear in the range of wavelengths shorter than the threshold wavelength required to generate electron-hole pairs in the photoresistor. (It is known that the output of incandescent lamps in the range of visible wavelengths increases faster than linear with the input power, as the peak of the black-body curve moves from the infrared toward shorter wavelengths.) Thus, the nonlinearities of the photoresistors and the incandescent lamps appear to largely cancel each other. Figure 4 shows the relation between device level and sensor level which is assumed in the system model used by the conventional controller.

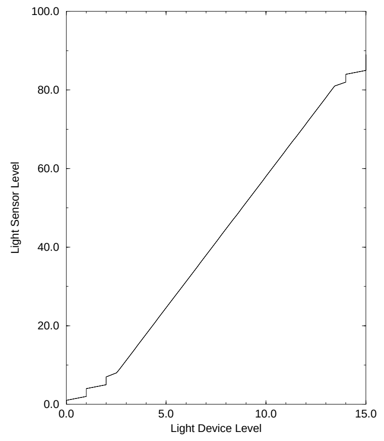

Figure 4: Photoresistor sensor output as a function of X-10 light device input. The relation is closely approximated by a cubic polynomial, except at the lowest and highest X-10 levels.

Due to cross-coupling between the zones and to variations in sensor responses, and, to a lesser extent, to nonlinearity, the conventional controller needs to be error-driven and loosely tied to the system model given by the regression curve. There are essentially no dynamics in the system. The general scenario is to set the light devices to a given X-10 level, check the response from the light sensors, and adjust the light devices accordingly. We are then left with a first order system.

We have divided the Great Room into four zones and treat each zone as a single input, single output system. Each zone has an overhead light and a pair of wall lights, except for the kitchen, which has only overhead lights. Each zone is treated as a separate system with its own output and sensor. Interactions or cross-coupling between the zones were left to be handled by the individual controllers and were not built into the model. To maintain a single-input, single-output design, the wall and overhead lights in each zone are kept at the same

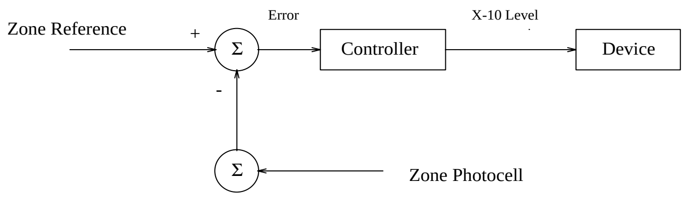

Figure 5: Block diagram of the single input, single output system assumed in each zone.

device level. See Figure 5 for a schematic of the single-zone system. Note that this limits the flexibility of the possible light device levels; the wall and overhead devices can actually be controlled separately.

3.2 The Conventional Control Algorithm

The control algorithm is then as follows. In each zone, the X-10 device level needed to meet the sensor setpoint is predicted by the regression model. To compensate somewhat for the difference between actual sensor response and the response predicted by the regression model, we allow the regression curve to translate along the X-10 device level axis; the translation is determined by the defect between predicted and measured sensor levels. It is usually necessary to repeat the measurement – adjustment cycle several times to achieve the setpoint. The control program gives up after 15 cycles even if not all setpoints have been achieved.

Here is pseudocode which describes the control algorithm. This algorithm is repeated for each set of four setpoints.

```
for each zone, turn all light devices off
get setpoints for light sensors (from a file)

while ( some zone has not yet achieved its setpoint )
{
  for each zone
  {
    map new setpoint levels to predicted X-10 levels
    get current sensor levels
    map current sensor levels to predicted X-10 level
    if ( abs( new setpoint levels - current sensor levels) > threshold )
    {
      difference = predicted target X-10 level
                     - predicted current X-10 level
      current X-10 level += difference
    }
  }
```

}

4 A neural network for lighting control

The neural network is one way to realize a many-input, many-output mapping of the kind needed in a control scheme. The relevant properties of the neural networks include smoothness (since the output is the result of the composition of smooth functions), and universal approximation capability. It has been shown [2, 6] that a net with one hidden layer can approximate any continuous mapping if the hidden layer has enough units, although in practice it is difficult to tell how many hidden units is "enough."

In the problem at hand, we need a *forward model* which predicts how light device levels (inputs) map to light sensor levels (outputs). This model is a neural network which implements a function $\hat{y}$ which takes as inputs,

- The *current* light device levels (7 inputs),
- The *current* light sensor levels (4 inputs), and
- The *new* light device levels (7 inputs),

and maps them to the outputs,

- The *new* light sensor levels.

We chose to include the current device/sensor state because this will give information about the ambient light levels, which is not explicitly available. In all, our forward model contains 18 inputs, 18 hidden units, and 4 output units. In addition to input-hidden and hidden-output connections, there are direct connections from the inputs to the outputs. The hidden units have hyperbolic tangent activations, and the activations of the outputs are linear. The effect of the direct connections from the inputs to the linear output units is to superpose a linear network on the nonlinear network containing the hidden units.

This superposed network can learn the easy (linear) part of the input–output mapping. As can be seen in Figure 4, the relation between device and sensor level is only mildly nonlinear.

Training data for the network was generated by setting random light device levels and then recording the resulting light sensor levels. The training data was recorded in two sequences, during the night and early morning of two consecutive days. There were, in all, 5850 training patterns. The network was trained to minimize the sum of squared error between the target sensor levels and the network's output[1]. Each input or output variable was translated by its mean over the training set and scaled by its standard deviation. Thus each transformed variable has mean zero and unit standard deviation. Altogether, 2925 patterns were selected at random for training the network, and the remaining 2925 patterns were used for validation, as described by Finnoff et al. [5]; it has been shown by Plutowski et al. [7] that validation error is an estimate of generalization error. The forward-model network was trained for about 3000 epochs. During this time the validation error continued to decrease, suggesting the the net is not over-trained. Further training (for about 10000 more epochs) also showed no overfitting, as the validation error continued to decrease. Three more networks were generated and trained using different selections of training and validation data, and none showed overfitting for as long as training was continued (about 3000 epochs each). This shows that lack of overfitting as not due to chance selection of the training and validation sets, and further suggests that the amount of data available for training and validation, 5850 data, is large enough to give low-variance weight estimates.

The trained network used to implement the network-based control scheme achieved a normalized mean squared error ($NMSE$) of 0.0152 on the training set, and the validation set $NMSE$ was 0.0157. (The $NMSE$ is discussed in more detail in Section 5.1.) A purely linear network (no hidden units) achieves a $NMSE$ on the training set of 0.0578, about 4 times as great as that of the linear + nonlinear network used in the control system. Thus while Figure 4 shows a strong linear trend in the device level to sensor level mapping, a nonlinear network can still learn substantial nonlinear structure in the data.

The initial weight distribution of the network is shown on the left side of Figure 6. Initial weights were picked from a uniform distribution such that the absolute values of the weights on the fan-in of a unit summed to 2. It is found that the distribution of weight magnitudes in the trained network is roughly gaussian, with a peak near zero and a standard deviation of about 0.25, as shown on the right side of Figure 6. It is of interest that the weight decay scheme discussed by Rumelhart et al. [9] assumes that weights should be distributed in just this manner; the decay parameter λ is related to the variance of the weight distribution, with $\lambda = 1/\sigma^2$, where σ^2 is the variance of the prior distribution of weights. While the number of very small weights did not change much during training, some of the medium-sized weights became larger, increasing the spread of weights around zero and increasing the nonlinearity of the network. For another view of weight distributions, see Weigend et al. [10].

[1]It is not known what performance index is most meaningful for training, but there is some justification for a squared-error performance index. The eye compensates automatically for small to moderate changes in brightness, and so giving small deviations from the target less weight than large ones seems appropriate.

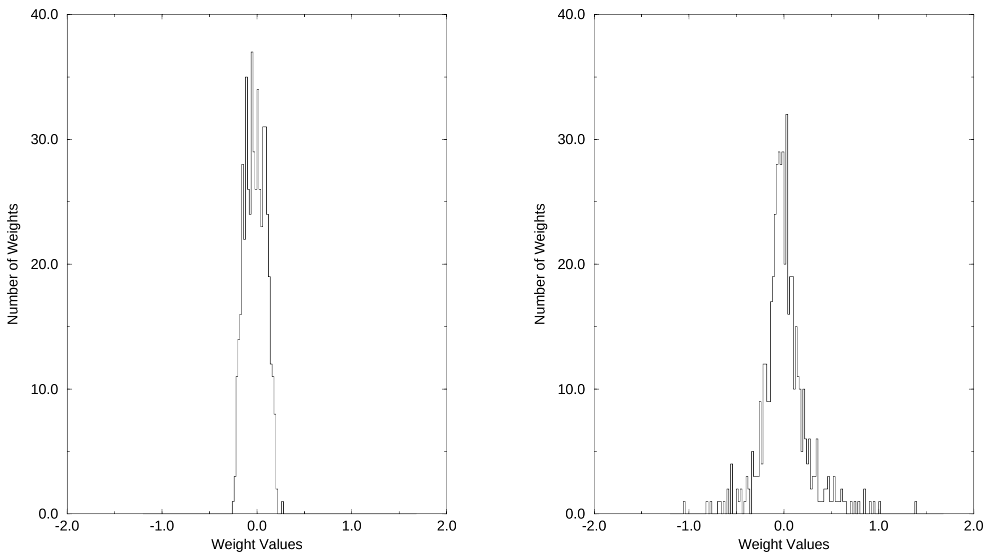

Figure 6: Development of weights over training. The histogram on the left shows the distribution of weights and biases before training, while the one on the right shows the distribution of weights and biases in the trained network. In all, there are 490 adjustable weights and biases in the network.

4.1 Implementation of neural network code

Since in the lighting control task, the neural network is but a part of the whole code to read the sensors and issue device commands, we chose to implement the neural network using stand-alone code instead of using a simulator package. The network is a C++ object, as is the sensor/device interface. The forward model class is derived from a standard back-propagation network, with added functions for carrying out the output error minimization with respect to the device levels. (This minimization is described in more detail in the next subsection.) The separation of the back-prop class from the forward model class means that existing back-prop code could be used, without compromising its integrity, in the experimental forward model class.

4.2 Determining controls with the model

The key idea behind the neural network light control is differentiating the network model [1], a technique also referred to as backpropagating through the model. Since the network maps device level inputs to sensor level outputs, it is necessary to adjust the inputs to achieve the desired outputs. Consider the output error as a function of the inputs, with the weights frozen,

$$||\hat{y}(x) - y||^2 \tag{1}$$

Here $x = (x_1, x_2, \ldots x_7)$ is the vector of input device levels, and $y = (y_1, \ldots y_4)$ is the vector of sensor setpoints. The neural network implements the function $\hat{y} = (\hat{y}_1, \ldots \hat{y}_4)$ to map input device levels to output sensor levels; the current device levels and current sensor levels are held fixed. Explicitly,

$$\hat{y}(x) = \hat{y}(x; \hat{y}', x') \tag{2}$$

where $\hat{y}'$ and x' are the current sensors levels and current device levels; the prime mark $'$ distinguishes them from the new sensor levels $\hat{y}$ (the setpoints) and the new device levels x.

Since the network is a composition of smooth functions, the output $\hat{y}$ is a smooth function of the input x, and to minimize Eq. 1 we can differentiate with respect to the device levels and apply a gradient-based optimization method. Simple gradient descent will work, but to speed convergence we employ the conjugate gradient method as implemented by Stephen Sullivan and available as a public-domain code. Sullivan's code uses the golden section method in the line search. The solutions found by minimizing Eq. 1 give non-integer values for the optimal device levels. Non-integer values are rounded off to the nearest integer, and values below zero or above 15 are set to zero or 15, respectively. It can be argued that it would be better to incorporate such constraints directly into the search for solutions, rather than applying them post-hoc.

The technique of differentiating the model is not specific to neural networks; it can be applied to any smooth model. The result of backpropagating through the model is very much the same as if we had repeatedly applied inputs to the actual system, observed the output errors, and adjusted the inputs to decrease the error. However, in the case of differentiating the model, the feedback loop is entirely computational. Since computations are much faster than physical device changes in the lighting system, this means the neural network control can be much faster than the conventional feedback control.

5 Comparison of control techniques

The performance of the conventional and neural network control techniques were studied experimentally at the Adaptive House. The control problem is to set the light device levels in the Great Room so as to achieve specified light sensor setpoint levels. Three sets of setpoint 4-tuples were constructed, containing 32, 30, and 300 tuples, respectively. The programs implementing conventional and neural control were executed during the night. For each setpoint, initial light device levels were set, and the control program had to determine the light device levels that would produce the specified sensor levels. Performance can be judged from the sum of squared sensor errors, where the error is defined to be the difference between the setpoint and the sensor level which actually occurred, and the sum is taken over the four light sensors in the Great Room.

Another performance index that we measured is the time needed for the control program to set the device levels. However, the most interesting performance index would compare the light levels as they are perceived against the desires of the inhabitant, and the sum of squared sensor errors is but a poor stand-in for this subjective error. Perhaps there is some way to measure the pleasure or displeasure of the inhabitant; then the control problem becomes a reinforcement learning problem instead of a supervised learning problem, in which correct targets are explicitly available.

5.1 Experimental results

The results of runnning the conventional and neural network control programs are summarized in Table 1. There are three test sets. Set 1 consists of 32 hand-chosen setpoint tuples, for example, (0,200,0,0) and (50,50,50,50). Set 2 consists of 30 setpoint tuples which were recorded from device levels set by the inhabitant (Mozer). In this respect this test set is perhaps the most realistic of the three. Set 3 consists of 300 setpoint tuples which were recorded from device levels generated by random device levels, in the same way the training data was generated. For each set the following quantities are reported; all are per-tuple average values. Column 3 records the average number of iterations of the control algorithm. In the conventional control this represents a measurement – adjustment cycle, and in the neural network control this represents a cycle of the conjugate gradient algorithm, the line search being the most time-consuming part. Column 4 records the average time elapsed to achieve (or attempt to achieve) the setpoint levels. This includes time for processing and time for issuing device commands; the latter dominates the elapsed time. Columns 5 through 8 record the average error on each setpoint. That is,

$$e_l = \frac{1}{N}\sum_{k=1}^{N} e_{kl} = \frac{1}{N}\sum_{k=1}^{N}(y_{kl} - \hat{y}_{kl}), \qquad l = 1, ...4. \tag{3}$$

The mean square error (not shown in Table 1) is defined as

$$MSE = \frac{1}{N}\sum_{k=1}^{N}\sum_{l=1}^{4} e_{kl}^2 = \frac{1}{N}\sum_{k=1}^{N} ||y - \hat{y}||^2, \tag{4}$$

and column 10 records the *normalized mean square error,*

$$NMSE = \frac{\frac{1}{N}\sum_{k=1}^{N} ||y - \hat{y}||^2}{\frac{1}{N}\sum_{k=1}^{N} ||y - \bar{y}||^2}. \tag{5}$$

Each setpoint tuple $y = (y_1, y_2, y_3, y_4)$ is compared to the actual sensor level that was achieved, $\hat{y} = (\hat{y}_1, \hat{y}_2, \hat{y}_3, \hat{y}_4)$. The mean of the targets over the training data,

Table 1: Summary of Test Results. Cols. 3 – 10 report average values.

		iter.	time, s	e_1	e_2	e_3	e_4	$NMSE$
Set 1,	conv.	3.66	23.10	11.70	13.30	15.30	0.91	0.0724
N=32	n.n.	13.30	5.05	4.94	3.37	5.56	-4.22	0.0587
Set 2,	conv.	4.47	53.70	10.10	0.60	5.80	-4.73	0.0320
N=30	n.n.	19.60	9.18	-8.70	-16.70	-15.10	-12.70	0.0530
Set 3,	conv.	4.37	33.90	-2.85	-5.09	0.95	0.80	0.0777
N=300	n.n.	19.30	12.00	-5.42	-11.10	-7.33	-3.79	0.0913

Table 2: Statistical Tests on Ratios of MSE

	MSE, conventional	MSE, network	ratio, max/min	d.f.	significance (two-tailed)
Set 1	4080.0	3310.0	1.230	128	0.24
Set 2	878.0	1450.0	1.652	120	0.0062
Set 3	766.0	899.0	1.174	1200	0.0054

$\bar{y} = (\bar{y}_1, \bar{y}_2, \bar{y}_3, \bar{y}_4)$, is intended as a "zero'th order" approximation to appropriate light sensor levels; the mean corresponds to a policy of always turning on the lights to about half intensity, no matter what specific setpoint is desired.

Other meaningful normalizing factors can be defined. For example, we might wish to compare the error of a sophisticated control scheme against the simple feedback control described in this paper. In this case, the normalizing factor would be the mean square error of the feedback control, and $NMSE$ describes the reduction of error attributable to using the sophisticated scheme.

Note that a MSE of 1000 means that the squared-error on each sensor is about 250, that is, the error on each sensor is about 16. This error is about $16/256 = 6\,\%$ of the full 0 to 255 range of the photoresistor sensors. It is not known whether errors of this magnitude are undetectable, noticeable, or bothersome to the inhabitant.

To judge the relative accuracy of the conventional and neural network control schemes, a simple statistical test was made on the MSE of each method. It is assumed that sensor setpoint errors $e_{kl}, k = 1, ..., N, l = 1, ..., 4$, are independent, identically distributed gaussian random variables with mean zero and variance σ^2. Then the squared errors e_{kl}^2/σ^2 are independent, identically distributed chi-square random variables with 1 degree of freedom (d.f.), and

$$\frac{N}{\sigma^2} MSE = \frac{N}{\sigma^2}\frac{1}{N}\sum_{k=1}^{N}\sum_{l=1}^{4} e_{kl}^2 = \sum_{k=1}^{N}\sum_{l=1}^{4}\frac{e_{kl}^2}{\sigma^2} \qquad (6)$$

is chi-square distributed with $4N$ d.f. The ratio $f = X_1/X_2$ of two independent chi-square variables X_1 and X_2 has an F distribution, and the significance of the ratio can be computed using the incomplete beta function I_α, as described by Press et al. [8]. Assuming that $f > 1$,

$$\Pr(F \geq f) + \Pr\left(F \leq \frac{1}{f}\right) = 2I_\alpha(\frac{\nu_2}{2}, \frac{\nu_1}{2}), \qquad \alpha = \frac{1}{1 + \frac{\nu_1}{\nu_2}f} \tag{7}$$

Here ν_1 and ν_2 are the d.f. of X_1 and X_2, respectively. It is assumed that $X_1/X_2 > 1$; if the test is two-tailed, we can simply take X_1 as the larger of the two variables and X_2 as the lesser. In the case at hand, we take X_1 as the greater of the MSE of the conventional controller and the MSE of the neural network controller, and we take X_2 as the lesser, and we have $\nu_1 = \nu_2 = 4N$. Table 2 shows the significance for each of the three test sets. If we set our critical significance level at 5%, for Set 1 we accept the null hypothesis that the MSE are equal, and reject it for Sets 2 and 3.

These results show that for Set 1, the MSE of the neural network controller is not significantly different at the 5% level from that of the conventional controller. However, the differences in MSE for Sets 2 and 3 are indeed significant at the 5% level. Note that while the difference in MSE is significant for Set 3 (by far the largest test set), it is not substantial; the difference is less than 10 %.

It should be kept in mind that the requirements for the F-test are only approximately satisfied. The errors are biased somewhat away from zero, as shown by the columns $e_1, ..., e_4$ in Table 1. Also, the errors on sensor 4 were found to have a smaller variance than those on the other sensors. It is not known to what extent the errors are dependent.

6 Conclusion

6.1 Comparison of Control Schemes

The conventional and neural network control schemes we tested show comparable performance in achieving desired setpoints. Both controllers do an acceptable job of achieving setpoints, with $NMSE$ in the range of 5 to 10 %. The conventional controller shows somewhat better performance in this respect, achieving significantly lower errors on two of the three test sets. However, the time required to achieve a setpoint is much greater for the conventional control than for the neural network control. This is due to the lack of an accurate system model in the conventional controller; the inaccurate model makes necessary several measurement – adjustment cycles. On the other hand, the neural network controller can achieve comparable performance because of its accurate system model, which can account for cross-coupling between zones and changes in ambient light, despite its lack of feedback from the real-world controlled system. Neural networks are nonlinear models of great flexibility, and as shown in this paper a neural network can accurately model the relation between device and sensor levels. Such accuracy is important to the neural network controller because it has no feedback cycle, and the first attempt at the setpoints is the only attempt.

The time elapsed between the transmission of a device command and the completion of the command is often on the order of 1 to 2 seconds, and commands can only be executed serially. For both controllers, much more time is spent waiting for the execution of device commands than is spent computing, so reducing the number of device commands greatly reduces the time needed to achieve setpoints. As Table 1 shows, the conventional controller required from 2.5 to 4 times as long to settle as did the neural network controller, and the accuracy of the conventional controller is only marginally superior. The neural network controller is superior to the conventional control in that light device levels are set only one time. In contrast, the feedback controller changes light device levels repeatedly before settling; the prolonged fluctuations of the light levels are annoying to the inhabitant.

The speed of the neural network control could be increased somewhat by using a more efficient code for minimization of Eq. 1. The code now used implements the conjugate gradient method using the golden-section method for the line search, and requires about 20 to 30 forward passes through the network per line search. A derivative-based search could reduce the number of forward passes to perhaps 10 to 15.

6.2 Future Research

While sensor setpoint errors are conveniently objective for analysis, it would be more meaningful to assess performance directly in terms of the inhabitant's satisfaction. This can be measured by the inhabitant's own light control actions. If the inhabitant uses the wall switches, that indicates dissatifaction with the existing lighting; if the inhabitant must greatly change the lighting, that indicates great dissatisfaction. Thus to some extent dissatisfaction can be measured by the magnitude of the changes which the inhabitant makes directly. We believe that a two-level control system may be appropriate here, a so-called "feudal" control scheme, as proposed by Dayan and Hinton [3]. In the Adaptive House, the higher level would anticipate the desires of the inhabitant, and would specify appropriate setpoints to the lower level. The lower level would consist of the neural network control system described in this paper, and would try to achieve the specified setpoints. In this scheme, the higher level is a reinforcement learner which is rewarded in proportion to the assessed satisfaction of the inhabitant. On the other hand, the lower level is close to the hardware, and it is trained using supervised learning to achieve hardware setpoint levels. By using a two-level control system, we hope to be able to translate the satisfaction of the inhabitant into a signal which can be used to control the lighting hardware.

Acknowledgements

The authors wish to thank Marc Anderson for his programming expertise; Marc implemented much of the code for data collection and the serial link. Thanks also to Gregor Henze for his help in preparing the manuscript. This research was supported by NSF Presidential Young Investigator award IRI–9058450 and grant 90–21 from the James S. McDonnell Foundation.

References

[1] Barto, A. "Connectionist Learning for Control," in *Neural Networks for Control,* T. Miller, R. Sutton, P. Werbos, eds. Cambridge, MA, MIT Press, 1990.

[2] Cybenko, G. "Approximation by Superpositions of a Sigmoidal Function." *Mathematics of Control, Signals, and Systems,* vol. 2 (1989), pp 303-314.

[3] Dayan, P., and G. Hinton. "Feudal Reinforcement Learning," in *Advances in Neural Information Processing Systems 5,* S. Hanson, J. Cowan, C. Giles, eds. San Mateo, CA, Morgan Kaufmann Publishers, 1993. pp 271-278.

[4] Dodier, R. "Increase of Apparent Complexity is Due to Decrease of Training Set Error," *Proceedings of the 1993 Connectionist Models Summer School,* Hillsdale, NJ, Lawrence Erlbaum Associates, 1993. pp 343-350.

[5] Finnoff, W., F. Hergert, and H. Zimmerman. "Improving Model Selection by Nonconvergent Methods." *Neural Networks,* vol 6 no 6 (1993), p 771.

[6] Hornik, K., M. Stinchcombe, and H. White. "Multilayer Feedforward Networks are Universal Approximators." *Neural Networks,* vol. 2 (1989), pp 359-366.

[7] Plutowski, M., S. Sakata, H. White. "Cross-Validation Estimates IMSE," in *Advances in Neural Information Processing Systems 6,* J. Cowan, G. Tesauro, J. Alspector, eds. San Francisco, CA, Morgan Kaufmann Publishers, 1994. pp 391-398.

[8] Press, W., B. Flannery, S. Teukolsky, W. Vetterling. *Numerical Recipes in C.* Cambridge, Cambridge University Press, 1988.

[9] Rumelhart, D., R. Durbin, R. Golden, and Y. Chauvin. "Backpropagation: The Basic Theory," to appear in *Mathematical Perspectives on Neural Networks,* P. Smolensky, M. Mozer, and D. Rumelhart, eds.

[10] Weigend, A., B. Huberman, D. Rumelhart. "Predicting Sunspots and Exchange Rates with Connectionist Networks," in *Nonlinear Modeling and Forecasting,* M. Casdagli and S. Eubank, eds. Santa Fe Institute Studies in the Sciences of Complexity Proc. Vol XII. Reading, MA, Addison-Wesley, 1992. pp 395-432.