Leading Indicators of CAD Experience

Suresh K. Bhavnani
James H. Garrett, Jr.

Carnegie Mellon University
Pittsburgh, PA 15213 USA

Doris S. Shaw

Construction Engineering Research Laboratory (CERL)
U.S. Army Corps of Engineers
Champaign, IL 61826 USA

Current interfaces and help facilities of CAD systems are not designed to respond to a user's performance. To develop such adaptive environments, a better understanding of the indicators of CAD experience is required. This paper discusses the results of studying behavior patterns of different types and levels of CAD users for a specific drawing task. The results show that the type and experience of the CAD user has a clear correlation to the pattern of commands used, the time taken, and the quality of drawing produced. By using the experimental data to train a neural network, the paper demonstrates a connectionist approach for experience assessment. This information, it is proposed, can provide input to an adaptive interface which generates unobtrusive interception to improve the performance of a CAD user. Future experiments to explore the issues of generality and interception are presented.

Keywords: CAD user modeling, adaptive interface, neural networks.

1 Introduction

In recent years, the number and complexity of features in interactive computer systems have rapidly increased. Intergraph's mechanical design CAD package (I/EMS or Intergraph Engineering Modeling System), for example, currently supports almost one thousand commands. CAD developers and engineering management believed that these improvements in design tools would improve performance, quality, and delivery time throughout the design workflow (Majchrzak, 1990). Unfortunately, the rapid implementation of the CAD technology has occurred at a pace well ahead of understanding how humans interact with them (Cuomo and Sharit, 1989). Recently, laboratory as well as field experiments have revealed the dismal performance of CAD users. Bietz et al. (1990) found mechanical engineering students who had passed a CAD course produce better quality and more complete drawings at a lower effort using paper and pencil than on a CAD system. Luczak et al. (1991) studied 43 subjects using 11 CAD systems in 11 factories. They found

U. Flemming and S. Van Wyk (eds.), ***CAAD Futures '93****, pp. 313-334.*
© 1993 Elsevier Science Publishers B.V. Printed in the Netherlands.

that even when the subjects were highly trained, the high complexity of the commands (due to many input parameters, restrictions, and requirements) led to low performance, reduced creativity, frictions, and frustrations. Finally, Majchrzak (1990) found no improvement in the performance of 25 engineers and 60 drafters using CAD systems compared to non-CAD users.

Aware of these problems, CAD developers have struggled to improve the performance of their customers through training, documentation, on-line help, and phone support. Intergraph Corporation, for example, offers three weeks of training for the new user of I/EMS. Besides being expensive, this training usually results in low retention due to information overload. Even after the intensive training, the phone-support is usually inundated with numerous requests. Additionally, the on-line help is used mainly by proficient users who require more detail about familiar commands; the novices, on the other hand, lack CAD concepts and therefore don't know which command-help to access (see Draper, (1984) for help usage on UNIX). Finally, commercial CAI tools provide canned examples that are not helpful for far-transfer tasks (tasks significantly different from the instructional examples) that the users face in their work environments.

While there is room for improvement in all the areas mentioned above, there is a growing awareness that most computer interfaces are designed for an average unchanging user (Norcio and Stanley, 1989). The plight of a computer user attempting to adapt to a fixed interface can be alleviated by providing an adaptive interface that responds to the specific needs of any user (Greenberg and Witten, 1985, Tyler and Treu, 1989).

This paper presents the argument that CAD systems, being complex and producing low human performance, can be improved by having adaptive interfaces. We begin with a brief discussion of several studies showing user differences on various computer applications and the need for similar systematic studies in the CAD arena. We then discuss various types of knowledge needed for adaptation and focus on user modeling as the main area of our current investigation. The results of a pilot study are presented and compared to results in the literature. We then present a brief overview of various computer representations used to model users and follow with a pilot experiment using neural networks. We conclude by presenting our future direction in the investigation and development of an adaptive interface for a CAD system.

2 Users are Individuals

Because users are individuals belonging to a heterogeneous community, a single model of a canonical user is inadequate in the design of interactive computer systems (Rich, 1983). Recent cognitive studies have revealed more precisely the nature of differences between computer users. The differences reported are along two dimensions: skill level and cognitive abilities.

Vaubel and Gettys (1990) observed command usage differences between experts and novices in a word-processing task. Compared to novices, experts used more power commands if they minimized the number of keystrokes needed to complete the command. A novice, for example, used a long series of "line down" commands to move around the document whereas the expert used "page down" to achieve the same result. Additionally, because the help was rewritten to be comprehensible to novices, they found command help requests decline with the increase of expertise. They recommend that such results could be used in the design of adaptive interfaces. The help and power command usage could determine the general expertise of a user, which in turn could be used to pick the comprehensibility level of the help text. Other factors such as command syntax errors could be used

to determine a user's expertise with a command, which would be useful to determine the content of help on that command.

Doane et al. (1990) also found differences between novice, intermediate, and expert users of the UNIX operating system. Even though the novices and intermediates were enrolled in introductory and upper level computer science classes, they had difficulty in executing tasks requiring iteration. For example, they used the commands "rm Y1" and "rm Y2" to separately delete files Y1 and Y2 instead of using the single iterative command "rm Y*" to remove both files. They also experienced difficulties in executing command compositions such as the use of redirection as in the command "ls > Y1" (used to put the contents of the current directory into the file Y1). The study also found that expertise in the use of a command was dependent on frequency of usage and could increase or decrease over time even for the so-called experts.

Sein and Bostrom (1989) observed cognitive trait and learning mode differences among users of an electronic mail system. They found that users with high visualizing ability outperformed users with low visualizing ability on near and far-transfer tasks; abstract learners performed better than concrete learners. They emphasized the need to tailor instructional aids to suit these individual differences. For example, users with low visualizing ability and concrete learners would benefit most by instructional aids using analogical models, whereas users with high visualizing ability and abstract learners would benefit most by abstract models.

While there have been many systematic studies on skill level and cognitive differences on various computer applications, there has been relatively little done in the area of interactive CAD systems. Lang et al. (1991) found differences in the way trained novices, a design expert, and a system expert used a CAD system. Their results show that the different categories of subjects differed in the type of knowledge they used based on the ACT* theory (Anderson, 1983). Although all the subjects had an equal amount of declarative knowledge (knowledge about commands), the novices performed poorly because of a lack of procedural knowledge (knowledge of how to apply the commands to achieve a goal). Additionally, the design expert, (who had experience on another CAD system and minimal training on the one being tested) could transfer his procedural knowledge to complete the task. The study emphasized the need to provide procedural and planning knowledge to improve the performance of novices and the need to provide command specific knowledge to users transferring from other CAD systems.

The above studies show a clear need for computer interfaces that respond to a varied and changing external environment. Adaptive interfaces, as discussed below, provide a way to respond to such an environment.

3 Adapting to a User

A self-adaptive interface monitors the user's activity and, based on a collection of good guesses, attempts to adapt itself to that user (Rich, 1983). This collection of good guesses is what makes up the system's internal model of the user's world. There are a number of arguments against this approach. For example, instead of guessing what the user needs, why not ask the user what is needed or better still why not provide tools that allow the user to modify the interface? Although this might appear simple, it has some inherent difficulties. First, the users must know what they do not know; so if the user is unaware of the existence of a concept, how can it be requested? Studies show that users are unable to accurately report their abilities (Shaw et al., 1991). Second, the user is not static, and it is not possible to have a constant dialog on the appropriateness of a particular environment

(Vaubel and Gettys, 1990). Third, modifying or tailoring an en-vironment presents yet another system for a novice to master (Greenberg and Witten, 1985). This defeats the purpose of making systems easier to use. The approach we intend to follow is to develop empirical cognitive models of CAD users in well-defined areas and then iteratively refine them based on user performance. The goal is to carefully design a system that adapts or makes suggestions in an unobtrusive way without making the user feel out of control. To achieve this goal, it is important to understand in more detail, the types of knowledge that have been used to model the user's world as discussed below.

The tremendous interest in adaptive interfaces has produced research in areas that include levels of adaptation (Totterdell et al., 1987), adaptive architectures (Norcio 1989), implementations (Tyler 1986; Krogsaeter et al., 1992) and their performance (Greenberg and Witten, 1985; Benyon and Murray, 1988), representations (discussed later), and tools to build adaptive interfaces (Andes and Rouse, 1990). Here we concentrate on the essential components of an adaptive interface.

An adaptive interface requires knowledge of at least four domains (Norcio and Stanley, 1989):

- Knowledge of the user
- Knowledge of the application
- Knowledge of the interaction
- Knowledge of the system

Each of these four domains has been discussed in detail in the literature (Norcio and Stanley, 1989, Norcio, 1991). Here we will attempt to capture their differences by examples taken from empirical results or implemented adaptive systems.

3.1 Knowledge of the User

As we have discussed earlier, a specific user differs from other users and that individual's abilities may change while interacting with the system. In an adaptive interface, the system must dynamically construct a model of the user's expertise and cognitive abilities based on the interactions. Based on the results of Vaubel and Gettys (1990), a rule could state:

if the user has used command *x* in a power mode and
the number of errors on the command are less than *n*
then the expertise on command *x* is *m*.

This rule could provide the conditions for another rule that could be:

if the command expertise on command *x* is greater than *n* and
the command *x* help is requested
then display only the syntax for command *x*.

The first rule determines the expertise level of a user on a particular command based on previous interactions. The second rule uses the expertise level of the user to select the appropriate level of help.

3.2 Knowledge of the Application

This knowledge models the user's attempt to achieve a goal. In order for the system to help the user achieve goals, the system must be able to predict what the user is attempt-

ing to achieve. Based on the results of Greenberg and Witten's (1985) adaptive implementation, a rule could state:

if command *x* is used more than *n* times and
the level of command *x* in the menu hierarchy is greater than three
then suggest command *x* to be in a top-level personal menu.

This rule has nothing to do with the expertise of the user but rather models how the system is being used. The rule makes frequently used commands that are deep in a command hierarchy more accessible at a higher level. The following rule could help users deal with command restrictions specific to an implementation.

if the command 'place hole' is activated and
the hole is being placed on a b-spline surface
then suggest using the Boolean operations command and
provide an example.

This rule intercepts the incorrect use of a command (for a specific CAD system) and advises the user to use a more primitive command to achieve the same goal.

3.3 Knowledge of the Interaction

An adaptive interface must keep track of the past interactions of the user in order to respond appropriately. Based on the results of Krogsaeter's (1992) adaptive layer for Excel (a Macintosh spreadsheet package) a rule could state:

if the user has customized a sequence of commands into a single command and
the user has used the sequence of commands independently
then remind the user of the existence of the customized single command.

This rule flags the user of an earlier manual adaptation if it is not being used; see Yang et al. (1992) for a similar approach for graphic cell management.

3.4 Knowledge of the System

This knowledge deals with what the system knows about itself. The system must know about its strengths and weaknesses and the appropriate time and form in which the adaptation should occur. Based on Sharatt's (1987) implementation of an adaptive interface to an electronic mail system, a rule could state:

if the adaptive feedback is not active and
the user's error level is two and
the user's help level is three
then make adaptive feedback active.

This rule determines when the adaptive feedback dialog should be active. The feedback dialog provides appropriate suggestions to the user about improving performance.

Modeling the user is a crucial input to such an adaptive environment. The following pilot study attempts to determine leading indicators that distinguish various types and levels of CAD experience.

4 Indicators of CAD Experience

The goal of our pilot study was to identify the leading indicators that distinguish novice CAD users, regular CAD users and expert users of Canvas. (Canvas is a 2-D drawing package for the Macintosh computer. It was chosen for the experiment because it has a minimal set of commands present in all CAD systems with a window interface that is reasonably intuitive for a new user).

Our study differed in goal and hypotheses from the experiment performed by Lang et al. (1991) discussed earlier. First, we believed that there was a class of users who use 2-D CAD systems without any formal CAD training and minimal experience; these users had not been studied before. All of the subjects in Lang's experiment had an introductory course in the use of the CAD system chosen for the experiment. Second, although the earlier study found no difference in command usage between the trained novices and the experts, we suspected that novices without the introductory course would differ in command usage. Because of this, the task given to our subjects tested a wide range of commands as discussed later. Third, we were interested in identifying specific behavior patterns in Canvas command usage that could be used in the design of an adaptive interface. Lang's experiment was directed towards identifying effective CAD teaching techniques for various levels of users. We did, however, agree with the earlier finding that experts would use more sophisticated planning and reach their goals sooner than novice users. Fourth, we were interested to see if behavior differences between different types of users were consistent across different tasks and systems; such comparative studies had not been reported.

To summarize, our hypothesis simply stated that experts use fewer and more complex commands, use more sophisticated planning, and reach their goals sooner than novice users. Six subjects were chosen from the CMU student and faculty of the Architectural and Civil Engineering departments. Two of the subjects had less than a week of experience using a CAD system, two had used CAD systems other than Canvas for between two and three projects in the past six months, and two were frequent users of the Canvas CAD system used in the experiment. The sample size was small as this pilot study, being exploratory in nature, was intended to give us suggestions for more extensive studies in the future.

The task consisted of drawing the plan of a dimensioned staircase with a curved landing, enclosing wall and door as shown in Figure 1. This task was designed to test the subject's ability of seeing and drawing repeated elements, merging curved and linear elements, dimensioning, labeling, scaling, styling and laying out a drawing. Each subject was instructed to reproduce accurately the drawing in Figure 1 using the Canvas package on the Macintosh computer. The instructions were read out and the drawing given to the subject. The subject was asked to "think aloud" as the task was being performed. The subject's verbal protocol was recorded on audio tape during the task execution and the completed drawing saved on disk.

Once the experiments were completed, the verbal protocols were encoded using the matrix shown in figure 2. The vertical dimension of the matrix shows the full range of commands used by all the subjects in the experiment. The commands were grouped into four categories: creation, modification, translation and setup commands. The horizontal dimension represents discrete command events; the numbers are from the tape counter providing information when interesting events occurred. The content of these events were also recorded. In the body of the matrix are several types of markings; a dot represents a completed command such as drawing a line, a dot in a circle represents a completed com-

mand using the keyboard, and an ‘x’ represents a command that was selected but not completed.

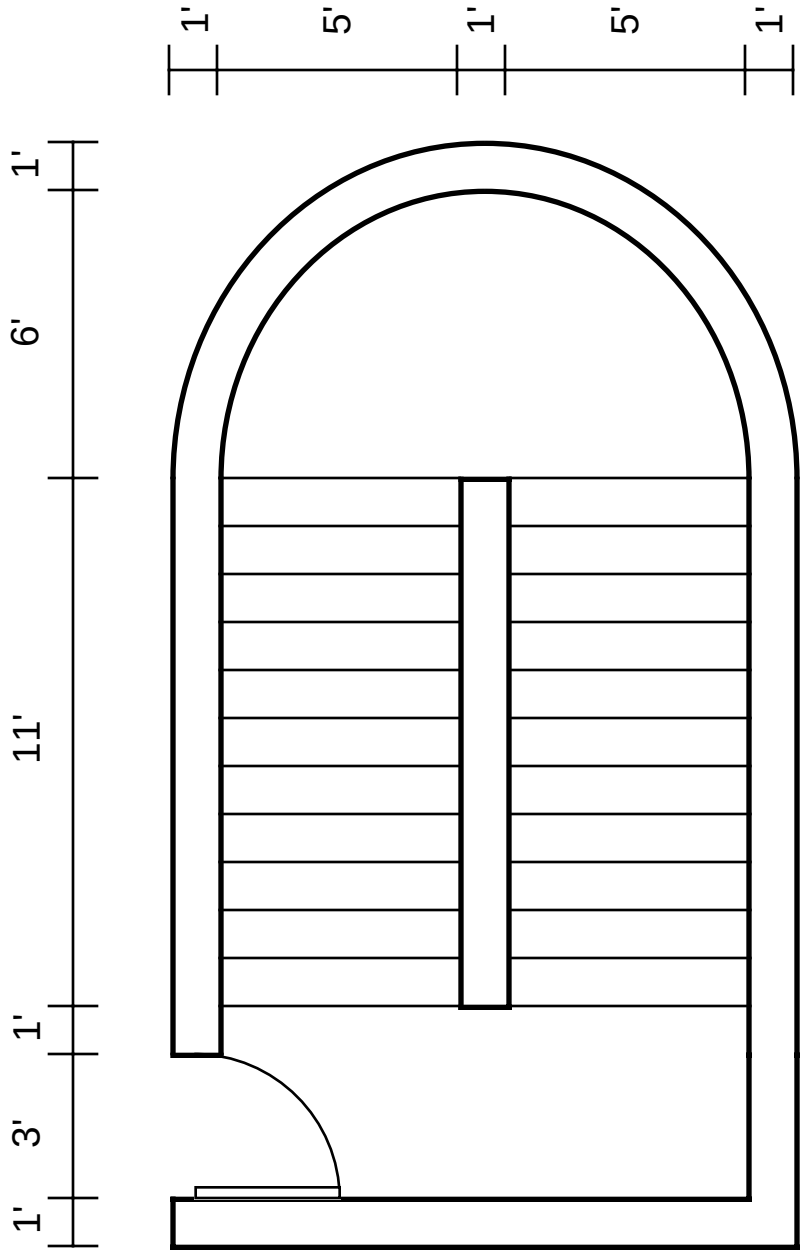

Figure 1. Drawing task given to subjects.

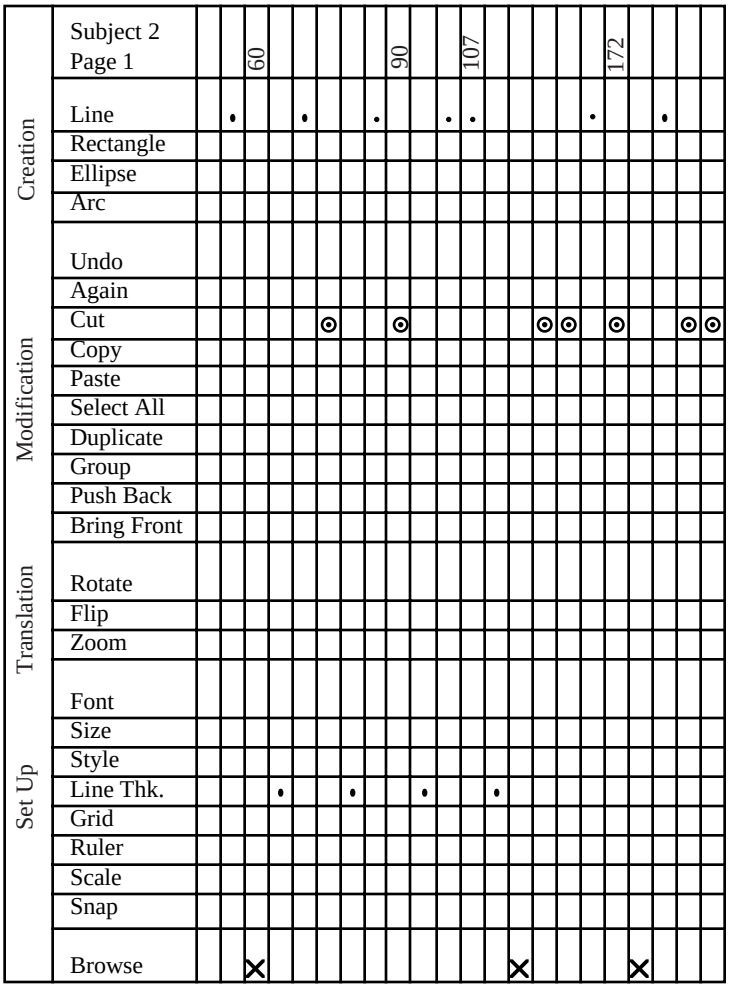

Figure 2. Encoding Matrix.

5 Results of the Pilot Study

Analysis of the encoded data and drawings show distinct behavior differences between the three categories of subjects. The differences show up in the type and sequence of commands used, the total time taken to complete the task, and the quality of the drawing produced.

5.1 Command Usage

Figure 3 shows the types of commands used by each of the three experience categories. In the low CAD experience category, each of the two subjects used a total of 10 commands to complete the task. (This number does not represent the events but the range of commands used. Hence each of the 10 commands was used one or more times during

the task). The high CAD experience users and expert Canvas user show a progressive increase in the types of commands used. The figure also shows distinct differences in the use of creation and setup commands between the experience categories. As expertise increases, creation commands form a smaller percentage of the overall vocabulary of a user; knowledge of setup commands, however, increases.

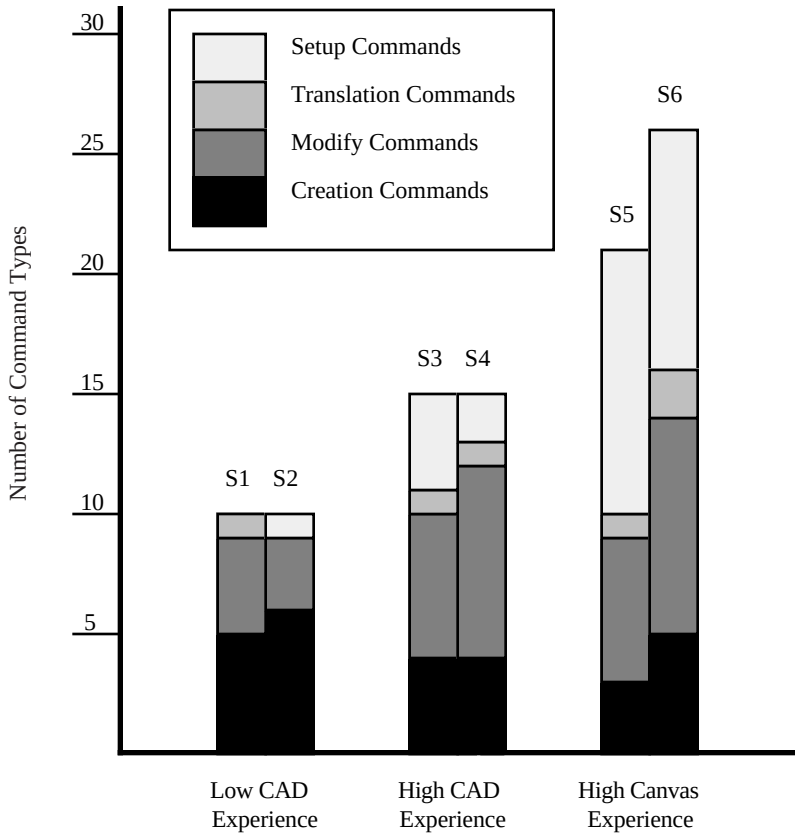

Figure 3. Command Vocabulary.

While observing the data, a frequently repeating command pair appeared in the low CAD user category. These were a create command immediately followed by an erase command. Figure 4 shows a distinct difference in the number of create-erase contiguous pairs in the three categories. Clearly, inexperienced users tend to delete elements that are not satisfactory (rather than modify them) more frequently than users in the other categories do.

5.2 Task Completion Time

The expert Canvas users took the least time to complete the task as shown in Figure 5. When studied in isolation, it might seem surprising that the high CAD experience users took more time to complete the task when compared to low CAD experience users. The reason becomes clear when the quality of their drawings and the total number of completed command events are also considered. The high CAD experienced users tried very hard to

find and use commands that they had used in other CAD packages and insisted on successfully producing dimensionally accurate drawings. As shown in Figure 6, they did not use more completed command events. (The figure does not include browsing events as they were frequently too rapid and non-verbal to be accurately captured in our audio recording). By spending more time than the other categories but not using more completed command events, we believe that the high CAD experience users spent much time browsing in an effort to find the commands they needed to produce better quality drawings.

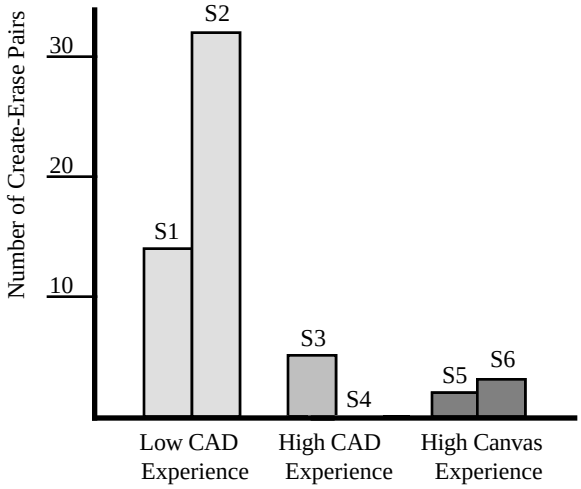

Figure 4. Number of Create-Erase Pairs.

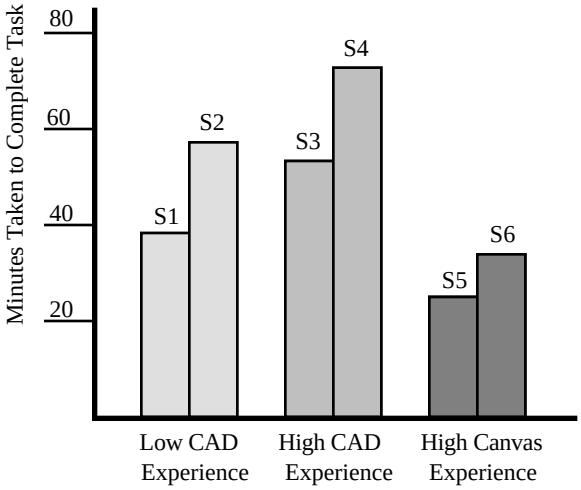

Figure 5. Time Taken to Complete Task.

The low CAD experience users, on the other hand, having little or no exposure to setup concepts like grid, scale and snap lock, became frustrated early and produced low

quality drawings. This resulted in shorter time spans (when compared to the high CAD experience users) to complete the task.

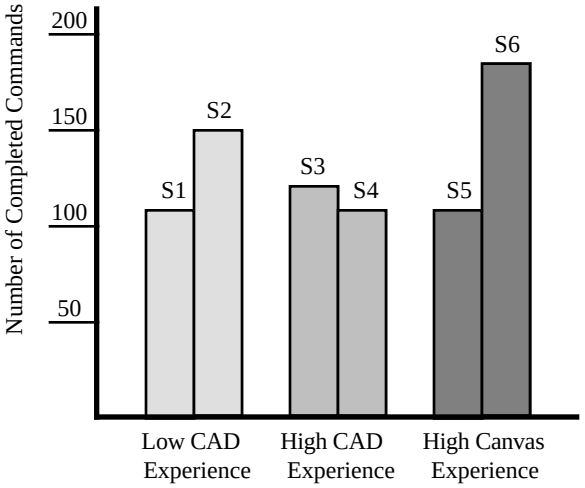

Figure 6. Total Number of Completed Events.

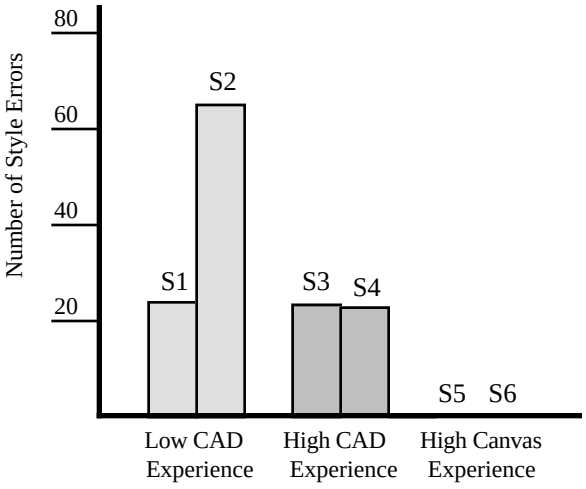

Figure 7. Number of Style Errors.

5.3 Quality of Drawing

Figures 7-9 show drawing errors that the users made on each of the 65 separate drawing elements in the task. A style error is an incorrect line thickness, font style or font size. A dimension error is an element not drawn to scale. A form error is an element which does not meet or line-up with another, a line that is not straight, or an arc which is not correctly drawn.

The low CAD experience users had more errors in all the categories when compared to the other users. Figures 10-12 show finished drawings for the three experience categories.

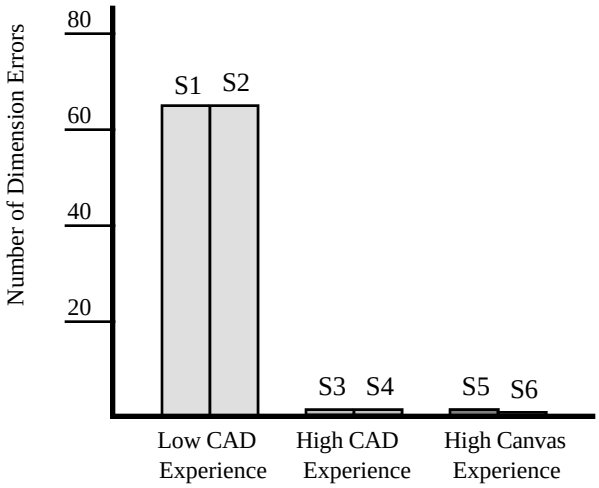

Figure 8. Number of Dimension Errors.

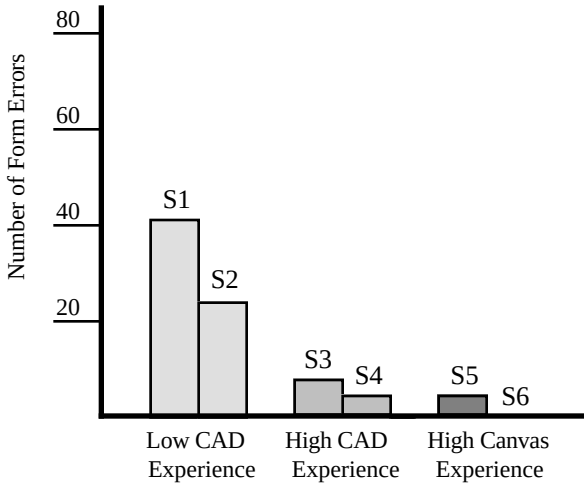

Figure 9. Number of Form Errors.

5.4 Discussion

The above results, though not counter-intuitive, are different from and significantly richer than our original hypothesis as discussed below.

1. We predicted that the experts would use complex commands and use fewer events to complete the task. This was clearly not the case as the total number of events had

no clear correlation to the user categories. Instead the CAD expert used a wider range of commands to achieve a higher quality drawing.

2. Expert Canvas users used more setup commands and less creation commands compared to low and high CAD experience users. This was not originally predicted.
3. The high number of create-erase contiguous events was an interesting behavior pattern exhibited by the low CAD experience user. This was also not hypothesized.

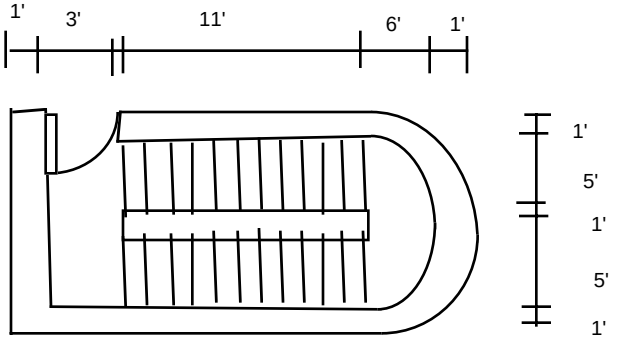

Figure 10. Drawing by Low CAD Experience Subject (S1).

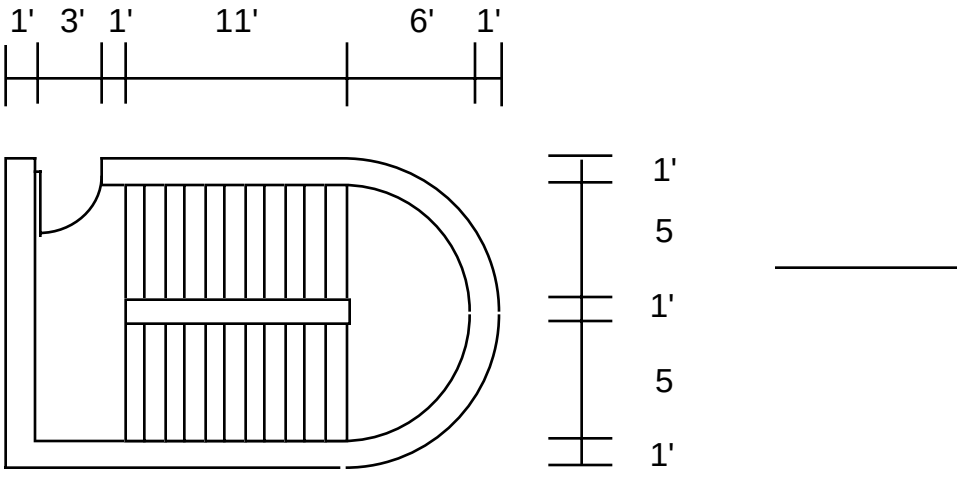

Figure 11. Drawing by High CAD Experience Subject (S3).

4. As hypothesized, the Canvas expert completed the task sooner than others. We also believe that the high CAD experience users took the most time, due to their insistence on quality but their inability to find commands to achieve it.
5. Finally, the quality of the drawing was not an issue in our hypothesis. To our surprise, the low CAD experience user made significant compromises and, although obviously unhappy with the drawing, decided to end the session.

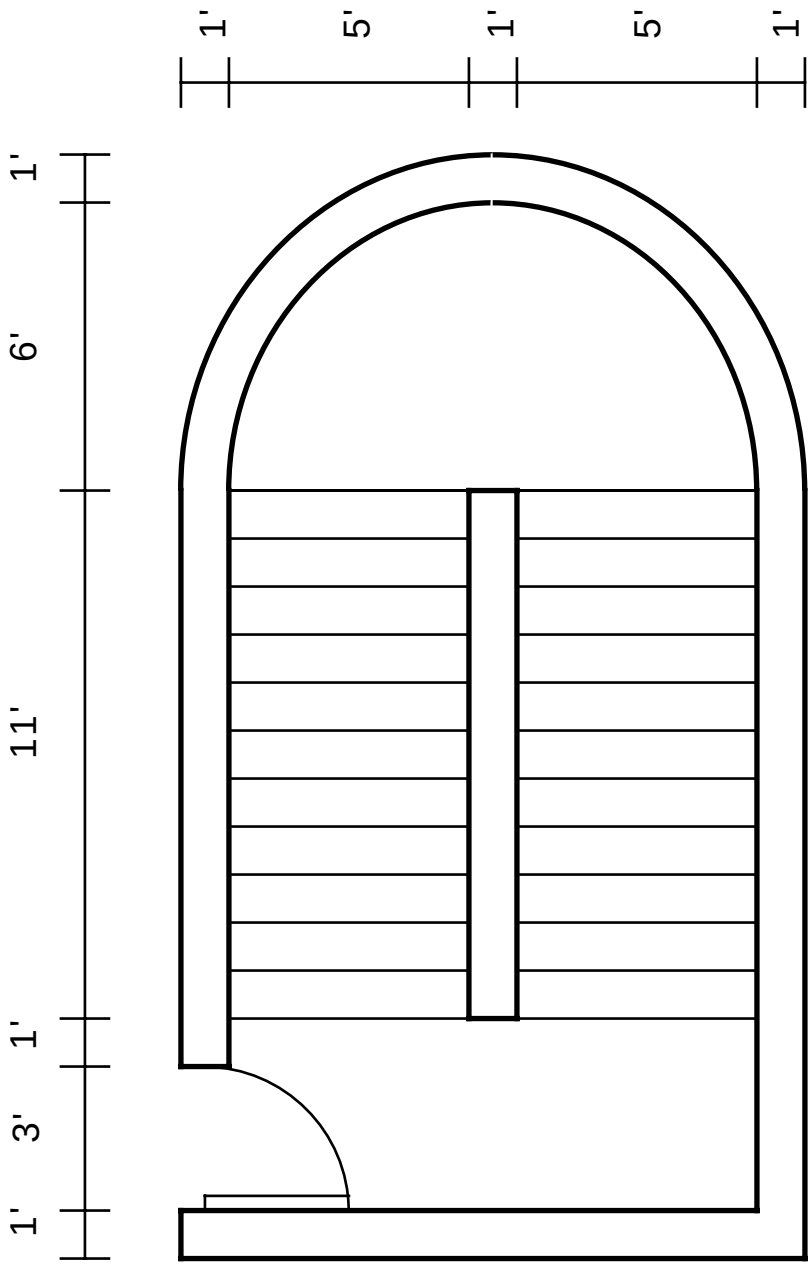

Figure 12. Drawing by High Canvas Experience Subject (S5).

6. Our results agree with and add to the results by Lang et al. (1991) discussed earlier. The novices, without any CAD instruction, were unaware of set-up concepts like ‘grid’ and ‘grid lock’ demonstrating a lack of essential declarative knowledge to

produce quick accurate drawings. The subjects with high CAD experience on other CAD systems were able to transfer their knowledge of CAD concepts and their procedural knowledge to complete the task. Their performance suffered most not because they did not know what commands to use but because they were unable to find them. In contrast, the Canvas expert not only had a good grasp of declarative and procedural knowledge, but also knew where to find the commands.

As a result of these observations, the experiment deepened our understanding of the behavior patterns of CAD users. The experiments confirmed our belief that users of different skill levels had distinct behavior patterns. The next section describes a second experiment in which we designed a neural network to classify a user's skill level based on command usage.

6 A Neural Network-based Approach to Modeling CAD Users

6.1 Representations Used to Model a User

The experiments, described in the previous section, demonstrated that the high band-width of information that is collected during verbal protocols enables researchers to study and characterize user differences. In contrast, the low band-width of information available during actual human-computer interaction makes user modeling a relatively more difficult task. For example, during actual interaction, the system does not have access to verbal protocols or a way to assess the quality of drawings. User modeling, therefore, relies mainly on different aspects of command usage. Many different representations have been used to model the user in adaptive as well as in intelligent tutoring systems. The information being monitored has been represented using production systems (Tyler, 1986; Anderson, 1990; Fisher and Lemke, 1987), fuzzy sets (Norcio, 1991), genetic graphs (Copeland and Eccles, 1992), and neural networks (Finlay and Beale, 1990; Fels and Hinton, 1990; Chen and Norcio, 1991). Future adaptive systems will need to exploit the strengths of one or many of these representations depending on the quality of information being monitored (Chen and Norcio, 1991).

6.2 Motivation to Use a Neural Network Approach

Earlier we identified two important aspects of a user model: user skill level and cognitive ability. As the goal of our pilot study was to identify differences in skill level, we will concentrate on using the data to model and infer the skill level of a user. The pilot study showed some clear behavior signatures for each of the three experience categories. Although a few of these behavior signatures were easily identified in our analysis, many other indications of skill level remain to be discovered as the command sequences are quite noisy. (The command sequences are considered noisy as it is improbable that every user in a class would use the exact sequences of commands identified as their behavior signature. Instead users in a class will have command sequences that are similar but not identical to each other). An adaptive system must therefore have the ability to learn complex behavior signatures from noisy behavior patterns of users and then detect these signatures from noisy data in real time. This skill level recognition task is thus a problem of mapping from a noisy pattern of input to a classification of skill level.

One machine learning approach that has shown a great amount of success in learning classification mappings for noisy input is the connectionist, or neural network-based, approach. (See Dayhoff (1990) for an introduction to neural networks.) For example, in the Glove-Talk system developed by Fels and Hinton (1990), a feedforward neural net-

work is used to map from digitized hand signals, generated by a human wearing a glove with absolute and relative position sensors, to the phonemes of the spoken word representing that hand signal. The phonemes are then fed into a modified digital signal processor to produce the actual utterance. The hand signals are noisy in that the human producing them is not likely to make them exactly the same way every time (much like handwriting). The phonemes are the classifications of the various hand signals. Thus, the Glove-Talk system is a perfect model of how the noisy CAD command sequences might be mapped into user skill level classifications. Thus, as described later, we used a feedforward neural network, of the same type used by Fels and Hinton, to acquire the mapping between CAD user command sequences and skill level.

Other researchers have also considered how neural networks might be used in user modeling. Finlay and Beale (1990) recognize that the machine learning abilities of some types of neural networks (associative memories and feedforward networks) can be used to perform dynamic user classification of skill level or task (thus, knowledge of application may also be gained by taking a similar approach). They viewed the human-computer interaction as a sequence of events to be input to the neural network, which produces as output a corresponding classification of that sequence of events; their domain of computer interaction was that of computer programming. Chen and Norcio (1991) have illustrated how a neural network can also be employed to model a user of an information retrieval system in order to provide that user with a tailored response to an information request. They, too, chose the neural network as a mechanism for providing a fault tolerant mapping between a "user image" (i.e., the information known about a user) and other classifications of the user that need to be inferred so as to be able to tailor an information request, such as skill level or level of expertise in an application area.

6.3 A Neural Network Experiment

A neural network that estimates the skill level of a CAD user requires some representation for the input layer (representing the raw data input) and an output layer representing the categories of experience. While the representation of the skill level classification in the output layer must consist of either a set of binary or a graded set of output units, the input layer could be represented in a variety of different ways. Two representations for the input layer are presented below.

The simplest approach is to have the units of the input layer represent each of the possible commands. The activation values of these units would represent the frequency of the command usage by the user over some specified period of time, normalized over the total number of commands used by the user. This type of network would learn, for example, that experts use more setup commands compared to inexperienced users.

Another approach is to have a discrete set of sequential events form the input to the neural network as shown in Figure 13. The prototype neural network accepts as input the first 20 events of the users (another neural network could accept as input the next 20 events and so on). Each event is represented by a set of four input units representing each of the possible classes of commands (creation, modification, translation, and setup) that can be invoked within an event. The neural network therefore has a total of 80 units in the input layer. The output layer has three units, each representing a category of experience used in the experiment (low CAD experience, high CAD experience, and high Canvas experience). The hidden layer has four units, a number which was determined empirically for the data set used to train the network.

Shown below is a sample training set for six subjects.

! SUBJECT 1 1 0 0 0, 0 1 0 0, 1 0 0 0, 0 1 0 0, 1 0 0 0, & 0 1 0 0, 1 0 0 0, 0 1 0 0, 1 0 0 0, 0 1 0 0, & 1 0 0 0, 0 1 0 0, 1 0 0 0, 0 1 0 0, 1 0 0 0, & 0 1 0 0, 1 0 0 0, 0 1 0 0, 1 0 0 0, 0 1 0 0, & 1 0 0	! SUBJECT 2 0 0 0 0, 1 0 0 0, 0 0 0 0, 0 0 1 0, 1 0 0 0, & 0 1 0 0, 0 0 1 0, 1 0 0 0, 0 1 0 0, 0 0 1 0, & 1 0 0 0, 1 0 0 0, 0 0 1 0, 1 0 0 0, 0 1 0 0, & 0 1 0 0, 1 0 0 0, 0 1 0 0, 0 0 0 0, 0 0 0 0, & 1 0 0
! SUBJECT 3 0 0 0 0, 1 0 0 0, 0 1 0 0, 0 0 1 0, 0 0 1 0, & 0 0 0 0, 1 0 0 0, 1 0 0 0, 0 1 0 0, 0 1 0 0, & 0 0 0 0, 0 0 0 0, 0 0 0 0, 0 0 0 0, 0 0 0 0, & 1 0 0 0, 0 1 0 0, 0 1 0 0, 0 0 0 1, 0 0 0 0, & 0 1 0	! SUBJECT 4 0 0 0 0, 0 0 1 0, 0 0 0 0, 0 0 0 0, 1 0 0 0, & 0 1 0 0, 0 1 0 0, 0 1 0 0, 0 1 0 0, 0 1 0 0, & 0 1 0 0, 0 1 0 0, 0 1 0 0, 0 1 0 0, 0 1 0 0, & 0 1 0 0, 0 1 0 0, 0 0 0 1, 0 1 0 0, 0 0 0 1, & 0 1 0
! SUBJECT 5 0 0 1 0, 0 0 1 0, 0 0 1 0, 0 0 1 0, 0 0 1 0, & 1 0 0 0, 1 0 0 0, 1 0 0 0, 1 0 0 0, 1 0 0 0, & 1 0 0 0, 1 0 0 0, 1 0 0 0, 0 1 0 0, 1 0 0 0, & 1 0 0 0, 0 1 0 0, 1 0 0 0, 0 1 0 0, 0 1 0 0, & 0 0 1	! SUBJECT 6 0 0 1 0, 0 1 0 0, 0 0 0 1, 0 0 1 0, 0 0 1 0, & 0 0 1 0, 0 0 1 0, 0 0 1 0, 1 0 0 0, 1 0 0 0, & 1 0 0 0, 0 1 0 0, 0 0 1 0, 1 0 0 0, 0 1 0 0, & 0 1 0 0, 0 1 0 0, 0 1 0 0, 0 1 0 0, 1 0 0 0, & 0 0 1

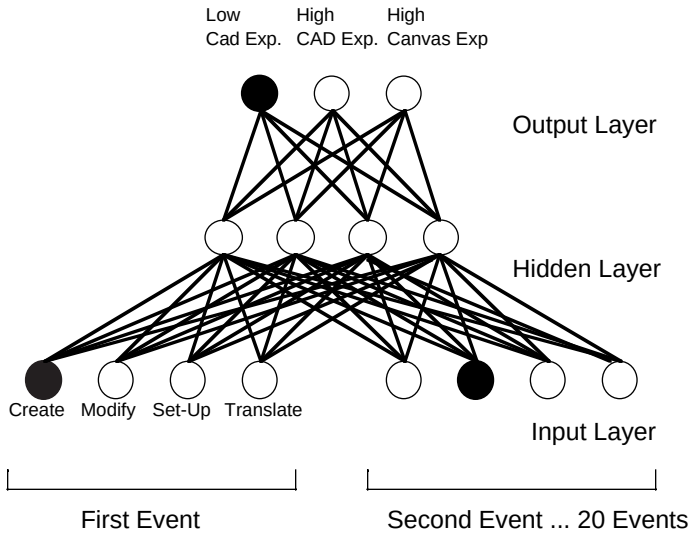

Figure 13. Design of the Neural Network.

The first four lines of each subject represent the training data for each of the 80 input layer units and the fifth line of each subject represents the training data for the output layer. Therefore, the data for the first subject shows that the first event chosen (represented by the first four numbers 1 0 0 0) was a creation command. Similarly, the second event chosen (represented by the next four numbers 0 1 0 0) was a modify command. The data also shows that this subject was a user with low CAD experience (represented by the three numbers 1 0 0 in the fifth row). As can be observed in the first subject, there is a repeated

pattern of create-erase pairs in the first 20 events. The neural network was trained on the training set for 15000 epochs using the backpropagation training method (Rumelhart and McClelland, 1986).

Shown below is a sample of the behavior of a fictitious subject used as a test case.

```
!   TEST SUBJECT 1
  0 0 1 0, 0 0 1 0, 0 0 1 0, 0 0 1 0, 0 0 1 0,
& 1 0 0 0, 1 0 0 0, 1 0 0 0, 1 0 0 0, 1 0 0 0,
& 1 0 0 0, 1 0 0 0, 1 0 0 0, 1 0 0 0, 0 1 0 0,
& 0 1 0 0, 1 0 0 0, 0 1 0 0, 0 0 0 0, 0 0 0 0,
& 0 0 1
```

This test case is not part of the training set. The format is the same as the training set although the data on the fifth line represents the expected output. As can be seen, this fictitious subject used five setup commands (represented by the event 0 0 1 0) followed by five creation commands (represented by the event 1 0 0 0). The fifth line of data shows that this subject is expected to be an expert CAD user (as represented by the numbers 0 0 1).

Shown below are the results when this test case was presented to the trained neural network.

```
0.000000  0.000000  1.000000
0.078611  0.001413  0.964814
```

The first three sets of numbers represent the expected output as given in the test data, and the next three sets of numbers represent how the neural network predicted the behavior of the subject. As can be seen, the neural network predicted quite accurately that the subject was an expert.

As discussed earlier, the neural network approach was considered in anticipation of more noisy and complex behavior patterns in training data as well as during real time interaction. This is only hinted by the limited data from the pilot study. Additionally, the same approach can be used to determine skill level at a finer-grain command level. More data and more testing are needed to determine if this neural network approach is appropriate.

7 User Modeling within an Adaptive System

The results of this small pilot study suggested some ideas about the overall adaptive environment. If the behavior signatures of the different categories of experience are so distinct, and if it is possible to detect them using a neural network approach, is it possible and appropriate to inform users of ways to improve their performance? Is it possible, for example, to detect that a user has not used setup commands like “grid” and “snap” in the first 20 command events and intercept that behavior? If, on the other hand, the user has used “grid” and “snap” might it be possible to introduce the user to more advanced set up commands like ‘delta dimensioning’? Such an adaptive environment would then emulate a good teacher; one that provides feedback by discrete interception. The interception would encourage and reward expert-like behavior patterns and discourage novice-like behavior in real time.

This section discusses how neural network-based user models like that described in the previous section (i.e., neural network-based mappings from CAD user behavior to var-

ious classifications such as skill level, information presentation preferences, etc.) can be used within an adaptive CAD system.

One possible architecture for such an adaptive system is shown in figure 14. Here the adaptive user interface module controls the form and content of interaction between the user and the CAD system. This module consists of a user interface manager that creates four different models of the overall interaction; a user model, an interface model, an application model and a system model. Each of these four models may use different knowledge representations (rules, frames, procedures and neural networks) to translate the user interactions into meaningful interface adaptations.

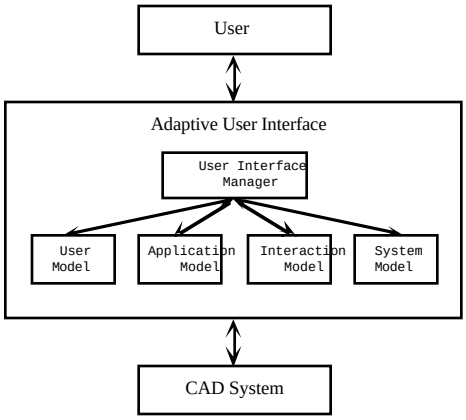

Figure 14. Architecture of an Adaptive CAD System.

In this system, the user interface module continually processes the last 20 user events using the four knowledge modules. For example, the user model would use the command sequences to determine the user's skill level and the application model would determine if the command is being used appropriately. If there are problems with the command usage, the interface model could use the skill level (determined by the user model) and the usage problem (determined by the application model) to decide the level of assistance required (i.e., the adaptation to be made to the interface).

In order for the interception suggested above to be beneficial, it is important that it occurs in a non-disruptive manner. It is proposed that the interception occurs in the form of a message box that flags the user when the system has generated a suggestion. In such an implementation, the user can choose to attend or ignore the suggestions generated.

8 Future Experiments

The pilot study, being exploratory in nature, provided feedback about our experimental techniques, results, and user modeling representations. Our future direction is discussed below.

1. Future neural network experiments should explore the space of available neural network architectures. There must also be a more rigorous understanding of the strengths of other user modeling representations.

2. It is not clear which behavior signatures are independent of the drawing task given in our experiment. Future experiments on other drawing tasks must be performed to determine which behavior signatures can be generalized across all drawing tasks and which are specific to a particular drawing task.
3. It must be clearly demonstrated that interception can improve the performance of a user. It is not clear whether the learning of a new concept like "snap lock" can occur quickly and effectively enough to improve performance. It is also not clear at what point this interception is most appropriate.
4. All the categories of users exhibited search and/or browsing behavior. Because this behavior was frequently too rapid and nonverbal, the audio recorded verbal protocols were inadequate. Future experiments must be videotaped to record and study this information.
5. Future cognitive studies should explore more explicitly goal states, cognitive ability variables, and the acquisition of CAD expertise over time.

The above experiments are motivated by a single purpose; to perform more rigorous experimentation leading to the development of an adaptive interface for a commercially available CAD system. Although we know the old adage "there is nothing so useful as a good theory," in the absence of one we have decided to progress with approximations and iteratively refine them. We believe that "design is where the action is" and "nothing derives basic science better than a good application" (Newell and Card, 1985).

9 Conclusions

In this paper, we cited strong evidence showing the dismal performance of CAD users. We proposed the argument that this may be due to an increasingly complex CAD environment that is unresponsive to the needs of a heterogeneous community of users. We cited empirical results showing the precise nature of differences between computer users based on skill level and cognitive abilities. Based on various implementations, we discussed the four knowledge components of an adaptive system: knowledge of the user, knowledge of the application, knowledge of the interaction and knowledge of the system.

Having presented the motivations and foundations of the adaptive concept, we then discussed the results of a pilot experiment, where we attempted to study the differences between different types of CAD users. We found that the type and experience of the CAD user have a clear correlation to the pattern of commands used, the time taken, and the quality of drawing produced. These results agree with and add to what little is known about the differences between CAD users.

We then demonstrated how the experimental data can be used to model a user using a neural network approach. We discussed various neural network designs and some ideas about an overall adaptive architecture, where a combination of representations could be used to model a user. Finally, we presented future experiments that would lead us to our final goal: to design an adaptive interface for a CAD system.

We believe that if today's CAD interfaces are designed for an unchanging canonical user and allowing minimal customization, tomorrow's CAD interfaces should dynamically adapt to the needs of many different users.

Acknowledgments

We thank Ulrich Flemming for his encouragement and excellent comments. The research was supported by The United States Army Construction Engineering Research Laboratory (USA-CERL).

References

Anderson, J. R., 1983. *The Architecture of Cognition.* Cambridge: Harvard University Press.

Anderson, J.R., Boyle, F.C., Corbett, A.T., and Lewis, M.W., 1990. "Cognitive Modeling and Intelligent Tutoring,"*Artificial Intelligence* 42, pp. 7-49.

Andes, R.C., and Rouse, W.B., 1990. "Specification of Adaptive Aiding Systems," IEEE *International Conference on Systems, Man and Cybernetics Conference Proceedings*, pp. 4-7.

Benyon, D., and Murray, D., 1988. "Experience with Adaptive Interfaces," *Computer Journal* 31(5), pp. 465-73.

Bietz, W., Langner, T., Luczak, H., Muller, T., and Springer, J., 1990. "Evaluation of a Compact CAD-course in Laboratory Experiments," *International Journal of Human Computer Interaction* 2(2), pp. 111-35.

Copeland, J.C., Eccles, S.R., 1992. "Skill Metrics on a Genetic Graph as a Mechanism for Driving Adaptation in an Operating System Interface," *International Journal of Man-Machine Studies* 36, pp. 697-718.

Chen, Q, and Norcio, A.F., 1991. "A Neural Network Approach to User Modeling," *Proceedings*, IEEE 1991 International Conference on Systems, Man, and Cybernetics, pp. 13-16.

Cuomo, D.L, and Sharit, J., 1989. "Human Performance in Computer-Aided Architectural Design," *Proceedings*, Third International Conference on Human-Computer Interaction 2, pp. 241-9.

Dayhoff, J., 1990. *Neural Network Architectures—An Introduction*. New York: Van Nostrand Reinhold.

Doane, S.M., Pellegrino, J.W., and Klatzky, R.L., 1990. "Expertise in a Computer Operating System: Conceptualization and Performance," *Human-Computer Interaction* 5, pp. 267-304.

Draper, S., 1984. "The Nature of Expertise in UNIX," in B. Shackel (ed.), *Human Computer interaction-Interact '84*, Amsterdam: North Holland, pp. 465-471.

Fels, S.S., and Hinton, G.E., 1990. "Building Adaptive Interfaces with Neural Networks: The Glove-talk Pilot Study," in D. Diaper, D. Gilmore, G. Cockton, B. Shackel (eds.), *Human Computer Interaction-Interact '90*, pp. 683-8.

Finlay, J., and Beale, R., 1990. "Neural Networks in Human-computer Interaction: A View of User Modeling," *IEEE Colloquium on Neural Nets in Human-Computer Interaction*, No.179, pp. 1-3.

Fisher, G., and Lemke, A.C., 1987. "Construction Kits and Design Environments: Steps Toward Human Problem-Domain Communication," *Human-Computer Interaction* 3, pp. 179-222.

Greenberg, S., and Witten, I.H., 1985. "Adaptive Personalized Interfaces — A Question of Viability," *Behavior and Information Technology* 4(1), pp. 31-45.

Krogsaeter, M., Bruning, I., and Thomas, C.G., 1992. "FLEXEL: Adding Flexibility to a Commercially Available Software System," Sankt Augustin: Gesellschaft für Mathematik und Datenverarbeitung mbH.

Lang, G.T., Eberts, R.E., Gabel, M.G., and Barash, M.M., 1991. "Extracting and Using Procedural Knowledge in a CAD Task," *IEEE Transactions on Engineering Management* 38, pp. 257-68.

Luczak, H., Beitz, W., Springer, J., and Langner, T., 1991. "Frictions and Frustrations in Creative-Informatory Work With Computer Aided Design-CAD-Systems," *Proceedings*, of the Fourth International Conference on Human-Computer Interaction, pp. 175-9.

Majchrzak, A., 1990. "Effect of CAD on the Jobs of Drafters and Engineers: A Quantitative Case Study," *International Journal of Man-Machine Studies* 32(3), pp. 245-62.

Newell, A., and Card, S.K., 1985. "The Prospects for Psychological Science in Human-Computer Interaction," *Human-Computer Interaction* 1, pp. 209-242.

Norcio, A.F., 1989. "The Software Engineering of Adaptive Human-Computer Interfaces," in *Proceedings*, IEEE International Conference on Systems, Man and Cybernetics. Cambridge, MA, pp. 14-17.

Norcio, A.F., 1991. "Adaptive Interfaces: Modeling Tasks and Users," in *Proceedings*, IEEE International Conference on Systems, Man and Cybernetics, Charlottesville, VA, pp. 13-16.

Norcio, A.F., and Stanley, J., 1989. "Adaptive Human-Computer Interfaces: A Literature Survey and Perspective," *IEEE Transactions on Systems, Man, and Cybernetics* 19(2), pp. 399-408.

Rich, E., 1983. "Users Are Individuals: Individualizing User Models," *International Journal of Man-Machine Studies* 18(3), pp. 199-214.

Rumelhart, D.E., and McClleland, J.L., 1986. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* 1, Cambridge: The MIT Press.

Sein, M.K., and Bostrom, R.P., 1989. "Individual Differences and Conceptual Models in Training Novice Users," *Human-Computer Interaction* 4, pp. 197-229.

Sharratt, B., 1992. "The Development of Heuristics for Self Adaptive Systems: A Case History," report / Scottish HCI Center, Edinburgh, UK.

Shaw, D.S., Golish, M.L., Webster, J.L., and Yang D., 1991. "Adaptive On-Line Help for Embedded Instructional Systems," *USACERL Technical Report* P-91/52.

Totterdell, P.A., Norman, M.A., and Browne, D.P., 1987. "Levels of Adaptivity in Interface Design," *Human Computer Interaction-Interact '87*, pp. 715-22.

Tyler, S.W., 1986. *SAUCI: Self-Adaptive User Computer Interfaces*. Ph.D. Dissertation, University of Pittsburgh, Pittsburgh, PA.

Tyler, S.W., and Treu, S., 1989. "An Interface Architecture to Provide Adaptive Task-specific Context for the User," *International Journal of Man-Machine Studies* 30(3), pp. 303-27.

Vaubel, K.P., and Gettys, C., 1990. "Inferring User Expertise for Adaptive Interfaces," *Human Computer Interaction* 5, pp. 95-117.

Yang, D., Garrett, G.H., and Shaw, D.S., 1992. "Cell Management Using Neural Network Approaches," *Proceedings*, Government Neural Network Applications Workshops 1, pp. 19-23.