JEEPERS: AN INTERFACE PERCEPTION RESEARCH TOOL

by

Suzanne Weghorst
Werner Stuetzle

TECHNICAL REPORT No. 185
November 1989

Department of Statistics, GN-22
University of Washington
Seattle, Washington 98195 USA
Jeepers: An Interface Perception Research Tool

Suzanne Weghorst  
Computer Science and Engineering, FR-35  
University of Washington  
Seattle, Washington 98195  
suzanne@june.cs.washington.edu  
(206) 547-4596

Werner Stuetzle  
Statistics, GN-22  
University of Washington  
Seattle, Washington 98195  
wxs@entropy.ms.washington.edu  
(206) 543-0711

Abstract

The graphical user interface introduces a high degree of uncertainty into software performance, suggesting a need for empirical exploration of interface perception. Jeepers was designed to be a broadly accessible tool to assist with research into interface perception problems. Three components of perceptual research are supported: (1) the production of stimulus displays, (2) management of the data collection session, and (3) management of experimental parameters and case data. The underlying research model and the HyperCard-based tool are described. Application of Jeepers to a problem in the area of data visualization revealed specific new findings. Other potential interface research applications are discussed.

Word Count: 4020

Topic: Analysis, Methodology, and Case Studies
The General Problem

Raster graphics monitors have liberated the computing community from the character stream interface. With pixel-level control it is now feasible to create elaborate and responsive visual displays. The introduction of the mouse (Engelbart and English, 1968) and other pointing devices provided a way to interact directly with such displays, and paved the road for the evolution of the direct manipulation interface (Hutchins et al., 1986; Shneiderman, 1983). Today human-computer interaction is more likely to be open-ended and user event-driven than procedurally-driven, and input and output are increasingly more likely to be graphical or locational than textual. But programs which rely on user behavior also depend on the user's perceptions. User perception of the program interface has thus become an important factor in determining software effectiveness (and in some sense its "correctness").

The Need for User Perception Research Tools

Throughout its brief history computer science has asked questions which have been generally more amenable to analytical than to empirical study. Empirical research methodology, with its concerns for internal and external validity, has not played a major role in the tradition of computer science. Direct manipulation interfaces introduce a strong element of uncertainty into software performance, however. As more "error variance" is introduced into the systems under study, we are increasingly compelled to supplement formal analysis with empirical study of user-system interaction.

Whether interface research is conducted by programmers, by interface research specialists working in collaboration with programmers, or by more broadly trained software engineers, the need for user interface research tools is certain. As with other areas of human endeavor, we should be finding ways to relegate its algorithmic aspects to computers, while building "intelligent" assistants to help us with those aspects which require human thought.
Research design should always be motivated by a concern for both the \textit{internal} and the \textit{external validity} of any conclusions we hope to draw. \textit{Internal} validity reflects our confidence that we have accurately accounted for our observations. \textit{External} validity reflects our confidence that our findings generalize to the phenomena of interest.

One way to achieve external validity is to assure that the \textit{demand characteristics} of the data collection situation simulate those of the situation to which we wish to generalize. Computer interface research holds some unique advantages in this regard, since the object under study can often serve as its own research environment. The computer itself can unobtrusively measure user behavior through the use of \textit{event history} recording and \textit{instrumented prototypes} (Landauer, 1988; Monk, 1985). Experiments can also be \textit{embedded} in real-time user interaction with the system. These and other program-controlled data collection methods help to maintain experimental realism and to minimize differential \textit{experimenter expectancy} effects across subjects.

\textit{Functional Requirements of the Tool}

A \textit{single} approach to studying user perception of various interface features (and their interactions) and a \textit{single} general-purpose research tool are probably precluded. There will always be a need for clever new research methods, particularly as unforeseen interface features evolve. As in other fields, the evolution of research methods is subject to the influence of many factors, including current technologies, understanding of the subject matter, and the conceptual framework in which the research is conducted.

Nonetheless there are many questions which \textit{can} be approached using relatively simple methods. The tool described here was designed to investigate questions about \textit{user perception} of CRT displays using a \textit{fixed-protocol controlled-variable experimental design} (in contrast to such methods as field experimentation or naturalistic observation). We identified three relevant research functions which could benefit from computer automation:
- The generation of displays to serve as test stimuli
- The data collection process
- Data management

Several aspects of *stimulus display production* are amenable to automation. Some displays may be produced under program control; for example, in studying the influence of various chart parameters on the perception of relative bar heights in a bar chart, we might randomly sample parameter values and then paint the chart algorithmically. The conversion of captured displays into a format required by the data collection tool might also be automated.

A useful *data management* tool needs to handle both subject data and experimental parameters. In an active research program many experiments may be under way concurrently; the data manager must aide the researcher in monitoring each of them. Data formats should be compatible with major statistical packages, and data exportation facilities must be included in any acceptable data management tool.

One approach to automating the *data collection* process is to adopt an experimental protocol which provides a framework for research into a wide variety of questions. The protocol adopted here is presented in Figure 1.

![Figure 1. The general research design model.](image)

An initial interface training phase introduces the subject to the method to be used to indicate their responses. Task training provides more specific
information about the experimental task the subject is to perform. Experimental data collection is structured as a loop in which a display is presented (for either a pre-determined or a subject-determined length of time) and the subject is then required to make some response. Series of display-response sequences may be interspersed with additional training or with rest periods. Summary performance feedback and the collection of additional self-report data (e.g., demographic information) may follow the experimental data collection phase.

Selecting an Implementation System

It was not immediately obvious which computer system (both hardware and software) would be most appropriate for implementing such a tool. We wanted a system that would (1) be widely accessible to researchers likely to study computer user perception, (2) provide effective tools for creating visual displays, (3) permit us to iteratively build a prototype fairly easily, (4) support a friendly user interface for non-programming researchers, and (5) be easily customized by research users with varying degrees of programming expertise.

We settled upon the HyperCard programming environment on the Macintosh system (Apple, 1988) because it met all of these requirements and it appeared on the scene at the right time. Its major strengths were its integrated painting tools, its simple interface to other Macintosh software (such as text processors and statistics packages) via the clipboard, its broad accessibility, and its programmability at various levels of computer expertise. In addition, the screen-based metaphor used by HyperCard seemed well suited to our general research paradigm.

Description of the Research Tool

Although Jeepers derives its name from the term graphical perception, it can be used to study a somewhat broader range of phenomena than the name suggests. Its main functions are to aid with each of the research activities described above. Research subjects interact directly with Jeepers in
experimental sessions which follow the model presented in Figure 1, and experimental data are collected and stored automatically.

The package consists of a main HyperCard "stack" (a file consisting of one or more interactive screens of information, called "cards") supplemented by a number of smaller stacks, as depicted in Figure 2. These auxiliary stacks provide modular alternatives for use in experimental protocols. A set of introductory stacks train the subject in the interface used during the experimental task. Training stacks provide more specific task training before each series of displays, presented in one or more display stacks.

![Diagram](image)

Figure 2. Overview of the Jeepers package.
Other components shown in Figure 2 are (1) ASCII text files which contain supplementary information, some of which may be specific to the particular research problem under study (as are the "point coordinates files" for the study described below), (2) a "Tour" stack which provides a researchers' introduction to Jeepers, (3) a "Link" stack which provides subjects with a simple desktop entree into the data collection session, and (4) a programmers' introduction to the HyperCard language.

**Stimulus Display Generation**

Jeepers currently includes several display generators, each constructed for a specific research problem. We have spent the greatest effort on developing those for research in the area of data graphics. Stimulus displays may also be produced "manually" and then formatted automatically for use with Jeepers. In general, each display exists on a single card and each series of display cards exists in the desired sequence in a single stack.

**Experimental Session Management**

The heart of the Jeepers program is the Session Manager card, which provides a mechanism for specifying the structure of an individual data collection session. As a session proceeds the active procedures refer to the parameter values on this card to ascertain the current experimental protocol.

Experimental session parameters set by the researcher include: (1) an introductory card which typically describes what the study is about and how many displays will be presented, (2) an optional interface training stack, (3) a display manager card which serves as the input focus for subject responses and coordinates the presentation of displays and display-related training, (4) a set of one or more task-specific training stacks, (5) a set of one or more stimulus display stacks, (6) an optional performance summary card, and (7) a set of optional "subject variable" cards which may be used to measure self-report variables. Display cards are shown for a period of time determined either *a priori* by the researcher or at run-time by the subject's behavior.
Data Management

The researcher is given the option of saving each set of experimental parameters; these can then be automatically installed on the Session Manager as needed. This parameter storage mechanism permits the researcher to conduct several studies within the same time frame (or to run multiple sets of experimental conditions in the same study).

A data management card is automatically created for each experiment. This card stores both the session parameters and the case data for each subject, written to this field automatically during a session. Data may be exported to an ASCII file for use within a statistical analysis program.

User Assistance

Feedback from pilot research users and reviewers led us to a multi-pronged approach to user help. The Tour stack introduces the program concept and its major functions. A series of general information cards is easily accessible when Jeepers is opened. General help is also available through the "Help" command in the Jeepers menu. Context-specific help is available in two formats: (1) messages which explain the functions and options present on the current card, and (2) researcher and programmer notes accessed via the Jeepers menu.

Modifying Jeepers

Jeepers is designed to serve as a framework for a fairly wide variety of research problems. A considerable number of original experiments may be generated simply by setting parameters on the display generators and the Session Manager. These "standard" modules may also be viewed as templates, requiring only simple text editing to customize to a new research problem. Other features are not so easily parameterized, however, and their modification requires knowledge of the HyperTalk scripting language. The programmer notes mentioned above offer some guidance here.
A Concrete Example

The effectiveness of any tool is best judged within the context of its application. The development and evaluation of Jeepers was guided by a specific research problem in the area of data visualization for exploratory data analysis. The problem was chosen both because it was of genuine research interest and because it represents a class of questions for which the programmatic research method described here is quite appropriate.

The Painting Problem

In contrast to "virtual worlds" graphics, data graphics are used to represent abstract relationships among variables as revealed by sets of sample data points. Prior to Cleveland's pioneering work in the early 1980s (Cleveland and McGill, 1984; Cleveland, 1985), data graphics fell almost exclusively into the domain of graphic design practitioners (e.g., Tufte, 1983). Since that time the empirical study of graphical perception has led to a much deeper understanding of how we extract information from abstract representations and how best to design data displays.

Painting (or brushing) of multivariate data displays (McDonald, 1982; Becker and Cleveland, 1984; Carr and Nicholson, 1988) is a dynamic data graphic technique in which points in a "follower" plot are acted upon (e.g., highlighted) by moving a cursor around an "active" plot which typically displays a different set of variables for the same cases. The goal of painting is to help the data analyst discern relationships between variables shown in different plots. Usually this means making a judgment about the distribution of highlights in the follower plot. Specifically, the data analyst wants to determine if the highlights in the follower plot are uniformly distributed over all the observations in the (follower) plot. If there is some pattern in the follower plot (i.e., the highlights are not uniformly distributed over all the observations in the plot), then we have found some association between the variables shown in the different plots.

This selective highlighting of a subset of points in a point cloud is an especially useful technique in the visual exploration of multivariate data.
But, as Stuetzle (1987, 1988) has observed, this task of assessing whether the highlights are uniformly distributed over all observations is a particularly difficult one, even for quite experienced users. The viewer must in essence visualize two separate "population" distributions based on two separate "samples" (the highlighted and the downlighted points) and make a judgment about the identity of those distributions.

**Methods and Experimental Design**

We were curious about some of the graphical perception issues surrounding this highlighting distribution judgment task. In particular, we were interested in characterizing some of the factors which might influence the degree of task difficulty:

- Does plotting symbol type affect accuracy in judging the distribution of highlights in a point cloud?
- Do certain "macro" features of point clouds affect accuracy on this task? In particular, what are the effects of overall point density (total number of points) and overall highlighting level (total percentage of the cloud highlighted)?
- Do these point cloud features interact with plotting symbol type (i.e., do plotting symbols affect task performance differently at different levels of point cloud density and highlighting percentage?)
- If there is a plotting symbol effect, can we identify the critical features responsible for this effect (e.g., apparent 3-dimensionality, symbol discriminability, "ink" differences between highlight and downlight symbols)?

**Judgment Scale**

One approach to constructing stimulus displays might be to use actual data sets from various research programs in other domains. However, we chose to simplify the task: rather than judging the highlighting uniformity of a single cloud, subjects were asked to compare the relative highlighting levels of two adjacent point clouds (each having highlights uniformly distributed over the total cloud, but differing in their total highlighting proportions). A measure of highlighting balance across two point clouds was devised which accounted for both the overall level of highlighting and the highlighting difference between the two clouds.
Point Cloud Generation

The point cloud generator developed for Jeepers consists of three cards. One card does the actual stack production, while two auxiliary cards (1) provide a database of a variety of plotting symbols, and (2) produce point coordinate files for use by the display stack generator. The latter function currently permits sampling from either a uniform or a gaussian distribution. The card prompts for the sample size and samples randomly from a unit circle whose diameter is about one third of the card width. To allow us to manipulate the shape of the point clouds the data point generator was designed to accept independent variance values (relative to the unit circle) for the two dimensions.

For this experiment we generated both the downlight and the highlight data points for each cloud from a 2-dimensional gaussian distribution. Given the constraints of the display window shape and size and our decision to present two adjacent yet distinct point clouds, we chose to generate vertically elongated clouds with a Y variance twice as large as the X variance.

![Plotting Symbols](image)

Figure 3. Experimental Plotting Symbols. Each highlighted point cloud display was plotted using one of six sets of symbol pairs. The highlight symbol is shown on the right for each pair. Symbol sets were designated as (a)"smallDots", (b)"bigDots", (c)"2DCircles", (d)"3DCircles", (e)"slants", and (f)"arrows".

Plotting Symbols

The six pairs of plotting symbols selected for testing are shown in Figure 3. While largely exploratory, our choices were designed to address
several possible hypotheses regarding symbol effectiveness. The "small
dots" and "big dots" pairs represent two sizes of "traditional" scatterplot
symbol. The circular symbols introduce both 3-dimensionality and a larger
mapping space per data point. Finally, the "slants" and "arrows" symbols
were selected because they represent a class of symbols which have intuitive
appeal when viewed in isolation, but which may be less effective in
aggregate displays.

Stimulus Displays

With the experimental questions and some additional practical
constraints in mind, we developed a carefully balanced scheme for the
generation of stimulus displays. Thirty displays were produced, five for each
of the six plotting symbol pairs. Each symbol was used to plot four displays
for all combinations of the primary point cloud parameters of interest ("high"
and "low" overall cloud density and "high" and "low" overall highlighting
proportion), plus a more extreme "obvious" display. Two different
presentation orders were randomly generated for the 30 displays. These were
presented to two equivalent groups of 10 subjects.

Figure 4 shows a typical stimulus display. The small box at the top
center of the card indicates the "highlight" symbol. For each display subjects
answered the question "Which point cloud had the higher percentage of its
points highlighted?" by positioning a slider bar along an on-screen scale
anchored at the ends by the statements "Left side percentage much higher"
and "Right side percentage much higher".

Subjects

Twenty volunteer subjects were recruited from the departments of
Computer Science and Statistics at the University of Washington. Two were
faculty members, three were non-technical staff members, one was a senior
undergraduate student, and the remainder were graduate students. All used
computers on a daily basis and were familiar with mouse input and graphical
user interfaces.
Procedure

Each subject was instructed to sit at one of three Macintosh IIs and to wear a set of headphones connected to the computer. They were then told to begin by double-clicking on a desktop icon labeled "JeepersLink" and to follow the on-screen instructions. Jeepers then guided each subject through the experimental protocol (without any intervention by the experimenter). A few subjects were simply told to go to the computer and "run themselves"; there were no reported problems using this approach.

Figure 4. A Sample Stimulus Display.

Subjects were trained to use a mouse-controlled on-screen slider bar scale, followed by more specific training in the experimental task. They were given the option of repeating this training until they felt comfortable with the task. Summary data on training performance were automatically recorded.
Each stimulus was displayed for 2 seconds, followed by presentation of the slider bar scale. Stimulus displays were shown in three sets of 10 during the experimental trials, with a short break after the 10th and 20th displays. During these breaks the subject was invited to relax for a moment and the statement of the task was reiterated. Following the last experimental trial, subjects were asked to provide some demographic information (e.g., age, sex, department). Subjects in the second group were also asked to rate how difficult it had been to perform the task with each of the symbol sets.

**Summary of Findings**

The results of this study are presented elsewhere (Weghorst, 1989) in much more detail than is possible here. In summary, those findings suggested that:

- Some plotting symbols elicit more accurate judgments of point cloud highlighting balance than do others. Specifically, the "arrows", "slants", and "bigDots" symbols elicited judgments which were significantly less accurate than those elicited by the other three symbols (relative to the judgment scale which subjects were instructed to use).

- More conservatively, the "arrows" symbols elicited less accurate judgments than did the "smallDots" and the "bigDots" symbols (relative to each subject's "internal" scale as estimated from the subject's responses).

- Subjective ratings of task difficulty for the various plotting symbols essentially reflected performance accuracy, with the exception of the "3DCircles" displays which were perceived as being easier to judge than they actually were.

- Judgment accuracy was **not** affected by overall point cloud density, but **was** affected by overall highlighting level; displays having relatively **low** levels of highlighting overall elicited less accurate judgments than did those having relatively **high** levels overall.

- Judgment errors were distributed differently for "good" and "bad" symbols, with "bad" symbol errors apparently biased toward both the center of the judgment scale and the side of the double point cloud with the higher point density.
The distinction between "good" and "bad" plotting symbols for this perceptual task can be seen in Figure 5. For the displays using the "good" symbols ("3DCircles", "2DCircles", and "smallDots") the actual highlighting balance (according to our scale) is highly predictive of subjects' estimates (N=300). However, for the displays using the "bad" symbols ("arrows", "slants", and "bigDots") there is no association between actual balance and subjects' estimates (N=300). Research in the areas of graphic design (Bertin, 1983) and visual perception (Ramachandran, 1988) offers some insight into possible determinants of symbol "goodness" in this context.

These results have several implications for the design of data exploration systems; one may be the need for user control over plotting symbol parameters. At another level, it is clear that trying to convey such a complex relationship using a static display places a heavy perceptual burden on the data analyst. One of the challenges in software design for exploratory data analysis is to implement avenues for meaningful and productive interaction with data.
Range of Applicability

Within the confines of its structured protocol, Jeepers is well suited to a wide range of interface perception research problems. Although the displays used in the painting study were confined to the standard black-and-white HyperCard window, the range of Jeepers stimulus displays can be expanded. XCMDs are available for displaying color PICT files from within a HyperCard script, for example. And HyperCard "clones" are now available which include both color and multiple resizable windows, and which promise to execute HyperTalk code with little modification.

A more general solution extends the range of stimulus displays even further, but requires HyperTalk coding to implement. The displays used in the current study were static and non-interactive; each was presented for a fixed period of time under programmatic control. Although the session management facility is structured so that the display manager card controls the flow of events, the availability of the "openCard" handler on each display card allows us to construct cards which "present themselves" according to an internal script. Such a script may include various dynamic features, such as sounds, animated graphics, and even serial port control of external devices. It may also include the sub-launching of other applications; control is relinquished to the application and returns to HyperCard when the application terminates. Although user interaction with the sub-launched application is not captured, subjects may be interrogated as usual upon return from these "displays".

Conclusions

The user interface research tool described here has, of course, both strengths and weaknesses. By making it easier to conduct interface research, Jeepers also makes it easier to conduct bad interface research; this potential for misguided application may need to be addressed if Jeepers is to become an effective research aid. A second major weakness is its lack of generality, both across research protocols and across computer systems. The latter limitation may soon be mitigated by the emergence of various "stackware" readers and
generators on other systems. The former limitation is basically a design decision. Since research is a dynamic domain, it is difficult to conceive of a tool which would serve all interface research purposes; we elected to focus on a particular paradigm, and view Jeepers as one of many potential tools in the interface researcher's arsenal.

Jeepers greatest strength, on the other hand, is that it allows us to delegate many of the more routine and tedious research tasks to the computer and to allocate more of our time and energy to those activities which require human thought and intervention. By doing so, it not only controls for some potential sources of error variance, but it also supports a more exploratory mode of research, making it easier to follow up on hypotheses derived from experimental data.

A second advantage of automating these tasks is that it may encourage empirical research by individuals who might otherwise not attempt it. Indeed one of our primary goals in developing Jeepers was to make user interface research easier for those in a position to influence user interface design: namely, interface specialists and programmers.

References


