Interpreting Eye Movements with Process Models

Dario D. Salvucci
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
+1 412 268 8102
dario+@cs.cmu.edu

ABSTRACT

Though eye movements provide a wealth of information about how humans interact with computers, the analysis of eye movement data can be extremely tedious and time-consuming. This paper outlines an automated approach to tracing eye movements, that is, interpreting eye movement protocols based on an underlying process model. The proposed tracing methods utilize techniques such as hidden Markov models to relate observed eye movement protocols to the predictions of the process model. These methods have been applied successfully in the domain of equation solving and will be extended to several other task domains.

Keywords

Eye movements, process models, trace-based analysis, hidden Markov models, ACT-R

INTRODUCTION

In fields such as human-computer interaction and the cognitive sciences, protocol analysis is a widely popular and successful method of inferring how humans reason in task situations. Protocols—sequences of actions recorded during the execution of some task—help researchers determine the cognitive strategies involved in performing a task. Most protocol studies to date have emphasized verbal reports or manual actions. However, due to improved eye-tracking equipment and better understanding of how to exploit it, eye movement protocols have played an increasingly significant role in the study of human behavior. Eye movements can assist in answering questions such as:

- How and when do computer users encode information?
- What on-screen information do users ignore?
- How do users interleave encoding and computation?
- How do users' encoding strategies evolve with practice?

Difficulties with Analyzing Eye Movements

Though eye movement protocols are extremely informative, they are also very time-consuming and tedious to analyze. Like verbal protocols, several trials of even a simple task can generate massive amounts of data, all of which must be coded into some more manageable form for analysis. In addition, eye movement protocols typically include a great deal of noise due to individual and equipment variability. For large eye movement datasets with hundreds or thousands of trial protocols, it is simply implausible for humans to code the data consistently, accurately, and in a reasonable time frame. Thus, computer tools that assist protocol interpretation allow investigators to analyze larger, more complex datasets in a detailed manner that would otherwise be impossible.

Trace-Based Protocol Analysis

This work centers on a rigorous form of protocol analysis, trace-based protocol analysis (TBPA), which uses sequential protocol data to test and refine the predictions of process models [5]. TBPA has proven useful in a variety of contexts. Builders of intelligent tutoring systems have utilized a type of TBPA called model tracing to determine the user's solution path through a student model of the domain [1]. Researchers in human-computer interaction have employed TBPA to study the fits of user models [3]. Cognitive scientists have exploited TBPA to analyze verbal protocols in various problem-solving domains [4].

This paper describes a novel approach to the trace-based analysis of eye movement protocols. The extension of TBPA to eye movements would benefit all the areas above. In human-computer interaction, TBPA of eye movements would provide important low-level information on users' encoding strategies, and could help interfaces with eye-based input devices to determine user intentions. In intelligent tutoring systems, eye movements would help disambiguate problem-solving strategies which cannot be inferred solely from the student's mouse clicks and key-presses. In the cognitive sciences, TBPA of eye movements would assist in prototyping and evaluating low-level models of visual attention using large sets of eye movement protocols.

TRACING EYE MOVEMENT PROTOCOLS

The central problem of this work is the *tracing* of eye movement protocols, that is, relating a sequence of observed eye movements to one of the action sequences predicted by an underlying process model. The proposed approach includes a toolkit of tracing techniques which provide robust analysis in the presence of noise, allow for non-deterministic process models, and are extendible to other types of data. Due to space constraints, we discuss only the most sophisticated of these tracing techniques: point tracing.

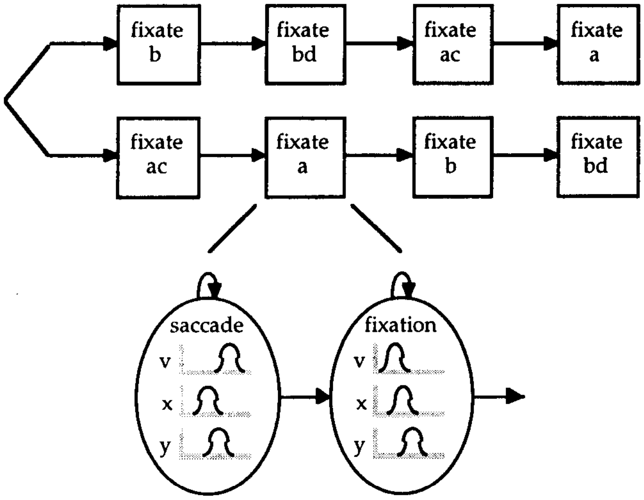

Figure 1: Point tracing hidden Markov model. The sequences of fixation submodels (squares) represent the process model's predicted action sequences. Each fixation submodel includes two states (ovals), a high-velocity saccade state and a low-velocity fixation state.

Point tracing uses hidden Markov models (HMMs) to map raw protocols directly onto model sequences. HMMs are probabilistic finite state machines that have been used effectively in such areas as speech and handwriting recognition. Point tracing using HMMs begins with the creation of an HMM submodel for each action predicted by the process model, such as fixating an item. The submodel represents a model of the data points produced when the intended action is executed; for instance, the fixation submodel might contain a high-velocity saccade (rapid eye movement) state linked to a low-velocity fixation state. Next, we string the action submodels into sequences representing the predicted model sequences, forming a composite model HMM, as shown in Figure 1. This composite HMM can then be used to decode the most probable state sequence for an observed eye movement protocol, thus mapping the point data to one of the model's predicted sequences.

For illustration, consider the eye movement protocol in Figure 2. In this task, students solved equations of the form *b x / ac = bd / a* by computing *cd*. Each point in the protocol represents a single sampling point (at a frequency of 8 ms); smaller points have a lower velocity, and lighter points occur later in time. Though the protocol is somewhat noisy, we can see that this student skipped the *x* and the operators and fixated only the numbers. We can interpret this protocol as the subject first fixating *b* and *bd* to compute *d*, then fixating *ac* and *a* to compute *c*, and finally multiplying *c* and *d* to reach the answer; we can call this strategy the *b-bd-ac-a* strategy. (Note that this student had already seen many of these problems, and thus had learned a very efficient strategy for solving them.) Assuming that our process model can generate such a strategy, point tracing would automatically arrive at this interpretation, computing it as the most probable classification for the observed data.

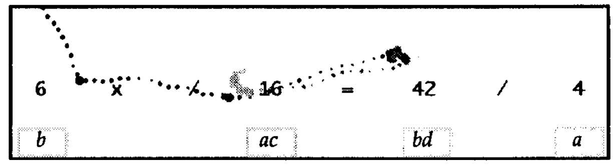

Figure 2: Sample equation solving protocol.

APPLICATION TO TASK DOMAINS

The proposed tracing techniques will be applied and evaluated in several task domains, including equation solving, reading, and eye typing. Thus far, we have performed a small-scale evaluation in the equation solving domain described above, and the results are promising. We first examined single trial protocols in an effort to deduce and identify subject strategies. The identified strategies were then used as a basis for a process model, implemented as an ACT-R production system [2]. To determine whether the tracing methods deduced sensible interpretations of the protocols, we evaluated each method with respect to human coding. In summary, two simpler tracing techniques produced 55% and 79% agreement with a human coder, and the point tracing technique produced 91% agreement. Considering that the automated interpretation required far less time than human coding, with the advantage of guaranteed consistency, these results are very encouraging.

ADDITIONAL WORK

In addition to presenting the tracing algorithms and their application to task domains, this work will address several other topics, including:

- reduction of eye movement protocols to gazes or fixations to facilitate strategy discovery
- visualization of raw and traced protocols to aid in process model discovery and evaluation
- extension of tracing techniques to other types of data, including mouse movements and key-presses

ACKNOWLEDGMENTS

This work was supported in part by an NSF Graduate Fellowship awarded to Dario Salvucci and ONR grant N00014-96-1-0491 awarded to John Anderson. I thank John Anderson for many helpful suggestions and Scott Douglass for assistance in data collection and analysis.

REFERENCES

1. Anderson, J. R., Boyle, C. F., Corbett, A. T., & Lewis, M. W. Cognitive modeling and intelligent tutoring. *Artificial Intelligence 42* (1990), 7-49.
2. Anderson, J. R., & Lebiere, C. *The Atomic Components of Thought.* Erlbaum, Hillsdale NJ, 1998.
3. Card, S., Moran, T., & Newell, A. *The Psychology of Human-Computer Interaction.* Erlbaum, Hillsdale NJ, 1983.
4. Newell, A., & Simon, H. A. *Human Problem Solving.* Prentice Hall, Englewood Cliffs NJ, 1972.
5. Ritter, F. E., & Larkin, J. H. Developing process models as summaries of HCI action sequences. *Human-Computer Interaction 9* (1994), 345-383.