
How Do We Read Algorithms?

A Case Study

Martha E. Crosby and Jan Stelovsky
University of Hawaii at Manoa

Despite a growing research interest in program comprehension, there is still much to learn about how individual differences in reading strategies influence comprehension of procedural text, such as algorithms. Textbooks usually depict algorithm definitions in either pseudocode or a programming language. Text is frequently accompanied by graphics to support understanding of an algorithm's behavior. Increasingly, computers use animated graphics to show dynamically how an algorithm works.

Traditionally, operating-system manuals and computer-related training materials are written with little regard to the reader. For example, keywords in programs are often printed in boldface type. This study suggests, however, that keywords are the least observed portions of a program's text.

Until recently, designers of programming languages relied primarily on their intuition to tell them what is "best" for programmers. Replacing instinct by in-depth knowledge regarding human understanding of algorithms can benefit many disciplines, from the writing of computer documentation to the design of programming and specification languages. The wide variety of individual styles docu-

Researchers want to understand how different reading strategies influence comprehension of procedural text. For example, does experience alter strategy and thus affect how or where we look?

mented in this study suggests that these materials would be more effective if they were adapted to reader preferences.

The Pascal version of an algorithm is considerably more succinct and therefore more complex than typical prose. Do we

read algorithms the way we read any other text, or does our reading strategy change to accommodate the complexity of procedural text? Graphical representations often contain text. Does this supplementary text draw as much attention as the graphics?

Procedural text incorporates areas of varying complexity. Highly formal code is accompanied by informal comments in English. Complexity varies even within the actual code; while keywords are predictable, Boolean expressions are terse yet replete with information. When viewing procedural text, do subjects prefer to concentrate on comments or code? Within the code, do complex areas attract more attention?

In addition to traditional methods used to analyze subjects' comprehension, monitoring of eye movement has emerged as a way to explore a subject's attention in unprecedented detail. The position and duration of eye fixations show where the subjects focus. Two opposing theories explain the relationship between eye movement and reading (see sidebar on eye movement). Nevertheless, research using prose suggests that analyzing individual scan patterns may shed new light on individual differences in reading and understanding algorithms.

Research shows that experience fosters comprehension (see sidebar on program comprehension). While we expected novices to have varying strategies, we hypothesized that with increasing expertise, strategies become more efficient and thus similar. Can we assume that experience promotes viewing strategies that expedite understanding? In particular, does experience influence the way our eyes scan material, or does it change the time we spend looking at important text areas? In other words, does experience effect *how* we look, or *where* we look?

The experiment

The purpose of this study was to explore the way subjects view an algorithm, written in Pascal, and the graphical representations of its behavior. We designed an experiment that examined the influence of programming experience on how subjects view a short but complex algorithm, and we analyzed the scan patterns of eye movement for subjects at two experience levels.

The experiment concentrated on the following questions:

- Is there a lag between viewing and processing complex text; that is, does the immediacy theory hold for reading algorithms?
- Is there a difference between reading simple text and complex text such as algorithms?
- Does a subject's experience influence a preference for comments versus code?
- Does experience influence a subject's focus on critical areas of the algorithm?
- Do the viewing patterns of experienced subjects differ from those of novices?

We assumed that reading algorithms would differ from reading prose because algorithms tend to be complex and are formulated on a high level of abstraction with little redundancy. We expected individual differences in reading strategies and hypothesized that experience would be the key factor—that reading strategies would become more efficient with increasing expertise.

We selected 19 volunteers randomly from the University of Hawaii's computer science program. The low-experience group consisted of 10 subjects from the second-semester, CS2, course. The high-experience group consisted of eight graduate students and one recent PhD faculty member. All subjects were familiar with Pascal. Only a sample of the subjects will be mentioned in our discussion of individual strategies; they will be called LA to LE (low-experience group) and HA to HC (high-experience group).

In a prior study we contrasted the comprehension of Pascal and English versions of 12 algorithms. In this study, we applied a fine-grain analysis using eye movement to one specific algorithm. The binary search was selected because it can be defined in a few lines of Pascal code, contains enough complexity, and can be expressed in a form that intrigues experienced programmers. During the semester, this algorithm was covered in CS2 lectures. In addition, two months prior to the experiment a laboratory session was devoted to presenting the algorithm's behavior using animated graphics.

Eye movement research

Since analysis of eye movements provides a wealth of data at a fine level of detail, it is particularly suited for the investigation of cognitive processes. Two theories link eye fixation data to comprehension. Bouma¹ proposes the existence of an internal buffer that stores the image viewed during an eye fixation. Since the image is processed later, the total fixation time per word cannot be directly related to comprehension. Just and Carpenter² devised a reading model based on immediacy and eye-mind assumptions. The text is interpreted immediately during an eye fixation, and the eye stays fixated on a word as long as necessary to process it. According to the latter theory, fixation duration is a measure of the subject's attention to a specific area.

The material used in most eye movement research is nonprocedural prose. A number of studies have shown that subjects' eye movements, fixation durations, and number of regressions are influenced by the complexity of the task. In addition, several studies re-

ported variability among subjects. For example, Kennedy and Murray³ note that regressions increase for complex text. Shebilske and Reid⁴ report differences in reading strategies between recreational and rigorous reading. They found that while poor readers scan text indiscriminately, good readers adjust their strategies to the task.

The prevalent opinion is that level of understanding is what affects the mechanics of reading. More recently, eye movement analysis has been applied to complex material. A study by Vonder Embse,⁵ using mathematical graphs, shows that experts use a small number of longer fixations and novices use a large number of short fixations. While novices do not discriminate among areas of the graphs, experts focus on key areas. Moreover, fixation durations are considerably longer than those reported for prose. This finding supports the immediacy theory for viewing complex material such as algorithms. Eye movement analysis of complex text can contribute to a deeper understanding of cognitive processes.

References

1. H. Bouma, "Visual Interference in the Parafoveal Recognition of Initial and Final Letters of Words," *Vision Research*, Vol. 13, 1973, pp. 767-782.
2. M. Just and P. Carpenter, "A Theory of Reading: From Eye Fixation to Comprehension," *Psychological Rev.*, Vol. 87, 1980, pp. 329-354.
3. A. Kennedy and W.S. Murray, "Inspection Times for Words in Syntactically Ambiguous Sentences Under Three Presentation Conditions," *J. Experimental Psychology: Human Perception and Performance*, Vol. 10, 1984, pp. 833-849.
4. W. Shebilske and L. Reid, "Reading Eye Movements, Macrostructure and Comprehension Processes," in *Processing of Visible Language*, Vol. 1, P. Kolars, M. Wrolstad, and H. Bouma, eds., Plenum Press, New York, 1979.
5. C. Vonder Embse, *An Eye Fixation Study of Time Factors Comparing Experts and Novices When Reading and Interpreting Mathematical Graphs*, UMI dissertation #8717747, Ohio State University, 1987.

The demonstration program visualizes search steps on a set of 15 numbers represented as a bar chart (Figure 1). As the numbers are ordered, the bars increase in height from left to right. The program provides another view when searching among 120 numbers (Figure 2). While the representation of 15 numbers can accommodate placing the numbers beneath the corresponding bars, the representation of 120 numbers uses bars only. In each step the algorithm narrows the subset of num-

bers that can contain the number to be found. This subset is represented graphically; the corresponding bars are enclosed in two brackets. While the bars remain static, one bracket is moved in each step. The choice of the bracket depends on a comparison between the number searched and the number in the middle of the enclosed subset. Again, the number searched is presented graphically as a highlighted bar. This bar is placed next to the bar in the middle of the enclosed set (to the left if it is

smaller, to the right if it is greater).

The program also incorporates a separate screen where students can see the algorithm written in Pascal. The slides used in the experiment were snapshots from the animated program demonstrating the algorithm's behavior. One snapshot was taken in the middle of the search process in both set sizes (Figures 1 and 2).

Typical program code contains portions of varying complexity. Longer Boolean expressions are comparable to compact

Program comprehension by novices and experts

Literature explaining the differences between novices and experts is profuse. In a landmark study, Chase and Simon¹ explored chess board pattern memorization. They found that experts and novices remember random patterns equally well, but expert players remember meaningful board configurations significantly better. In addition, research from psychology and linguistics suggests that the better text fits a meaningful context, the more likely it is to be assimilated.

Numerous theories explain performance differences in problem solving. For example, Lesgold et al.² found that while novices search longer for a solution, experts spend more time building up a representation of the problem. Therefore, experts are more likely to invoke a solution strategy appropriate to the problem. The ability to organize material in a meaningful way emerges as one of the distinguishing characteristics of an expert.

Novice-expert differences are particularly evident in computer programming. Expert programmers are more likely than novice programmers to find programs intelligible. In a variation of the Chase and Simon experiment, Shneiderman³ presented short computer segments in either shuffled or executable order to subjects with varying levels of expertise. Analogously, no significant difference was found in the recall of the shuffled program segments. Experienced subjects who understood the meaning of the statements in executable order recalled them better. Adelson⁴ suggests that experts remember programs differently; while novices group statements syntactically, experts rely on functionally or semantically equivalent statements.

Program understanding can be studied at different levels of detail, from eye fixations and memory recall to protocol

analysis. Shneiderman³ proposes a bottom-up, line-by-line model of program comprehension, where the programmer first deals with the syntactic aspects of the program and then advances to lower level semantic structures such as identifying statements and logical chunks. Alternatively, Brooks⁵ devised a top-down model of program comprehension based on a hypothesis about the program's function. Curtis et al.⁶ report large discrepancies in performance even among expert programmers with similar experience. Brooks' model attributes the disparities among subjects with similar experience to slight differences in training.

Brooks observed that programmers do not study programs line by line but search for key lines (beacons) to verify their hypothesis about a program's function. Weidenbeck⁷ found that this did not apply to novices. Weidenbeck and Scholtz⁸ suggest a causal link between beacons and comprehension, since comprehension by experienced programmers declines when beacons are disguised. Soloway and Ehrlich⁹ hypothesize that programmers have plans and rules that allow the chunking of related program segments. They found that experts do much better than novices on "plan-like" programs, while the differences diminish on "non-plan-like" programs.

Pennington¹⁰ outlines a model of program comprehension based on the theories of text comprehension posed by Van Dijk and Kintsch. Using protocol analysis, she found that professional programmers with a high level of comprehension use a cross-referencing strategy that alternates among systematic study of the program, its verification, and its application. Programmers with a low level of comprehension focus on either the program or its application, but not both.

References

1. W. Chase and H.A. Simon, "Perception in Chess," *Cognitive Psychology*, Vol. 4, 1974, pp. 55-81.
2. A. Lesgold et al., "Diagnosing X-Ray Pictures," in *The Nature of Expertise*, M. Chi, R. Glaser, and M. Farr, eds., Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1988, pp. 311-342.
3. B. Shneiderman, "Exploratory Experiments in Programmer Behavior," *Int'l J. Computer and Information Sciences*, Vol. 5, 1976, pp. 123-143.
4. B. Adelson, "Problem Solving and the Development of Abstract Categories in Programming," *Memory and Cognition*, Vol. 9, No. 4, Apr. 1981, pp. 422-433.
5. R. Brooks, "Towards a Theory of the Cognitive Processes in Computer Programming," *Int'l J. Man-Machine Studies*, Vol. 9, 1977, pp. 737-751.
6. B. Curtis et al., "Measuring the Psychological Complexity of Software Maintenance Tasks with Halstead and McCabe Metrics," *IEEE Trans. Software Eng.*, Vol. 5, No. 2, 1979, pp. 96-104.
7. S. Weidenbeck, "Cognitive Processes in Program Comprehension," in *Empirical Studies of Programmers*, E. Soloway and S. Iyengar, eds., Ablex, Norwood, N.J., 1986, pp. 48-57.
8. S. Weidenbeck and J. Scholtz, "Beacons: A Knowledge Structure in Program Comprehension," in *Designing and Using Human-Computer Interfaces and Knowledge-Based Systems*, G. Salvendy and M. Smith, eds., Elsevier Science, Amsterdam, 1989, pp. 82-87.
9. E. Soloway and K. Ehrlich, "Empirical Studies of Programming Knowledge," *IEEE Trans. Software Eng.*, Vol. 10, No. 5, 1984, pp. 595-609.
10. N. Pennington, "Comprehension Strategies in Programming," in *Empirical Studies of Programmers*, G. Olson, S.B. Sheppard, and E. Soloway, eds., Ablex, Norwood, N.J., 1987, pp. 100-113.

mathematical formulas, while keywords contain standard information. The complexity of comments also varies; while assertions and invariants are frequently stated as mathematical formulas, plain English is often used to describe the purpose of the code statements. Because we were interested in how complexity affects viewing patterns, the slide with the Pascal text was slightly different from the version of the binary search algorithm used in the demonstration program. The meaning of the statement "found := left = right + 2" is hard to grasp even for expert programmers (Figure 3). This alternative form, however, was explained in an earlier CS2 lecture.

The textual slide contained an error: The direction of the comparisons was reversed (" $\leq$ " in line 5 and " $\geq$ " in line 6). For programmers, this is not an unusual sight, as a substantial portion of their coding time is devoted to tracing errors.

Subjects were tested individually. The order of the algorithm slides was the same for all subjects: The Pascal code in Figure 3 was followed by the graphical slides in Figures 2 and 1. The subjects were told that they could view the slides for as long as they needed to understand and remember them. They indicated verbally when they were finished looking at the slide. At the beginning of the experiment, the subjects were given a comprehension pretest based on general knowledge of the binary search algorithm. This quiz was repeated after the text slide and at the end of the experiment. After the text slide, subjects were also asked to correct the Pascal code if necessary.

Cloze tests followed every slide. The Cloze procedure is a more convenient and objective measure of reading comprehension. The procedure requires every n th word (typically the fifth) to be replaced with blanks. Then, on the basis of their global understanding, the subjects must reconstruct the text. To determine whether the subjects memorized the slide or reconstructed the correct algorithm from prior knowledge, the reversed comparisons were left empty in the Cloze test. We plan to analyze the effect of errors on the viewing pattern of subjects who notice them.

The subjects' experience was the independent variable. The dependent variables were (1) fixation time, calculated either as the total time in seconds of all fixations in an area or as a percentage of the total time, and (2) number of fixations, calculated either as the total number of fixations in an area or as a percentage of the total number of fixations.

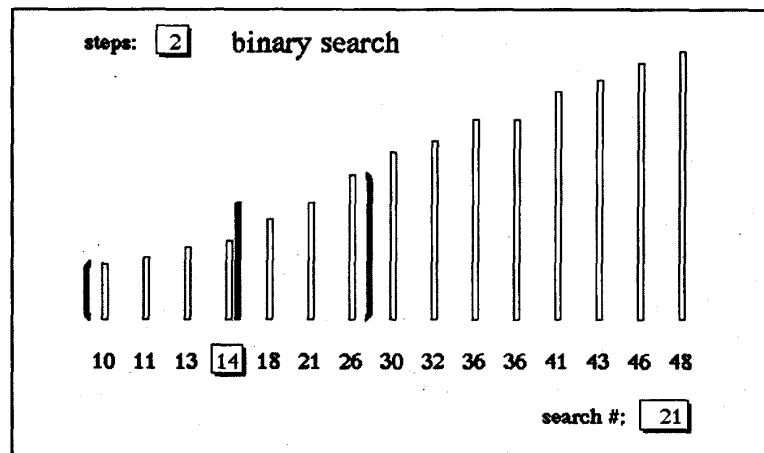


Figure 1. Graphics slide: "15 bars."

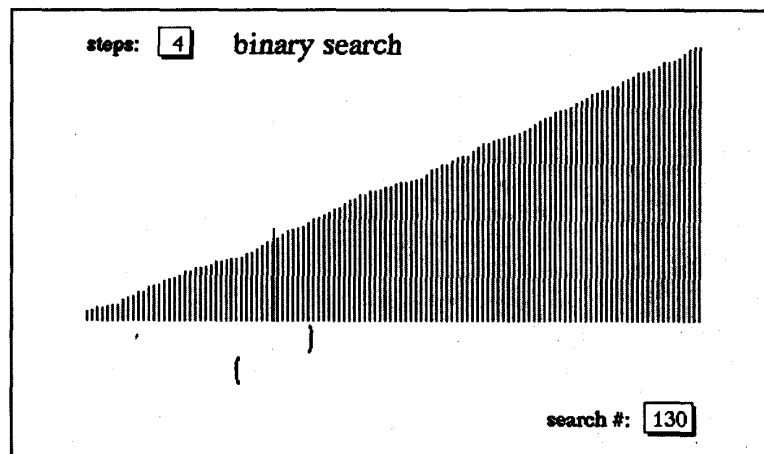


Figure 2. Graphics slide: "120 bars."

```

begin (*array A sorted, i.e. A[i] ≤ A[j] and left ≤ i < j ≤ right*)
  repeat (*A[left] ≤ x ≤ A[right] and left ≤ right *)
    (* choose middle to split A into 3 subarrays *)
    middle := (left + right) div 2;
    if A[middle] ≥ x then      left  := middle + 1;
    if A[middle] ≤ x then      right := middle - 1;
  until left > right;
  found := left = right + 2;
  index := middle;
end; (* BinSearch *)

```

Figure 3. Text slide with Pascal code.

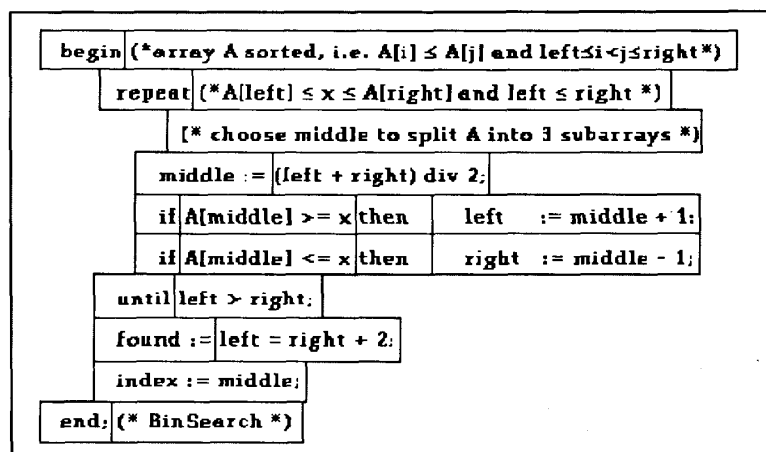


Figure 4. Subdivision of the text slide into areas.

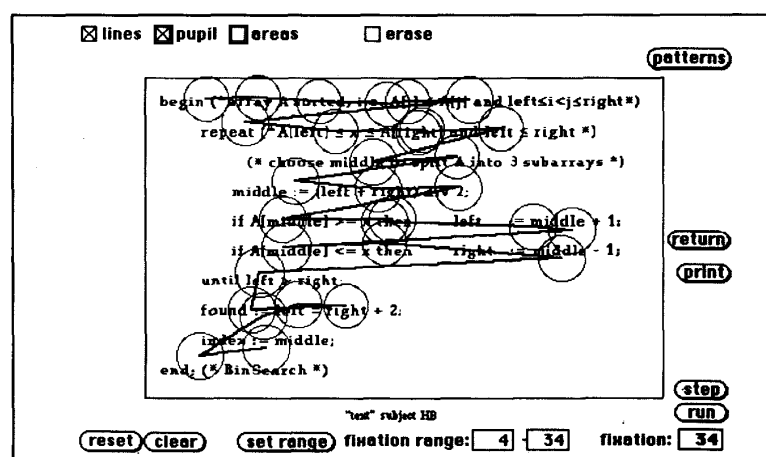


Figure 5. Partial sequence of HB's eye movements when reading text.

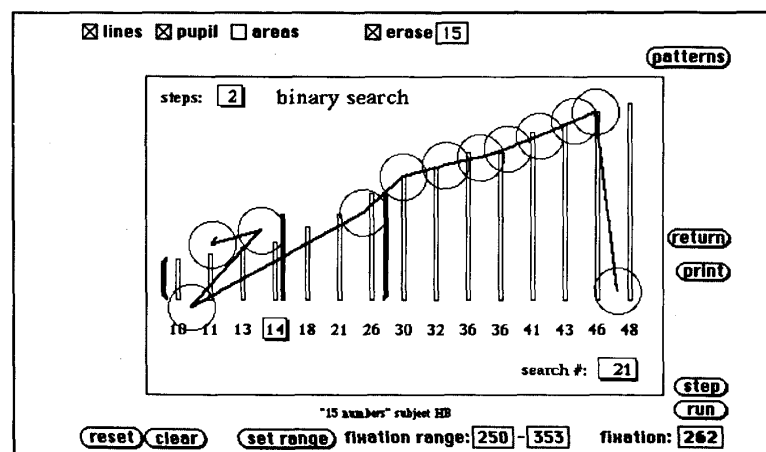


Figure 6. Partial sequence of HB's eye movements when viewing the graphic in Figure 1.

The experiment was performed at the eye-monitoring laboratory in the Department of Educational Psychology at the University of Hawaii. The equipment consisted of an Applied Science Laboratories eye movement monitor, controlled by a host computer. The host computer's software calculated the eye fixations from data generated by the eye movement monitor in real time.

Materials and analysis software were developed on Macintosh computers. This software for postexperimental analysis was designed to provide a highly visual, dynamic picture of the subjects' eye movements. Eye fixations are superimposed on a gray, scaled-down picture of the slide. Each fixation point is displayed in time proportional to its real duration. A fixation can be visualized in a variety of modes: as a corner of a path where lines connect subsequent fixation points; as a circle whose radius is proportional to the pupil size; or by highlighting the enclosing area.

Another feature of our analysis software allows the experimenter to partition the slide into areas that can be used for a more global analysis. Figure 4 shows how the text slide was subdivided into areas of varying complexity. Then, the total number of fixations (or total time) can be shown in each area. Early analysis of the experimental data indicated that some subjects apply periodically similar scanning patterns. To visualize these patterns, the software was enhanced to produce two-dimensional graphs showing the sequence of areas in time.

Results

Immediacy theory and procedural versus simple text. According to the immediacy theory, the total number of fixations is a measure of the relative attention subjects devote to each area (see sidebar on eye movement). Figure 5 shows a typical left-to-right, top-to-bottom reading pattern. Notice how closely the individual eye fixations match the centers of the relevant textual areas. This correspondence strongly suggests the validity of the immediacy theory, at least for procedural text. The close fit of material and eye fixations, however, is not limited to text, as the viewing strategy in Figure 6 shows. The correspondence of eye fixations and material viewed was not confined to isolated cases, but was observed consistently throughout the entire experiment. On the

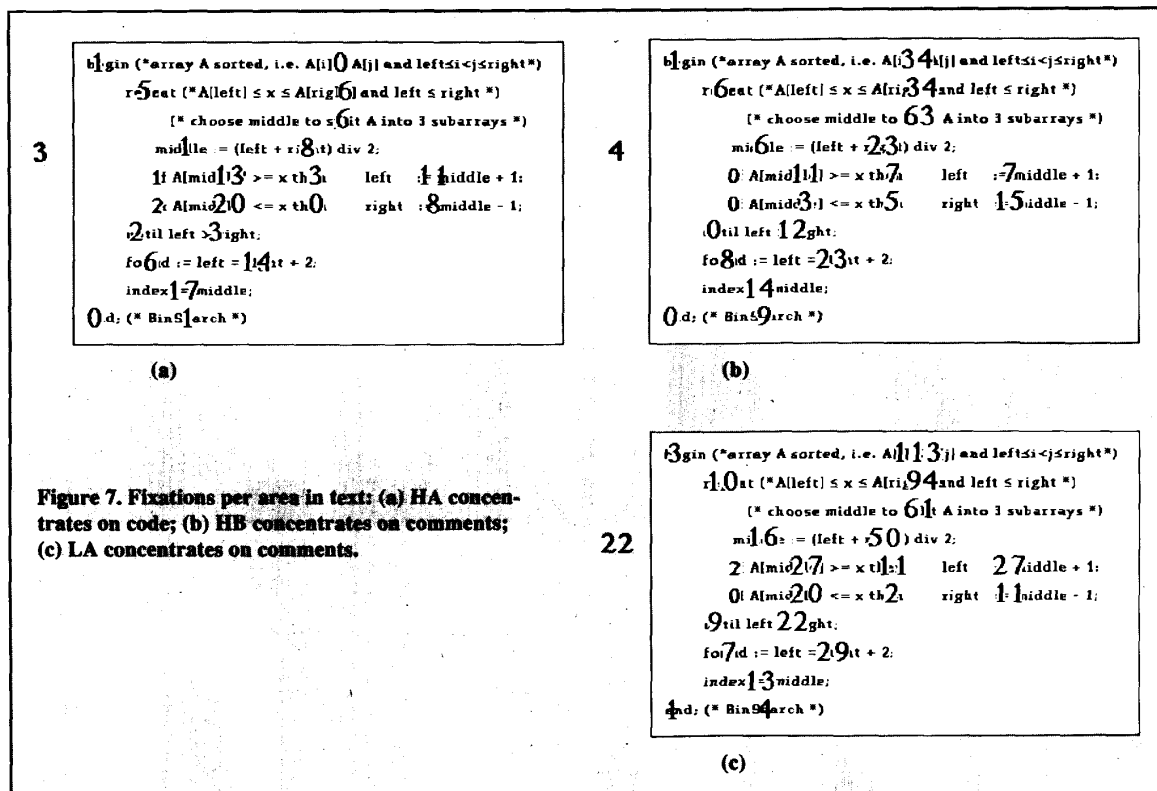


Figure 7. Fixations per area in text: (a) HA concentrates on code; (b) HB concentrates on comments; (c) LA concentrates on comments.

basis of the correctness of the immediacy theory, we can interpret an eye fixation as a measure of attention devoted to the surrounding area.

Notice that Figures 5 and 6 are only snapshots demonstrating how our analysis software can replay a portion of a subject's eye movement sequence (in this case fixations 4 to 34 and 250 to 353) superimposed on the slide displayed in the inner rectangle. As we will show in the subsequent figures, all subjects needed to view important text areas numerous times. The number of eye fixation regressions in our experiment is far greater than the number reported in studies using nonprocedural prose.

Comments versus code. The text of a computer program contains two primary types of information: code to be executed and comments that contribute to its understanding. Figures that show the total number of fixations per area (Figure 7) superimposed on each of the areas defined in Figure 4 clearly distinguish code-oriented and comment-oriented subjects. (Fixations outside all areas are displayed outside the border.)

Analysis of subjects' attention showed a wide range of focus distributions regardless of experience level. For example, subjects HA and HB from the high-experience group showed opposite tendencies. While HA (Figure 7a) concentrated primarily on code, HB (Figure 7b) used the

comments extensively. Similarly, we found in the low-experience group both code-oriented and comment-oriented subjects. Subject LA (Figure 7c), for instance, demonstrated a distribution comparable to that of HB.

Figure 7 also clearly demonstrates that

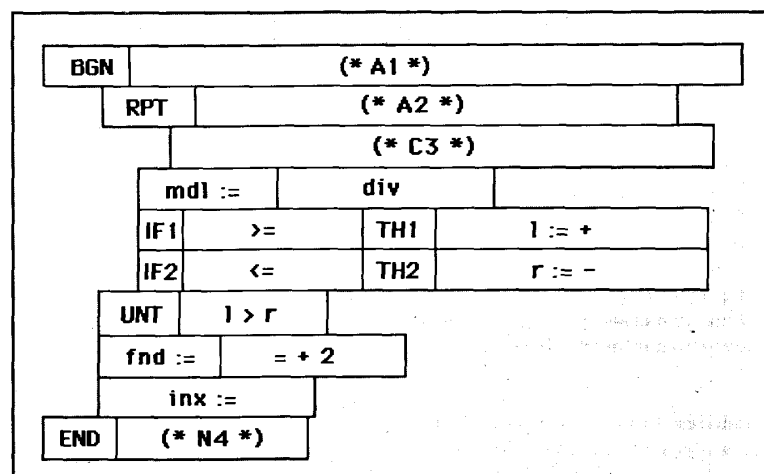
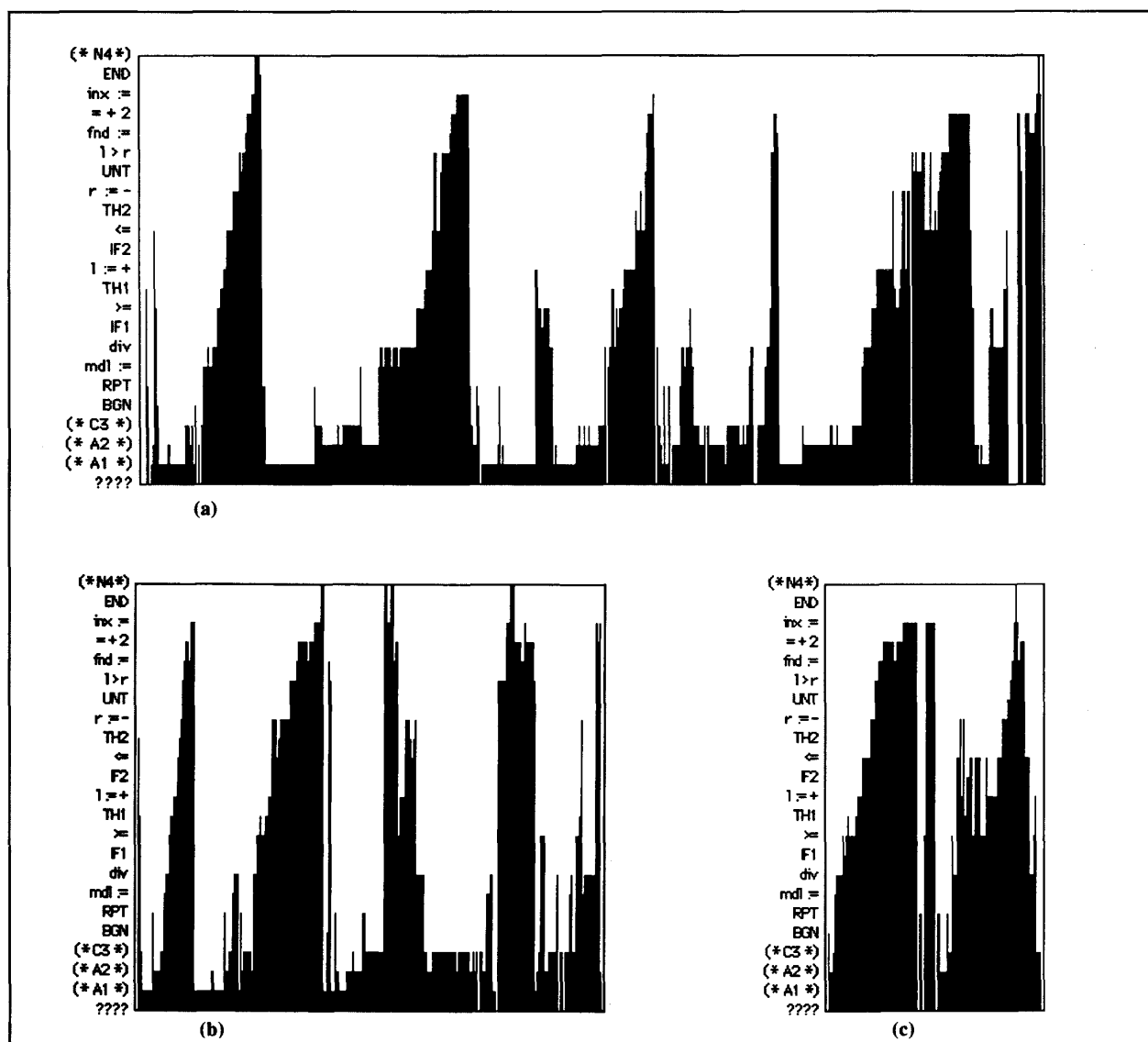


Figure 8. Labels of subareas in the text slide.



these subjects needed many eye fixations in most of the algorithm's areas.

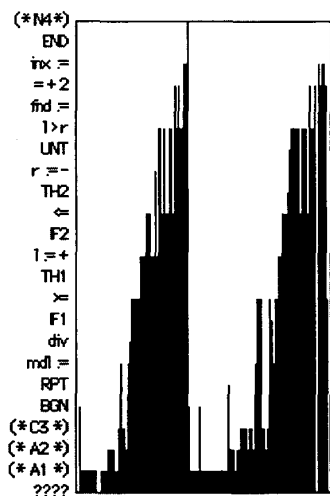
To compare the time devoted to comments by subjects in the two groups, we determined the percentage of time spent viewing comments. While the low-experience group averaged 44 percent of the time viewing comments, the high-experience group viewed them only 35 percent of the time.

Individual viewing strategies. For further analysis of individual viewing patterns, we developed "area/fixation graphs" that depict the sequence of viewed areas. In

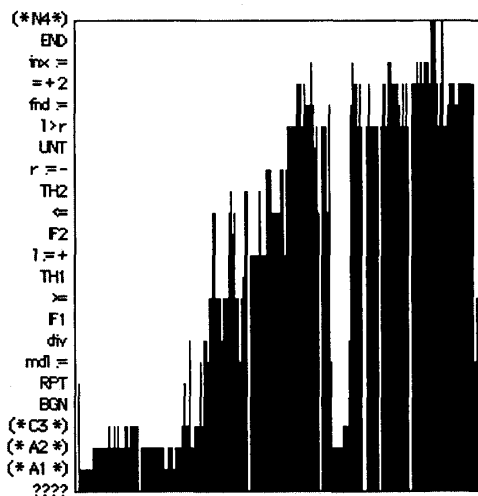
these graphs the horizontal axis represents the sequence of fixations (one fixation is a unit) and is therefore closely related to time. The vertical axis represents areas. Figure 8 assigns each area a label, shown on the vertical axis of the graph. (* A1 *) and (* A2 *), for instance, represent the assertion comments in the first two lines, (* C3 *) the descriptive comment on the third line, *BGN* and *RPT* the keywords "begin" and "repeat" on lines 1 and 2, *mdl* and *div* the two parts of the statement "middle := (right + left) div 2," and ??? means outside (compare to Figure 4).

The dynamic evaluation of subjects' eye

movement while looking at the text slide revealed distinct periodic scanning patterns. The area/fixation graphs depicting the viewing sequence confirmed this finding. Of course, the order of areas, which can be chosen arbitrarily, determines the visual impact of these analysis graphs. Notice that except for the comment areas, we chose an order that reflects the "natural" left-to-right, top-to-bottom flow of text (see Figure 5). Comments were grouped to highlight strategies that compare them with corresponding code. This sequence of areas produced the best contrast.

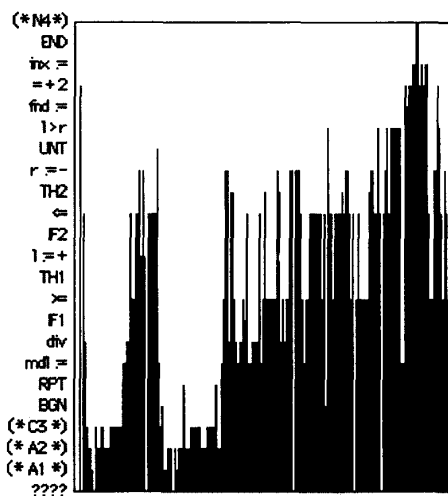


(d)



(e)

Figure 9. Text-scanning patterns: (a) LA uses a multiple-scan strategy; (b) HB's multiple-scan strategy; (c) HA uses a two-scan strategy; (d) LB's two-scan strategy; (e) LC uses one scan; (f) LD's comparative strategy.



(f)

Let's look at individual viewing strategies. In Figure 9a subject LA reveals a periodic "multiple scan" viewing strategy. The monotonic increasing peak of the first scan (the length of the bars increases steadily) shows a regular left-to-right, top-to-bottom perusal of the text. The subsequent low, flat area signifies concentration on comments. The next scan maintains the natural reading order but, more importantly, begins to show some selectivity and stress on individual statements. Notice, for instance, the area "(right + left) div 2," labeled as *div*. The tendency toward selectivity is particularly visible in the last scan.

Although the order is still maintained, this scan shows a distinctly comparative approach (short bars follow long ones and vice versa).

Subject HB (Figure 9b) exhibits a pattern strikingly similar to that of LA. Again, the first linear scan is followed by a second scan that concentrates more on specific areas. The second half of the graph shows attention to comments interrupted by comparative excursions into the code.

Subject HA efficiently used two distinct scan periods (Figure 9c). Little attention was paid to comments (only a few bars end at the heights (*A1*), (*A2*), and

(*C3*)). During the first scan more attention was devoted to individual areas. The second scan shows a more comparative approach.

Subject LB is an extreme case of the "two scan" strategy; the scans are virtually identical (Figure 9d).

Multiple scanning patterns were not used exclusively, however. Subject LC used a "single scan" strategy that concentrated on individual areas and exhibited few comparisons with neighboring areas (Figure 9e).

Subject LD used an extreme case of a "comparative" strategy (Figure 9f). The

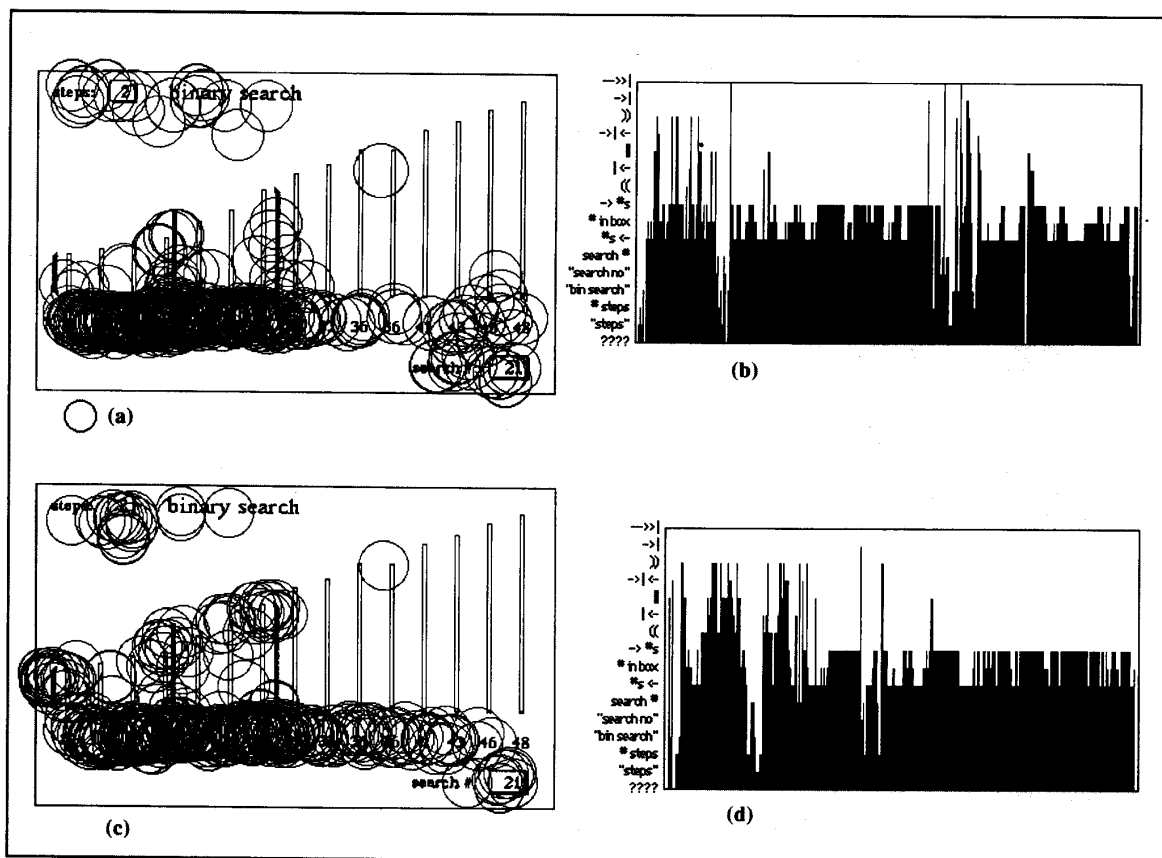


Figure 10. For the graphic in Figure 1 ("15 bars"), both LE and HC concentrate on numbers: (a) LE's eye movements; (b) LE's scanning pattern; (c) HC's eye movements; (d) HC's scanning pattern.

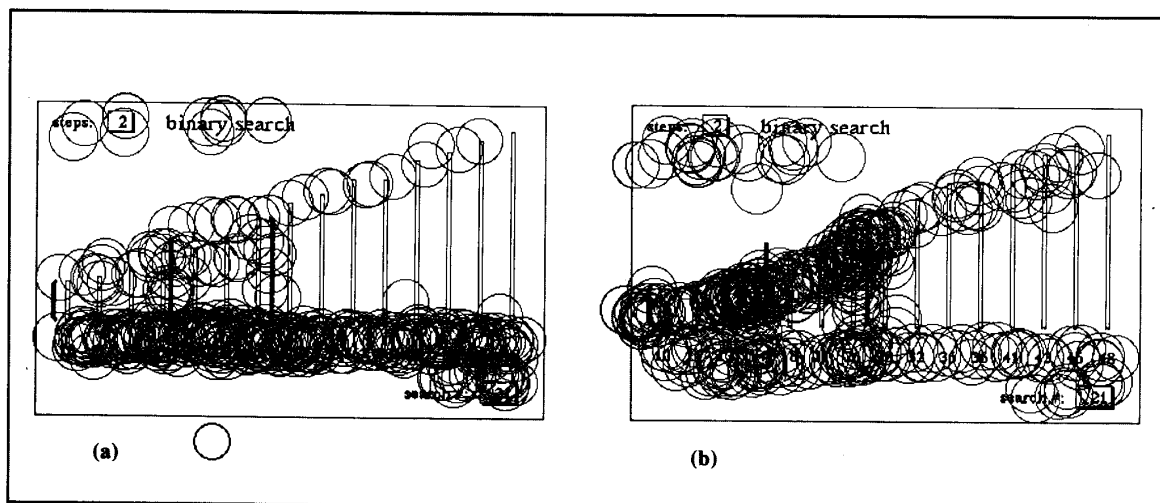


Figure 11. "15 bars." (a) HB concentrates on numbers and scans the tops of bars; (b) LA's pattern is similar to HB's.

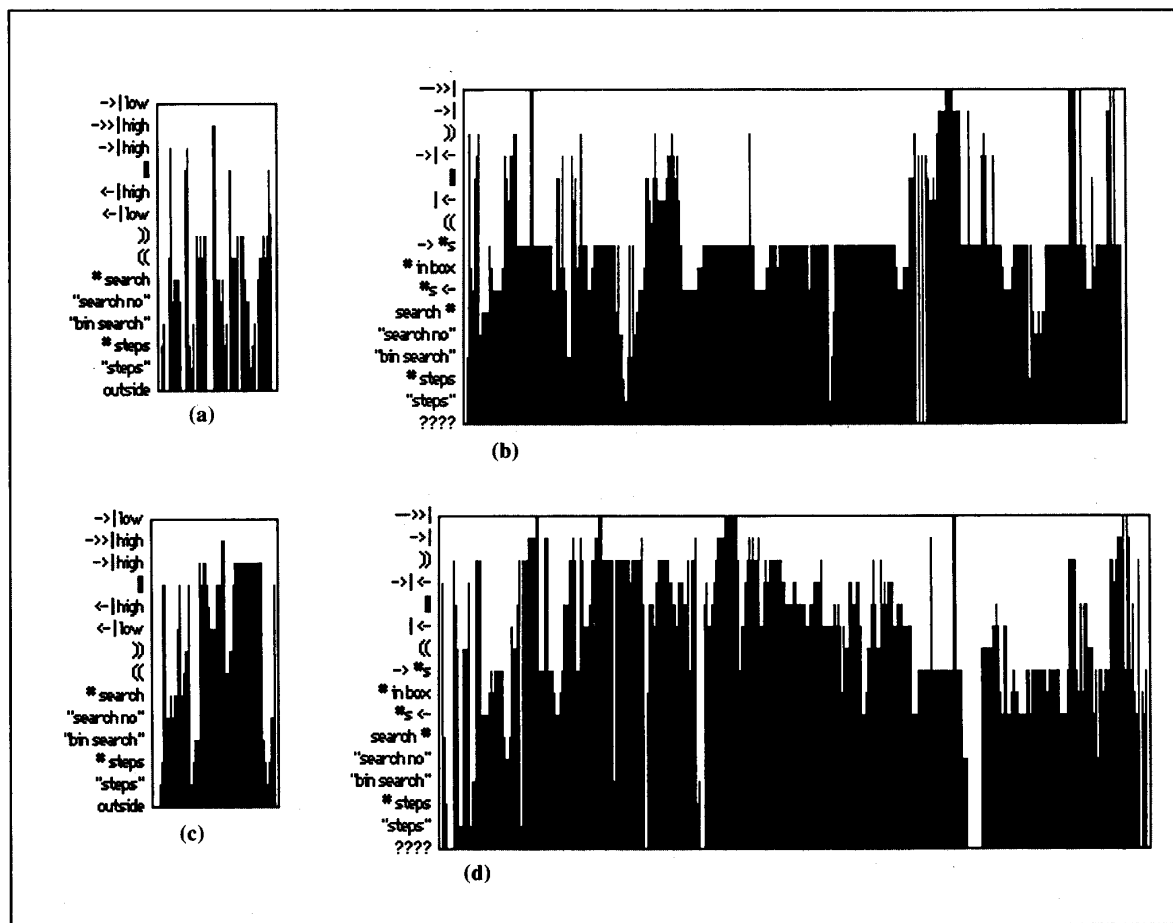


Figure 12. Both HB and LA used five times as many fixations for "15 bars," which contains 17 number elements, as they did for "120 bars," which contains only two number elements. (a) HB's scanning pattern for "120 bars"; (b) HB's scanning pattern for "15 bars"; (c) LA's scanning pattern for "120 bars"; (d) LA's scanning pattern for "15 bars."

natural order was no longer predominant, as comparison became the overriding scheme. Nevertheless, this strategy was purposeful rather than random, since the subject's test scores demonstrated a perfect understanding.

Strategies for viewing graphics. Besides area/fixation graphs, analysis of graphical scenes used pupil concentration patterns showing all the fixations, each represented as a circle superimposed on the original slide.

The slides were partitioned into areas according to their prominence (for example, highlighting) and their relevance to the interpretation of the algorithm. A natural order for graphical areas is not obvious. After replaying eye movements and analyzing total pupil concentration patterns (for example, Figure 10a), we arranged the

areas in the area/fixation charts in three groups: text related, numbers, and bars. Within each group the areas were ordered top to bottom, left to right.

Analysis of total pupil concentration patterns and area/fixation graphs revealed that while most subjects barely looked at the numbers, a few concentrated on them.

The one strategy common to both experience groups is represented by subjects LE (Figures 10a and 10b) and HC (Figures 10c and 10d). Both subjects concentrated on the numbers, particularly those under the highlighted graphical elements. These subjects were not interested in the tops of the bars on the right.

The similarity between scanning patterns is further exemplified in the area/fixation graphs (see Figures 10b and 10d). Both subjects first perused all areas of the picture and then concentrated on the num-

bers (areas #s <-, # in box, and -> #s). While HC's overview phase took longer and was followed by exclusive attention to the numbers, LE interrupted the number scan for another general overview.

Let's compare these strategies with those of subjects HB and LA (Figure 11). HB spent more time viewing numbers than did LA. Nevertheless, just as with the textual presentation, there are similarities in HB's and LA's scanning strategies for the graphical presentation. Unlike subjects LE and HC, both subjects dedicated much attention to a careful scan of the tops of the bars on the right. While LE and HC concentrated on numbers below the highlighted bar, HB's and LA's attention to individual numbers was uniform.

As the corresponding area/fixation graphs demonstrate (Figure 12), subject LA put more emphasis on the areas be-

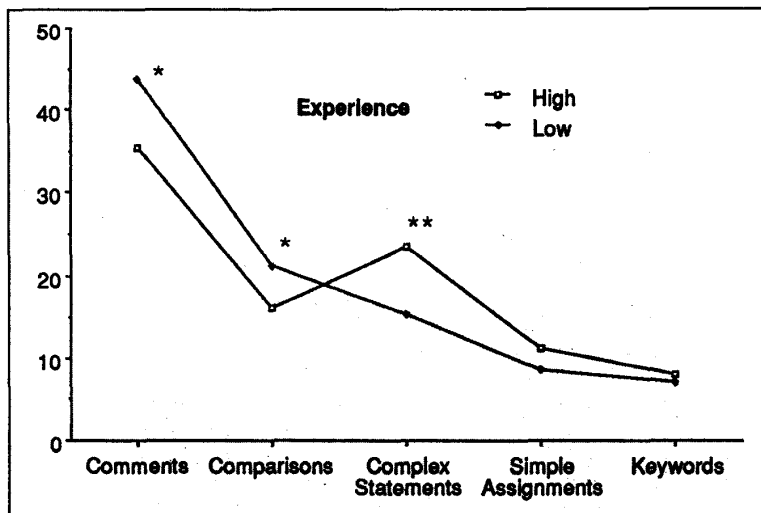


Figure 13. Algorithm areas viewed by high- and low-experience groups.

tween highlighted graphic elements than did HB, who exhibited a very consistent pattern when viewing numbers.

As the juxtaposition of Figures 12a and 12b with Figures 12c and 12d shows, subjects HB and LA needed almost the same number of fixations for the two graphic slides. They both briefly perused the picture "120 bars," which contains only two number elements. Then they used five times as many fixations for the picture "15 bars," which contains 17 numbers (notice the 1:5 ratio in the length of Figure 12a versus 12b and Figure 12c versus 12d).

Experience and focus on critical areas. Program code contains areas of varying complexity and importance. Portions of text in the Pascal version of the algorithm were classified as comments, comparisons, complex statements, simple assignments, or keywords, according to ratings by computer science faculty. The total fixation durations in each class for both low- and high-experience groups was computed to determine the influence of experience on subjects' attention to critical program areas.

Comments vary in their difficulty level, but they give subjects basic information. Although comments indicate the algorithm's overall purpose, the statements and comparisons and their interrelations are indispensable for algorithm comprehension. Comparison areas are deceptive; although they seem easy to understand, it is hard to recall the direction of the

comparison operator. Complex statements provide the most information about algorithm details. While simple assignments are statements of moderate complexity, they are essential to comprehension of the algorithm, as they provide the necessary contextual information. Keywords in Pascal such as "begin" and "end" are predictable and provide little semantic information; therefore, they can be considered the least important area of the program.

Figure 13 compares the low- and high-experience groups' attention to the five classes of program areas. It shows the average percentage of fixation time spent on each class. (** means significant difference with probability $p \leq 0.01$, and * with $p \leq 0.05$. It is generally accepted that if the difference is significant with probability $p \leq 0.05$ or less, we can be confident that the difference between the two values is not due to chance.)

As anticipated, keywords and simple statements did not attract much attention, and the reading time in these areas was not significantly different for the two experience groups. The comment areas received the most attention from both groups. The reading did, however, depend on experience: The low-experience group spent significantly more time reading the comments. Experience was also a predictor for viewing the comparison areas. Again, the low-experience group perused these areas significantly longer than did the high-experience group.

The greatest and most significant differ-

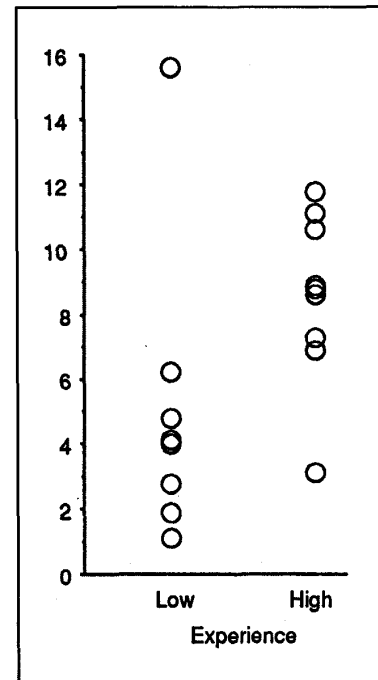


Figure 14. Percentage of time spent viewing "left=right+2."

ence ($p \leq 0.001$) in viewing time between the two experience groups occurred in the complex statements. The high-experience group devoted 68 percent more fixations to these areas than did the low-experience group.

The perception that experienced subjects concentrate on complex statements was further strengthened by our analysis of the most complex area of the binary search algorithm. The statement " $l = r + 2$ " was considerably more complex than versions commonly encountered, as it used a Boolean assignment to condense information usually found in an entire "if-then-else" construct (for instance, the less obtuse but inefficient version "if left = right + 2 then found := true else found := false").

Two months before the experiment, in the CS2 course, the low-experience group was taught two versions of the binary search algorithm, including the " $l = r + 2$ " version. The high-experience group had more exposure to several variations of the algorithm throughout the curriculum. As Figure 14 clearly demonstrates, with the exception of two outliers all subjects in the high-experience group had higher average fixation duration in " $l = r + 2$ " than all subjects in the low-experience group.

(Differences between experience groups were significant, with $p \leq 0.01$.) These results confirm in detail that with increasing experience, subjects learn to discover and focus on the key areas of information, as suggested by research in program comprehension (see sidebar on program comprehension).

Discussion

All subjects in our experiment needed numerous fixations in most areas of the algorithm and spent more time viewing relevant slide areas than did subjects in studies using simple text. This phenomenon is consistent with other studies using complex material. We conclude that viewing strategies for algorithms indeed differ from those for prose.

Moreover, our analysis revealed striking correspondence between subsequent eye fixations and the relevant sequence of attention focus, suggesting the validity of the immediacy and eye-mind assumptions in correctly modeling the reading of complex, abstract material.

Dynamic analysis of eye movement revealed a wide variety of scanning patterns for subjects viewing both text and graphics. While reading Pascal text, subjects showed varied strategies regarding the attention devoted to comments and to code. On the average, subjects in the low-experience group devoted significantly more attention to comments than did those in the high-experience group. Nevertheless, we found code-oriented and comment-oriented subjects in both groups.

When viewing time for graphics slides with two number elements was contrasted with viewing time for graphics containing several numbers, we found a category of subjects who used five times more fixations on the slide with several numbers. Again, those subjects were found in both experience groups.

Using the natural left-to-right and top-to-bottom ordering of the slide areas, we identified distinct classes of scanning patterns. The patterns vary in number and length of scan periods, as well as in the frequency and duration of comparisons. Moreover, a similar distribution of total fixation points was detected.

The classes of scanning patterns transcended the experience levels. The two subjects whose patterns were most similar for scanning both text and graphics belonged to opposite experience groups. The

wide variety of scanning patterns suggests a broad spectrum of cognitive styles. However, our results show that the classification of scanning patterns does not follow the common categorization along the dimension of experience.

Experience does, however, influence viewing strategies with respect to allocation of reading time. Highly experienced subjects recognize and spend more time concentrating on meaningful areas.

While existing literature reports a great variety of scanning patterns, we did not anticipate being able to classify them. Such classifi-

cation should be the focus of further fundamental studies that would provide a basis for tailoring the presentation of material to users' characteristics.

Currently, we are relating viewing strategies to other characteristics of subjects — for example, comprehension and cognitive styles. Moreover, we designed follow-up experiments that contrast the reading of (1) simple stories versus English versions of procedural text, and (2) procedural versus natural-language forms of nested if-then-else rules. These experiments will help us further refine the analysis of individual strategies, classify them, and separate the influence of language formalism from that of semantic content. ■



Martha E. Crosby is an assistant professor in the Department of Information and Computer Sciences at the University of Hawaii at Manoa. Her research interests include human-computer interaction, human factors, cognitive science, models of user cognition, intelligent computer tutors, and evaluation of human use of computer interfaces and applications. Previously, she was a research mathematician at Harry Diamond Laboratories in Washington, D.C., and at the National Bureau of Standards in Boulder, Colorado.

Crosby received her BS in mathematics from Colorado State University in 1959, and both her MS in information and computer sciences and her PhD in educational psychology from the University of Hawaii, in 1975 and 1986, respectively. She is a member of the IEEE, ACM, the American Educational Research Association, the Human Factors Society, and the Pacific Basin Center for Cognitive Science.



Jan Stelovsky is an assistant professor in the Department of Information and Computer Sciences at the University of Hawaii at Manoa. He is director of the Hypermedia Laboratory that organizes hypermedia workshops and produces authoring tools for hypermedia (HyperTalk-to-C translator, video editing), interdisciplinary courseware (for teaching Japanese, Japanese literature, and musical instruments), and hypermedia presentations (guides to the Manoa campus and to Hawaii). His research includes hypermedia, human-computer interaction, computer-aided instruction, and software engineering tools. Previously, he was on the Computer Science Department faculty at the ETH Zurich (Swiss Federal Institute of Technology).

Stelovsky received a diploma in mathematics from ETH Zurich in 1977, an MA in mathematics from Washington State University in 1979, and a PhD in computer science from ETH Zurich in 1984.

The authors' address is University of Hawaii, Dept. of Information and Computer Sciences, Keller Hall 319, 2565 The Mall, Honolulu, HI 96822.