Chapter 19
Case Studies

This chapter presents summaries of studies conducted with various equipment housed in Clemson's eye tracking laboratory. Study examples include the measurement of eye movements in head-mounted virtual reality (VR) and in desktop applications. VR applications used the binocular eye tracker embedded in the helmet-mounted display built by Virtual Research/ISCAN (see Chap. 6). Desktop applications used the table-mounted binocular eye tracker from Tobii (see Chap. 9). Typical use of these devices is shown in Fig. 19.1.

All but one of the examples presented are diagnostic in nature inasmuch as the display does not change or otherwise respond to instantaneous gaze information. The first three examples describe immersive (head-mounted) and desktop virtual environments that were developed from scratch and instrumented from the outset to record eye movements. The fourth example uses the desktop eye tracker and its associated commercial off-the-shelf software (ClearView) to evaluate previously developed applications that were not reinstrumented for eye movement collection in

Fig. 19.1 Head-mounted and desktop (binocular) eye trackers

A.T. Duchowski, *Eye Tracking Methodology*, DOI 10.1007/978-3-319-57883-5_19

any way. The final example is an interactive, or gaze-contingent, application written specifically to evaluate gaze for interactive object selection.

19.1 Head-Mounted VR Diagnostics: Visual Inspection

Sadasivan et al. (2005) used eye movements as a training aid in a visual inspection task in virtual reality. In this study eye movements were also used to corroborate, or rather help explain, performance metrics gathered in the course of training evaluation. As seen in Fig. 19.2, an expert inspector's eye movements were recorded in the virtual environment (left) and then stylized for clarity (right). The stylized display depicted the direction of visual search and the amount of time spent at each Region Of Interest (ROI).

The hypothesis put forth in the study was that participants receiving scanpath-based, feedforward search training would exhibit better performance, as measured by traditional (speed, accuracy) metrics as well as process measures derived from eye tracking.

The experimental design used in this study was a specialized form of the pretest–posttest control group design with one independent variable, the treatment condition at two levels: training and no training.

Performance and process measures were reported in terms of number of defects (targets) detected (accuracy, in terms of true positives), time to task completion (speed), and eye movements (number of fixations, fixation clusters (groups), and mean fixation duration). As indicated in Fig. 19.3 and evaluated by the t test, the effect of training appears to cause significant improvement in accuracy ($p < 0.05$) with a simultaneous deterioration in time to completion ($p < 0.01$). Eye movements helped explain this classic speed–accuracy tradeoff. Although there was no observed significant difference in the number of fixations or fixation groups, the t test of the relative difference in the mean fixation duration was significant ($p < 0.05$). These results suggested that participants who received feedforward training significantly increased their mean fixation duration resulting in a slower-paced inspection strategy. The given interpretation of these results was that novices learned a more deliberate target search/discrimination strategy that required more time to execute.

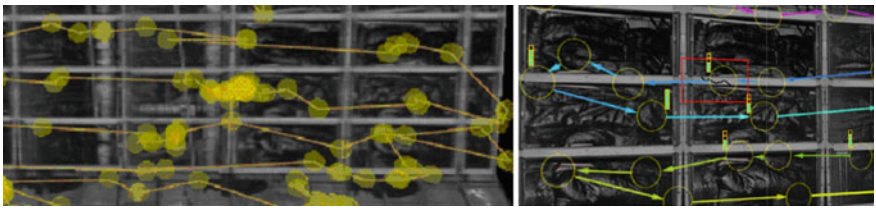
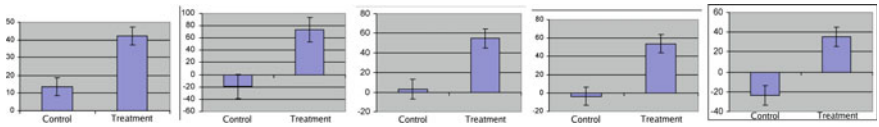


Fig. 19.2 Eye tracking data of expert inspector (*left*), feedforward training display (*right*). From Sadasivan et al. (2005) © 2005 ACM, Inc. Reprinted by permission



(a) True positives. (b) Time to complete. (c) Fixation points. (d) Fixation groups. (e) Fixation duration.

Fig. 19.3 Performance and process measures: relative difference (%). From Sadasivan et al. (2005) © 2005 ACM, Inc. Reprinted by permission

19.1.1 Case Study Notes

There are several limitations of this study, particularly in terms of generalizability. First, because the experiment involved college students, it is not known whether results extend to professionals in the field. It is reasonable to expect that they do because the training examined here is essentially by example which is expected to transfer to the actual task.

Second, the more difficult question of training transfer can be directed toward virtual reality itself: does training in VR readily transfer to the physical realm? The results of the study must inherently be restricted to reporting the effects of training in virtual reality. To examine whether training transfers between the simulated and actual tasks can be posed to most simulators where training transfer is assumed or has otherwise been validated. For example, flight simulators have achieved sufficient sophistication so that they are in some cases used for assessment as well as training. The visual inspection environment used by Sadasivan et al. had been previously evaluated for evoking feelings of "presence" (e.g., "being there") suggesting at least subjective fidelity to the actual task. However, one would need to conduct performance evaluations in the field to completely address this issue.

19.2 Head-Mounted VR Diagnostics: 3D Maze Navigation

The diagnostic study in VR performed by Vembar et al. (2004) used eye movements to test users' utilization of an exocentric overhead 2D map for navigation within a virtual maze environment. The map was a type of "you are here" map wherein the independent variable manipulated was the presence of a directional cue (arrow). The arrow within the maze was either absent (control condition), directionally ambiguous (a dot instead of an arrow), or the arrow indicating the viewer's orientation in the maze (one factor at three levels).

The task given to participants was the same: navigate as quickly as possible toward the center of the maze. Based on previous wayfinding literature, Vembar et al. hypothesized that the directional cue (arrow) would afford the fastest navigation performance, as measured by time to task completion.

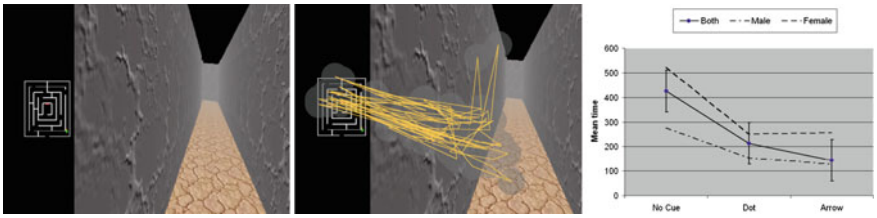


Fig. 19.4 Simple 3D maze with 2D map, fixations, results. From Vembar et al. (2004) © 2004 Eurographics Association. Reprinted by permission

Of 24 participants originally recruited, data from 9 were rejected due to problematic recording (verified by calculating aggregate fixation error over calibration targets). The remaining 15 subjects were randomly assigned to one of three groups, promoting between-subjects analysis.

The test environment, sample recorded and analyzed gaze data, and performance metrics are shown in Fig. 19.4. One-way ANOVA between subjects showed significance effect of the navigational cue on mean completion time ($F(2, 14) = 5.29$, $p < 0.05$). Participants given either the directional arrow or directionally ambiguous cue performed faster than with no cue. Posthoc analysis suggested that gender appeared to influence completion time, with males performing faster than females.

Although gender effects are harder to explain (perhaps gaming or other forms of experience may have played a part), eye movements clearly showed how participants made use of the maze map. Although participants were expected to look at the maze, it was interesting to find that participants given the dot or arrow visual cues would almost exclusively use the 2D map to navigate the maze hardly needing to look at the 3D maze itself. Consequently, it was observed that participants would walk the 3D maze looking at the floor. This blind navigation could be attributed to the over-reliance of the participants on the 2D map and the simplicity of the 3D environment.

19.2.1 Case Study Notes

This was a relatively simple but poignant example of using eye movements to gauge the visual accessibility of an informative visual element. Its simplicity (e.g., stark appearance of the maze environment) may be a valid criticism of the study, however, its direct application of eye movements presents a good example of a straightforward, and quick eye tracking experiment. The study, from conception to analysis was completed in roughly one semester (4 months).

19.3 Desktop VR Diagnostics: Driving Simulator

Balk et al. (2006) conducted a study to evaluate how eye movements change when talking on a mobile phone while driving. To do so, they created a low-fidelity desktop VR driving simulator, instrumented to collect eye movements during simulation trials. The simulator and collected scanpaths are shown in Fig. 19.5. Half of the 16 university student participants viewed dynamic driving scenes while performing a simulated talking-on-a-mobile-phone-while-driving (TMWD) task. The task simulated TMWD by playing audio excerpts from a Japanese language instruction CD. Participants were asked to follow the language instruction audio by speaking when the audio asked to repeat phrases while simultaneously looking for potentially hazardous events in the scenes (eminent collisions, tailgating). Scenes contained four or seven vehicles (all same model, color), giving a within-subjects 2×2 factorial design with the number of vehicles and TMWD task forming the IVs. All participants viewed 12 trials with four vehicles and 12 trials with seven vehicles and then answered multiple choice questions regarding the driving scenarios. A Tobii ET-1750 eye tracker (50 Hz, 0.5° accuracy, 1280×1024 display) was used to collect eye movements with five-point calibrations performed at the beginning and in the middle of each session.

Performance measures indicated significant differences between conditions, as indicated by post-driving questionnaire responses. Repeated measures ANOVA revealed a significant effect in the number of vehicles in the scene ($F(1, 380) = 11.86$, $p < 0.01$). Overall, more questions (60.9%) were answered correctly with four-vehicle scenes than with seven-vehicle scenes (44.3%). A significant effect was also seen between the TMWD and driving-only conditions ($F(1, 14) = 49.59$, $p < 0.01$). Participants in the TMWD condition answered fewer postscene questions correctly than people in the driving-only (control) condition (38.5 vs. 66.7%).

In this dynamic environment, a critical component of analysis was registering fixations over moving vehicles. To facilitate this, the simulator was instrumented to

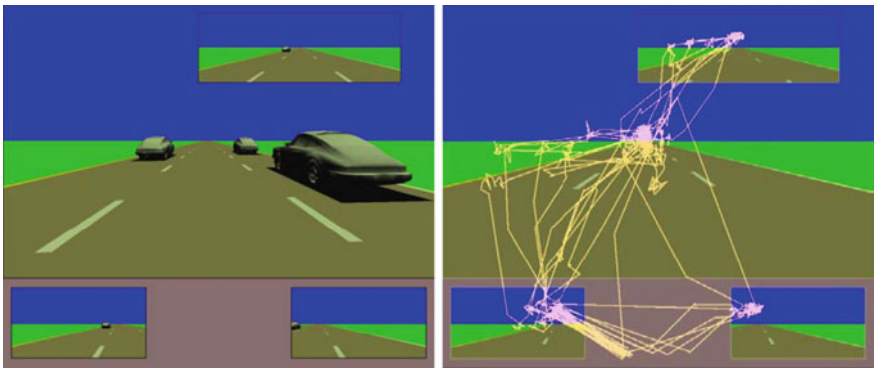


Fig. 19.5 Low-fidelity desktop VR driving simulator and scanpaths. Courtesy of Stacy Balk, Kristin Moore, William Spearman, and Jay Steele

count fixations over the vehicles' 150 × 150 pixel 2D projected bounding boxes (Regions Of Interest, or ROIs). Repeated measures ANOVA showed a significance difference in the percentage of fixations on vehicles involved in hazardous events between driving conditions ($F(1, 14) = 7.37$, $p < 0.05$). Throughout the trial the percentage was greater for the driving-only condition (40 vs. 29%). ANOVA of mean fixation duration also showed that the overall mean duration of fixations in ROIs was greater for the driving-only condition (9.6 vs. 6.5 s, on average, $F(1, 14) = 6.51$, $p < 0.05$).

Participants in the driving-only condition devoted more fixations and time in the ROIs (both during the hazardous event and the entire trial) than people in the TMWD condition. This is not surprising when taking the number of correct responses for each group into consideration. That is, people in the driving-only group appeared to fixate on the vehicle involved in the critical event more than those in the language group: keeping an eye on the hazardous vehicle, so to speak. This is probably due to one of two reasons: an increased mental workload caused the TMWD group to not follow vehicle location after the event, or they did not see the event occur. This study's findings tended to support Gugerty's (1997) finding that when mental workload is increased, drivers are less able to maintain high situation awareness.

19.3.1 Case Study Notes

This study is a good example of a diagnostic study of a dynamic environment on the desktop (desktop VR). Of particular note is the instrumentation effort required to keep track of users' gaze locations over dynamic stimuli (the moving vehicles). Computationally this can be seen as the registration problem, and in this case is not particularly difficult because it requires calculating the 2D projection of a 3D object's bounding box. Of course development of the 3D gamelike simulator in the first place is nontrivial. The point is that given such an environment (a game engine, basically), it is not particularly difficult to reinstrument it to collect eye movements. One key component that must be included, however, is calibration. This requires providing a "calibrate eye tracker" function which, when invoked, presents a number of calibration points to the user. Modern eye tracking hardware may allow the system to store this calibration which can then be used in subsequent sessions (provided the user sits at about the same location as when calibration was obtained).

19.4 Desktop Diagnostics: Usability

The KISS principle was used in the design of an eye tracking usability study of two related software components for the U.S. Navy.

Apparatus. A Tobii ET-1750 eye tracker was used to collect eye movements during display of the Windows XP desktop using Tobii's ClearView software Tobii Tech-

nology AB (2003) with "screen" as the stimulus option (at 15 fps). Both the display and eye tracking server ran on a 2.0 GHz Intel Pentium M Dell Inspiron 9300 laptop equipped with 1 GB RAM and an nVidia 6800 Go graphics card.

Subjects. Four U.S. Navy personnel were recruited (3 M, 1F, average age 25). All had 20/20 corrected vision. One was an experienced operator, the rest were novice users. The experiment was conducted on the exhibits floor of the ForceNet conference.

Design. The usability scripts developed for the study were drawn from a mock Software Acceptance Test (SAT) demonstration performed by an expert (software developer). The demonstration lasted approximately 14.5 min. At Tobii's 50 Hz sampling rate, this yields about 43,500 raw gaze points. Although analysis (i.e., removal of signal noise, classification of fixations, removal of saccades, and fixation clustering) reduces this figure considerably, the resultant aggregation of fixations may still generate a somewhat blurry picture of attention distribution. For example, Fig. 19.6, easily generated by the analysis software, shows the expert's fixation "hotspots" from the entire session over a single representative screen snapshot. Is this type of analysis meaningful, e.g., as informative as Web search's "golden triangle" (Eyetools et al. 2006)? Clearly one can deduce that attention was generally split between the upper-left region (e.g., over menu items) and screen center (e.g., pop-up windows, as shown in this particular freeze frame), but nothing else meaningful readily pops out beyond this simple observation.

To break down the expert's demonstration into manageable chunks for novice users to perform, the task was split into seven subtasks, each renamed as a Software Usability Task, or SUT. The analysis performed to break down the initial ~14.5 min. task resembled Hierarchical Task Analysis (HTA) in spirit but in practice was more informal (formal HTA would probably be quite useful here). The goal was to define tasks that were as atomic as possible, e.g., perform a complete action in each task. On the other hand, the goal was also to keep the subtasks as short as possible. The final seven SUTs partially achieved both goals, although SUTs 001 and 005 were still somewhat long both in duration and number of execution steps. Table 19.1 lists the duration and number of steps of each SUT, as measured for each of the four participants in the study ($S_1 \ldots S_4$) as well as the expert (S_e). SUT-003 and SUT-006 are given in Tables 19.2 and 19.3 in their entirety.

The eventual design adopted for this study was a between-subjects factorial design with an independent variable of task instruction (either visual or verbal). Half the participants were assigned randomly to each group.

Procedure. Participants were seated 50 cm in front of the display and were verbally introduced to the nature of the experiment. They were told that speed and accuracy counted, and were instructed to "complete each task as fast as you can with minimal mistakes (e.g., avoid typing errors)." Each user performed all of the seven SUTs. Each trial consisted of prescripted commands that the user needed to execute. None of the steps was designed to generate errors or other artificial problems for the users. All steps generated the expected responses.

The nature of the software being tested necessitated that each participant follow exactly the same order of SUTs (from SUT-001 to SUT-007) because successive

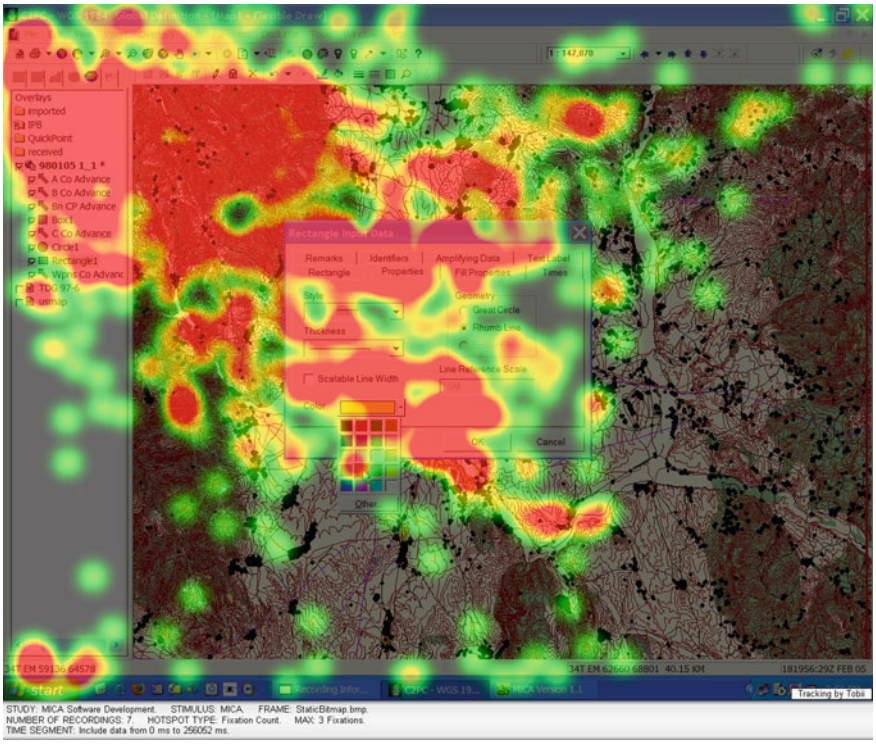


Fig. 19.6 "Hotspots" from expert's ∼14.5 min eye tracking session over a single representative screen snapshot

Table 19.1 Example Software Usability Task durations (in minutes)

Instruction →	Visual		Verbal			
Task ↓ User →	S_1	S_2	S_3	S_4	S_e	# Steps
SUT-001	8.80	8.97	7.40	6.97	4.18	42
SUT-002	2.38	3.22	2.30	2.50	2.05	18
SUT-003	1.48	1.53	1.18	1.25	1.52	4
SUT-004	1.82	1.20	0.88	0.76	0.42	8
SUT-005	6.30	6.30	4.22	4.50	1.60	23
SUT-006	1.42	1.67	0.93	0.73	0.40	7
SUT-007	3.77	3.68	3.05	2.32	2.03	19

steps in the sequence depended on results from previous steps (e.g., entry of database records). Two of the four participants read the instructions and two listened as the instructions were read to them. The expert performed all steps from memory.

Traditional Usability Results. None of the traditional usability metrics is particularly informative. Participants performed all tasks at about the same speed

Table 19.2 Example Software Usability Test SUT-003: RFI submission

User action	System response
Step 76: From the main menu, select **DBA** and then choose **RFI Manager**	**Expected Response**: A new **RFI Manager** window is displayed
Step 77: From the RFI window's menu, select **File** and then choose **New RFI**	**Expected Response**: An RFI fields are cleared for a new RFI to be created
Step 78: Type the following data in the appropriate fields: • Unit – `2mardiv` • RFI Number – `1` • Date Submitted – `181645Z May 00` • LTIOV – `231645Z May 10` • Priority – `1-Flash` (select from drop-down menu) • Status – `OPEN` (select from drop-down menu) • Details – `This is a test of the RFI Submission Process` Select `OK`.	**Expected Response**: The RFI record is added and the scroll buttons become available in the bottom left corner of the window
Step 78: Close the **RFI Manager** window	**Expected Response**: The **RFI Manager** window is closed

Table 19.3 Example Software Usability Test SUT-006: NAI definition

User action	System response
Step 111: Maximize MICA	**Expected Response**: The MICA Application appears
Step 112: Expand the **MICA** level. Expand the **United Endeavor** tree item. Expand **Main Attack**. Click on **DIVCP1**	**Expected Response**: MICA list is expanded. United Endeavor list is expanded. Main Attack is expanded. DIVCP1 is highlighted and the DIVCP1 Details window is displayed in the right pane
Step 113: Right-click on the first item posted to the PIR group box and select **PIR Details** from the menu	**Expected Response**: The PIR window opens
Step 114: In the NAI box on the PIR window, right-click and select **Add NAI**	**Expected Response**: An Overlay Files and Segments window opens listing Overlay files
Step 115: Double-click the **MICA** folder in the left pane of the window	**Expected Response**: The MICA test overlay file appears in the right pane. The file should be named MICA test.mgc
Step 116: Double-click on **MICA test.mgc** in the right pane to expand. Click on one of the boxes created with C2PC and then select **OK**	**Expected Response**: An NAI box is added to the NAI section of the PIR window.
Step 117: Close the PIR window	**Expected Response**: The PIR window is closed

Table 19.4 Task-specific mean responses over all tasks (five-point Likert scale where 1 = strongly disagree, 3 = neutral, and 5 = strongly agree)

	Question	Mean rating (SD)
1, 5, 9, 13, 17, 21, 25	Overall I am satisfied with how easy it was to use this system to accomplish this task	4.69 (0.47)
2, 6, 10, 14, 18, 22, 26	I was able to complete my task quickly using this system	4.63 (0.62)
3, 7, 11, 15, 19, 23, 27	I felt comfortable using this system to accomplish this task	4.69 (0.48)
4, 8, 12, 16, 20, 24, 28	This system had all the functions and capabilities I wanted it to have to accomplish this task	4.75 (0.45)

(with the verbally instructed group performing slightly faster because they did not need to devote time to reading). Accuracy metrics also suggest lack of informative value because none of the participants appeared to make any gross mistakes (e.g., all were adequate typists and sufficiently proficient with the traditional Windows/Icons/Menus/Pointer interface metaphor). A short task-specific questionnaire was administered after each of the seven SUTs and a longer general questionnaire was given following completion of all SUTs. Answers to the questionnaires qualitatively suggest strong satisfaction with the usability of the software (on a five-point Likert scale, users generally either agreed or strongly agreed with questions regarding perceived satisfaction).

In all, eight satisfaction questionnaires were administered to each of the four test participants. Each response sought an agreement opinion from the participant, with the response options given on a five-point Likert scale ranging from "Strongly Disagree" to "Strongly Agree". The first seven sets of four questions were given to participants after completion of each SUT, thus task-specific responses where obtained regarding system usability pertaining to each SUT. The eighth set of 18 questions asked about the user's general impression of the system. Table 19.4 lists the mean task-specific responses over all tasks.

Table 19.5 lists the mean general responses. With the exception of three questions (#36, 38, 40) all the responses agree or strongly agree with the questions, suggesting that the system was generally well organized, capable, easy to use, and well liked by the users. Of course, alternative explanations may be that either the SUT tasks as a whole or the manner of following instructions was overly easy (or both). Only question #36 drew an ambivalent response (neither agree nor disagree) regarding the utility of the systems error messages. Given that the perceived level of errors generated by the system was low in the first place, this response can be interpreted as either being inappropriate to the general simplicity of the tasks, or truly insufficient system responsiveness (feedback). Eye movement evidence provides at least one instance of support to the latter.

Table 19.5 General mean responses (five-point Likert scale where 1 = strongly disagree, 3 = neutral, and 5 = strongly agree)

	Question	Mean rating (SD)
29	Overall, I am satisfied with how easy it is to use this system	4.25 (0.47)
30	It was simple to use this system	4.50 (0.58)
31	I can effectively complete my mission using this system	4.00 (1.15)
32	I am able to complete my mission quickly using this system	4.00 (1.15)
33	I feel comfortable using this system	4.50 (0.58)
34	It was easy to learn to use this system	4.25 (0.50)
35	I believe I became productive quickly using this system	4.00 (1.15)
36	The gives useful error messages that clearly tell me how to fix problems	3.00 (0.82)
37	When I make a mistake using this system, I can recover easily and quickly	4.00 (0.82)
38	The information (help, on-screen messages, tool-tips, etc.) provided is clear	3.75 (0.50)
39	It is easy to find the information I need	4.25 (0.96)
40	The information is effective in helping me complete the tasks and scenarios	3.50 (0.58)
41	The organization of information on the system screens is clear	4.50 (0.58)
42	The interface of this system is pleasant	4.00 (1.41)
43	I like using the interface of this system	4.25 (0.96)
44	This system has all the functions and capabilities I expect it to have	4.25 (0.96)
45	Overall, I am satisfied with this system	4.00 (0.82)
46	I am confident about the results I produced	4.25 (0.96)

Comparing speed of completion of all users and an expert over all tasks, on average, the factor of instruction was significant ($F(4, 575) = 5.8375$, $p < 0.001$). Note that this comparison was performed over steps performed by all subjects including the expert on whose tasks the SUTs were based. Unfortunately, the resultant SUTs deviated slightly from the expert's actual actions, and although all four users performed all of the SUT steps, for the purposes of this analysis, some of the expert's steps must be considered missing.

Mean time to completion for all subjects is shown in Fig. 19.7. Posthoc pairwise t tests with Bonferroni adjustments indicate significant difference only between each of S1 and S2 and the Expert ($p < 0.01$).

Performance analysis (time to completion) does not provide very meaningful insights into any potential problematic aspects of the user interface. Rather, it merely indicates that, on average, users given verbal instructions performed the tasks faster than those who read the instructions. Both of these groups of users were slower

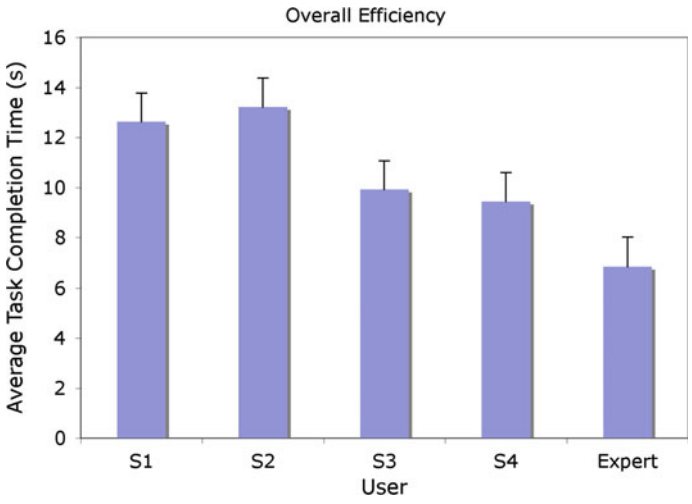


Fig. 19.7 Mean completion times of all users and expert (with SE whiskers)

than the expert who, for analytical consideration of the experiment, had the steps memorized.

To identify particularly troublesome and/or errorprone tasks, all 136 task durations were averaged between users and sorted. The most time-consuming tasks were ones that required typed input; e.g., Step 78 (64.5 s average duration) required entering five typed strings and selection of two drop-down menu items. Steps that did not require as much time generally consisted of single button mouse presses, e.g., Step 23 (Select OK; 2.5 s average duration). Note that the time recorded also included time to read or hear the instruction.

Although most steps did not indicate potential usability effectiveness problems, one action did suggest difficulty: Step 105 (30.5 s average duration, fifth highest). This action requires choosing the Properties dialog from a drop-down menu associated with a graphical box that users were required to overlay atop the DTED map displayed by C2PC. The problem with this instruction is that in order to obtain the correct Properties dialog, the user must right-click on the edge of the graphical element (the box in this case). This is not easy to do if the line width of the element is narrow, as it appears to be by default. This representative user action is shown in Fig. 19.8.

Eye Tracking Results. Investigative (ad hoc) analysis of eye movements revealed three usability problems. First, visualization and playback of SUT-002 (Step 68), shown in Fig. 19.9, revealed seven errant saccades committed by one of the participants (S_2) during visual search for the Edit button. Figure 19.9 shows the sequence of fixations (#74–#81) just prior to Edit button activation. The six saccades leading up to Edit button activation (between fixations #74–76) are directed toward the menu bar, where, presumably, the participant expected to find the Edit button. In terms of satisfaction, this participant did not report perceiving a problem with the Edit button's

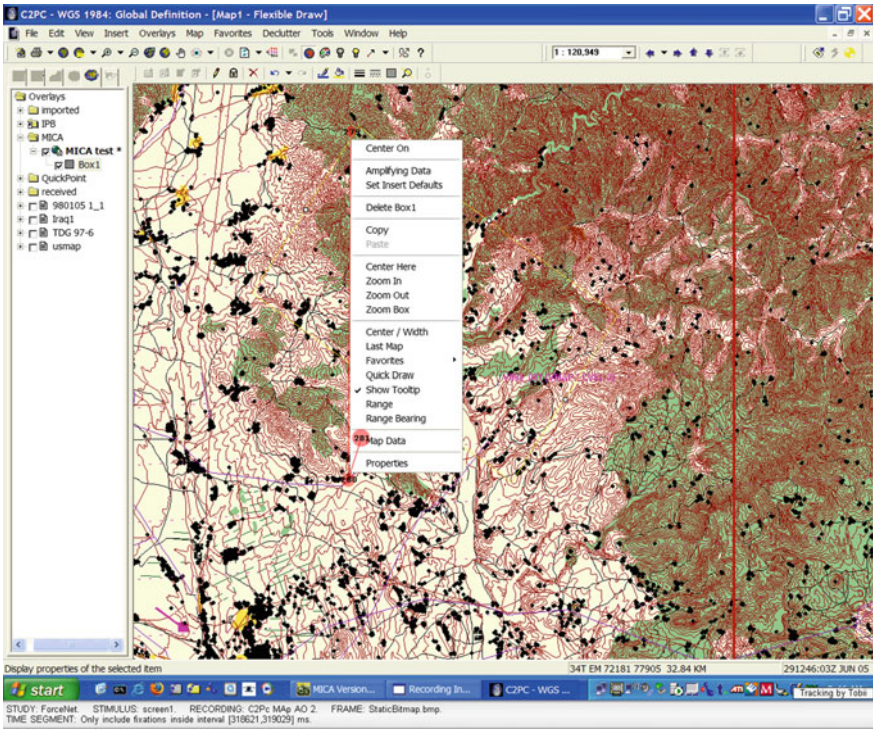


Fig. 19.8 Example of problematic Properties dialog selection. After two unsuccessful attempts, the user has finally right-clicked on the narrow box outline (at *top left* of the popup menu)

unconventional location. However, recall that users may not always report a distracting element during a study Bojko (2005). This scanpath interval lasted about 9045 ms (according to the temporal information reported by Tobii's Gaze Plot analysis; this may seem excessive but is probably due to the participant taking time to read instructions that are away from the monitor). Excluding the first and last fixations (618 ms and 1,096 ms, resp.), 7.3 s were lost to this inefficient search (this duration also includes the time to process verbal or written instructions, however).

Second, and also related to performance, feature visibility analysis shows the lack of prominence of an important system status indicator. The interface, shown in Fig. 19.10, displays fixational "hotspots" collected from all four participants on this task. The image also shows experimenter-defined AOIs (postfacto), overlaid over specific interface features on a representative image of the interface. Two of these features, the status icon and the status bar, relate the application's level of activity. When the application is busy, the status icon (a small red stop sign in the toolbar) glows red and the status bar reports the reason for the delay (e.g., "drawing map"). None of the users looked at the status icon, even though they appeared confused by the application's unresponsiveness. Only one user remarked he "could not complete the

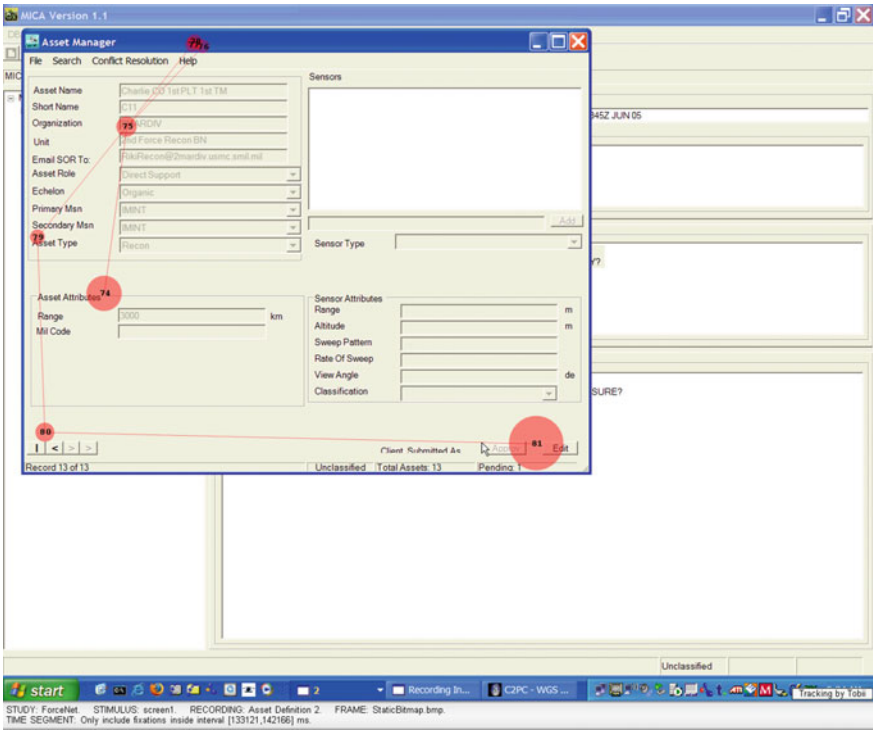


Fig. 19.9 Exemplar errant visual search for Edit button

Overlay task" (ostensibly due to the delay). Figure 19.11 shows descriptive statistics regarding the number of fixations over selected AOIs. Although difficult to read in this compressed version, the second element along the abscissa (status icon) clearly indicates 0 fixations.

Third, area coverage indicators can offer insight into how a user (literally) views the workspace and plans motor actions. In the seventh task, users needed to copy text objects from a source window into a destination window's textbox via two different methods. In once instance, users needed to select the object, then press Add to make the object appear in the destination window. In another instance, users had to click-and-drag the object instead of pressing Add. Area coverage visualizations, ranging from "hotspots" to scanpaths, as shown in Fig. 19.12, showed the need for lookahead eye movements to complete the click-and-drag operation. The user had to move the topmost source window out of the way before visually locating the destination location. The Add button clicking action required neither this time-consuming window moving, nor lookahead eye movements. None of the users reported this as a perceived detriment to performance or satisfaction.

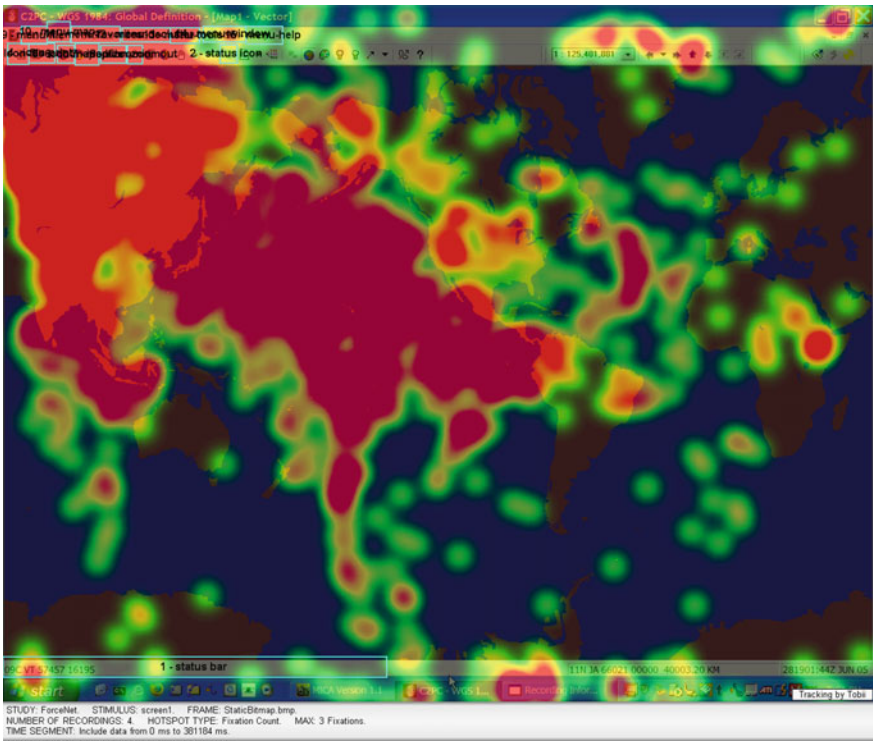


Fig. 19.10 Fixation "hotspots" and selected AOIs

19.4.1 Case Study Notes

Traditional usability metrics alone may not fully capture all aspects of a usability study, whether it's due to the design of the experiment, number of participants, procedures, or environment. Control in this case study was quite weak, and thus the efficiency (speed), effectiveness (accuracy), and satisfaction metrics did not reveal a great deal in this instance. Eye tracking metrics, in contrast, identified rather significant problems with the software suite under inspection.

The key deficiencies in the usability of the system exposed by eye movement metrics all stem from the disregard of rudimentary human factors design principles (consistency, visibility, feedback; Norman 1988) and violation of official user interface style guides (e.g., Microsoft's Windows Microsoft Corporation 1999). Specifically, the misplaced Edit button is a violation of visibility because the button's placement is not immediately obvious (nor consistent with Microsoft's traditional menu layout). Affordance was not obvious in the click-and-drag action of various items, especially when other similar actions provided the explicit use of an Add button. Finally, feed-

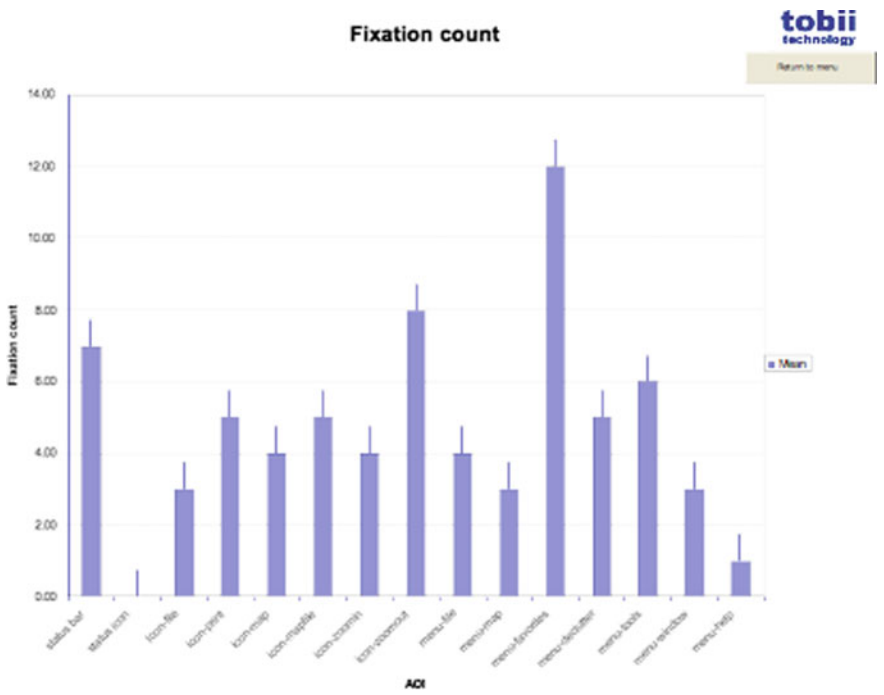


Fig. 19.11 Fixations per AOI (overall, with SE whiskers). Although the labels along the abscissa are barely readable, the noteworthy aspect of this figure is the height of the second element clearly indicating 0 fixations recorded over the status icon

back was not provided during the one of the applications' busy states; the simple inclusion of a progress bar would quickly remedy this problem.

In terms of experimental design, this case study is instructive for two reasons. First, it is counterexemplary in that it shows rather poor experimental design. The design suffers from a very small number of participants. Furthermore, the setting is rife with distractions (the conference exhibits' room floor is neither a good example of a controlled lab setting nor an in-field place of work). The detailed nature of the tasks rendered users' actions too simple; very little cognitive effort was probably exerted during execution of the SUTs. There are other numerous problems, all undoubtedly contributing to the inconclusive performance measures obtained. However, the strength of this example lies in its adoption of the KISS principle. Task duration need not be long to expose usability problems. The inconsistent placement of the Edit button, for example, was identified in only 7.3 s worth of eye tracking data. It's quite possible that had SUT-002 been longer than 2.6 min (on average over the four participants), this significant but rather short event may have been missed.

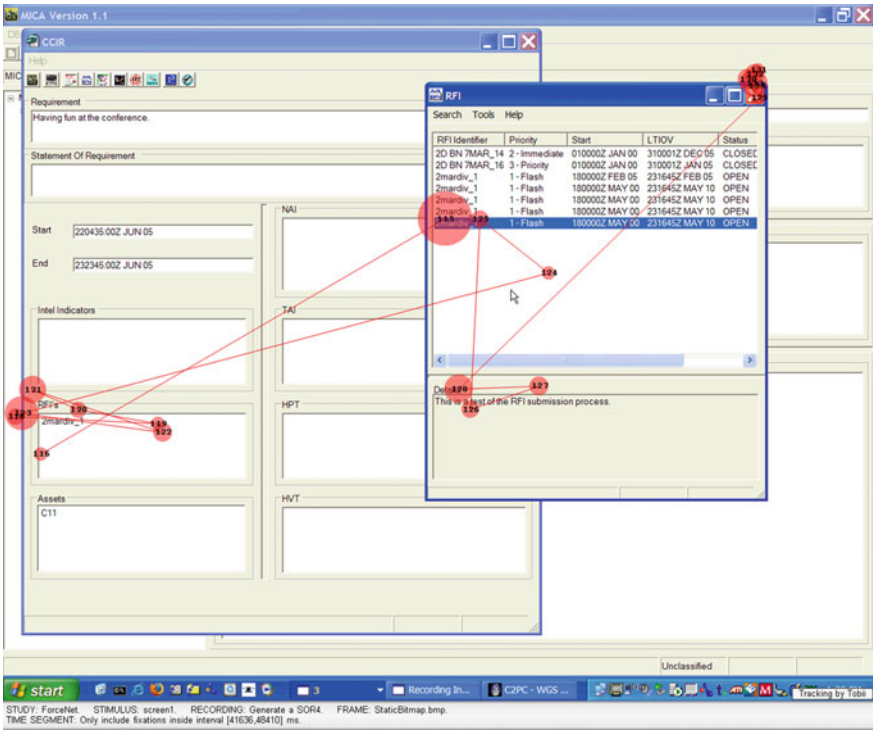


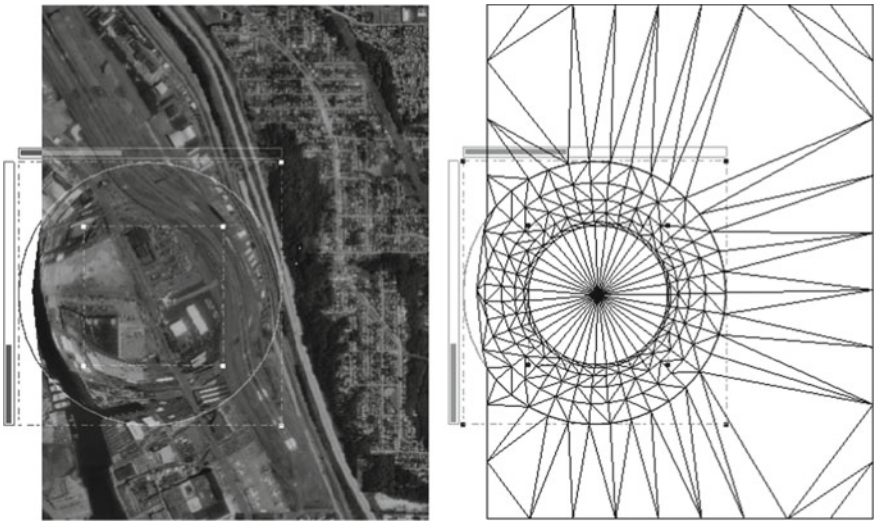
Fig. 19.12 Look-ahead saccades for click-and-drag

19.5 Desktop Interaction: Gaze-Contingent Fisheye Lens

Motivated in part by classic Fitts' law studies of manual pointing and in part by Zhai et al. (1999) gaze cursor warping experiments, Ashmore et al. (2005) evaluated a gaze-contingent fisheye lens as a gaze pointing and selection aid. Applying the Fitts' law model to eye pointing, the study predicted reduced mean selection time through reduction of target distance or expansion of target size, as magnified by the fisheye.

The fisheye lens used in the experiment was the Pliable Display Technology (or PDT) lens, based on the work of Carpendale et al. (1995) and available commercially from Idelix Software.[1] The PDT allows local magnification of data while keeping a continuous visual connection to the underlying display Idelix Software Inc (2004). The lens relies on a geometric transform of data space to display space through a pliable or elastic mesh surface, shown in Fig. 19.13b, enabling content to be stretched from below the surface with lenses of the appropriate height, volume, and shape. Matching data space resolution to magnification factor allows simultaneous target

[1] http://www.idelix.com.



(a) Rendered with lens controls. (b) Underlying mesh representation.

Fig. 19.13 The Pliable Display Technology (PDT) fisheye lens

expansion in motor and display space. The resulting visualization provides a "detail-in-context" view of, and interaction with, the display.

Slaving the fisheye to instantaneous gaze coordinates, preliminary anecdotal observations suggested two problems with the gaze-contingent fisheye. First, a continuously slaved fisheye lens appeared to oscillate during gaze selection. Second, being always in front of one's gaze, the fisheye appeared to disrupt preattentive (peripheral) preview benefit; due to its chosen size the lens contracts screen content beneath its shoulder region (see Fig. 19.13).

A repeated measures (within-subjects) factorial design was used with the style of fisheye lens interaction as the independent variable, at four levels: nonexistent (no lens; control condition), omnipresent (lens continuously slaved to gaze), MAGIC (lens appearing only after fixation onset), and Grab and Hold Algorithm, or GHA (as with MAGIC but fixed in place once visible after fixation onset. The MAGIC moniker was borrowed from Zhai et al.'s MAGIC cursor, but unlike the cursor, the appearance of the MAGIC lens was modulated by real-time fixation estimation and was not based on predicted direction of eye movement).

To evaluate the fisheye interaction for gaze pointing, a combined visual search and Fitts' law pointing task was devised with randomly variable target distance (amplitude) and fixed target width when not magnified by the fisheye lens. A 9×9 grid of target boxes served as the targeting stimulus, as shown in Fig. 19.14 (left). Animation feedback for gaze targeting and selection was provided by first marking the target box with a large $\times$ through its center. Target selection was achieved following a 500 ms fixation (dwell time) within the box boundaries in display space (snap-to-target was

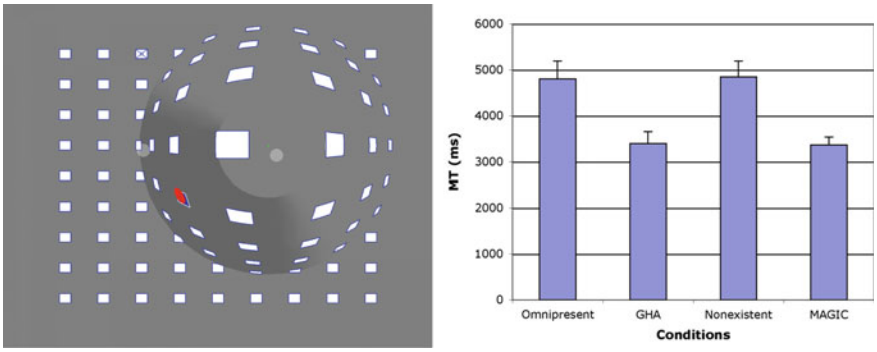


Fig. 19.14 Gaze-contingent fisheye lens stimulus and performance results. From Ashmore et al. (2005) © 2005 Canadian Information Processing Society. Reprinted by permission

not used). Visual confirmation of selection was provided by highlighting the selected box and drawing a red disk centered at the mean fixation position with radius equal to the standard deviation of fixation points collected during the fixation. Following calibration, each participant completed four blocks of 40 trials. Odd-numbered trials required the participant to fixate the central home position box. Even-numbered trials required the participant to search for and select a randomly chosen target from the remaining 80 target boxes (excluding the central target).

Mean selection time (MT; in milliseconds) of participant data pooled per fisheye condition is shown in Fig. 19.14 (right) with standard error (SE) whiskers. A standard weighted-means, one-way ANOVA of correlated samples was performed to examine the main effect of different fisheye conditions. The main effect (fisheye) was significant ($F(3, 1436) = 7.8407$, $p < 0.01$). Posthoc Tukey pairwise comparisons revealed significant differences between the Omnipresent condition and both GHA and MAGIC conditions ($p < 0.01$) but not the Nonexistent condition. A significant difference was also found between the GHA and Nonexistent conditions ($p < 0.01$) and between the Nonexistent and MAGIC conditions ($p < 0.01$). The GHA and MAGIC conditions gave the fastest performance, with neither significantly better than the other. The Omnipresent and Nonexistent conditions gave the slowest performance and were not significantly different from each other.

Similar analysis was performed for accuracy, as measured by the gaze point's deviation from the target center. Once again the main fisheye effect was significant ($F(3, 1436) = 5.59$, $p < 0.01$). Posthoc Tukey pairwise comparisons revealed significant differences between the GHA and MAGIC conditions ($p < 0.01$) and between the Nonexistent and MAGIC conditions ($p < 0.01$). The MAGIC and Omnipresent conditions provided the best accuracy, with neither significantly better than the other. The GHA and Nonexistent conditions provided the worst accuracy and were not significantly different from each other.

The most likely reason given for speed improvement outcomes was that both GHA and MAGIC lenses provided the viewer with preview benefit when conducting visual

search for the target. Of the three lenses, only the GHA lens was held at a fixed location following fixation onset. If the user's initial accuracy was poor, the target appeared in the distorted lens shoulder, negating any potential target expansion benefits. Although the GHA lens may have offered a speed benefit during visual search, it hindered target selection, thus providing a possible explanation for improved accuracy of only the Omnipresent and MAGIC lenses.

19.5.1 Case Study Notes

This study is a good example of an interactive, or gaze-contingent, eye tracking experiment. It is fairly simple in its construction and control (e.g., simplicity of box targets).

The experiment is also a good example of development of an eye tracking (and PDT fisheye lens) client application. In essence, the resultant application invoked several Software Development Kits (SDKs) to achieve its desired functionality: the Tobii SDK (under Linux), the PDT SDK (from Idelix), as well as less-specialized Application Program Interfaces (APIs) such as OpenGL, Posix threads, and Qt.

It is also worth mentioning that project management in terms of collaborative and revision handling was provided by subversion (svn). Analysis (automated via Postgres MySQL queries) was facilitated by the R analysis language. Eventually, following data collection (the actual experimental trials), the entire analysis was performed by simply typing "make" at the Linux prompt. Paper typesetting was performed by LaTeX, the source of which was also held in the same svn repository.

19.6 Summary and Further Reading

This chapter reviewed a number of eye tracking studies performed in Clemson's eye tracking laboratory. Most of the work is described in the associated referenced publications and these ought to be consulted for further details. Indeed, the proceedings that contain these papers or posters hold a wealth of information embodied in similar works. These sources include the proceedings of ETRA (Eye Tracking Research and Applications), SIGCHI, Graphics Interface (GI), EuroGraphics conferences (there are numerous specialized meetings and workshops), and APGV (Applied Perception in Graphics and Visualization). APGV is a fairly recently formed conference, a spinoff from SIGGRAPH, hence is largely concerned with graphics-related work.