Behavior Research Methods, Instruments, & Computers
2002, 34 (4), 605-612

ILAB: A prog ram for postexperimental eye movement analysis

DARREN R. GITELMAN
Northwestern University Medical School, Chicago, Illinois
and V. A. Healthcare System, Chicago, Illinois

The recording and analysis of eye movements are fundamental to a diverse set of research applications, including studies in which reading, visual search, and both overt and covert visuospatial attention are examined. Software tools supplied with commonly available eye-tracking equipment have generally been limited in functionality and nonextensible. Because of this dearth of available software, ILAB was created to provide an extensible framework for analyzing various aspects of eye movements. The program consists of a series of open-source MATLAB functions. The program's data structures keep raw data, analysis preferences, and analyzed data separate, thus maintaining data fidelity and promoting extensibility.

The recording and analysis of eye movements are important research tools in such areas as neuroscience and vestibular and oculomotor control. Quantifying the position of the eyes and the pattern of eye movements has been critical for understand visual and cognitive behaviors, including visual search (Gitelman, Parrish, Friston, & Mesulam, 2002; Rayner, 1995), driving (Land & Lee, 1994), reading (Rayner, 1995), and arithmetic (Suppes, 1990), among others. These few examples touch on only a tiny fraction of the large number of experiments that utilize eye movements.

Eye-tracking hardware has shown steady improvements over the last decade. Units employing infrared technology are noninvasive, are generally easy to use, and are marketed by over a dozen companies (Wooding, 1999). Although manufacturers of eye-tracking equipment generally include software for analyzing eye movements, the programs are usually proprietary, nonextensible (i.e., they cannot analyze data from other systems), and nonmodifiable in the sense that the user cannot add additional analyses or modify the algorithms. Furthermore, because eye movement datasets are frequently quite large, memory limitations may make them difficult to analyze in standard spreadsheet programs. The development of an extensible software platform for analyzing eye movements therefore became critical for our own research. This article describes a software program, ILAB, designed to fill this need. It has been developed in our laboratory and is freely available to other investigators.

Two events helped shape the development of the ILAB software. The first was the increasing distribution of "free" software throughout the 1990s on the basis of the principles of the Free Software Foundation (http://www.fsf.org/philosophy/free-sw.html; Williams, 2002). Examples of programs that followed these principles include Statistical Parametric Mapping (SPM; Frackowiak, Friston, Frith, & Dolan, 1997; Friston, Ashburner, Heather, Holmes, & Poline, 1994–2002) and Analysis of Functional NeuroImages (AFNI; Cox, 1996) for the analysis of functional imaging data and the Psychophysics Toolbox (Brainard, 1997; Pelli, 1997) for psychological stimulus display and data collection. All three programs have been used in hundreds, if not thousands, of laboratories worldwide. An important aspect of this manner of program distribution is that users are allowed, indeed encouraged, to adapt and improve the program as dictated by their needs. This principle promotes two-way interactions between the developer and the users and potentially enables more rapid bug fixes and program improvements that would have been difficult to implement had the developer worked in isolation (Williams, 2002).

The second important event leading to the development of ILAB was the availability and increasing sophistication of the MATLAB (Mathworks, Natick, MA) software environment. This software is designed around linear algebra principles and is able to work with large datasets (such as eye movement data). It includes tools for algorithm development (a high-level interpreted language) and enables the rapid design of advanced graphical interfaces. In addition, many additional toolboxes of prepackaged algorithms (e.g., signal and image processing, statistics, error minimization, curve fitting, system identification, wavelet, etc.) are available for the MATLAB software platform. How-

---

This work was supported by National Institute on Aging (NIA) Grant K23-AG00940 (to D.R.G.). The contributions of John Bernero and Roger Ray in programming ILAB have been invaluable and greatly appreciated. In addition, the comments of four reviewers have helped to significantly enhance the functionality of ILAB. D.R.G. is affiliated with the Beinberg School of Medicine and the Cognitive Neurology and Alzheimer's Disease Center at Northwestern. Correspondence concerning this article should be addressed to D. R. Gitelman, M.D., Northwestern University Medical School, 320 E. Superior St., Searle 11-470, Chicago, IL 60611 (e-mail: d-gitelman@northwestern. edu).

ever, in order to make ILAB accessible to the widest audience, the program was designed to run with only the default set of MATLAB toolboxes. Therefore, the user does not need to make additional purchases beyond a basic MATLAB license.

ILAB could have been written in a low-level computer language such as C, which would have freed it from dependency on other software. However, although low-level languages are very powerful and flexible, they are not as conducive to rapid program development or easy implementation of graphical interfaces. Interpreted languages, such as BASIC, allow easier program development than do low-level languages but still do not include all the prepackaged algorithmic and graphical tools in the basic version of MATLAB. Thus, MATLAB was chosen for its relative ease of use and because it was one of the few software packages of its type available at the time ILAB was first being developed in 1995.

The specific goals in designing ILAB were the following: (1) to display basic eye movement data, including scanpaths, fixations, and saccades; (2) to allow basic quantification of analyzed eye movement data, including tabulation of fixations and saccades and correlation of eye position with a region of interest (ROI); (3) to allow accurate mapping between the eye tracker and the computer screen; (4) to enable display of eye movement data on the same images as those the subject was viewing; (5) to display pupil size data; (6) to enable export of data and results to facilitate analyses in other applications; (7) to apply any of the preceding methods to a variety of eye-tracking systems; and (8) to develop a system that allowed expandability over time.

Program Design

ILAB takes advantage of MATLAB's ability to define data structures, which permits the association of various types of related information. For example, one can maintain an association between a subject number, a study date, and eye movement data within the same data structure. This organization facilitates program design by having related data linked together. The three main data structures in ILAB are the following: the ILAB structure, which contains raw data, subject information, and the coordinate mapping from the eye tracker to the computer screen coordinates; the PLOTPARMS structure, which contains information about the display of various types of data and calculation results; and the ANALYSISPARMS structure, which contains parameters for performing various calculations and the results of those calculations.

Figure 1 is a diagram of the overall design of the ILAB program. All eye movement data are first converted into the fields stored in the ILAB variable and saved to disk in MATLAB's native *mat-file* format. This allows rapid loading of the data for future analyses and the ability to quickly access the original data for all calculations. The mat file is also hardware independent, enabling data to be analyzed on any platform that runs Version 5.2 or later of MATLAB. The four fields in the ILAB variable that must be filled in are the eye data, an index for each trial, the acquisition rate, and the coordinate system. The eye data field is an $n \times 4$ array[1] in which the first two columns define the horizontal and vertical eye positions and the third contains numerical flags allowing the partitioning of the dataset into time-limited subsets or trials or permitting time-locking of the eye data to external events. The fourth column contains information about pupil size when it is available.

The definition of trials is an important feature of ILAB, since it allows the user to designate an association between particular time intervals in an experiment and a corresponding set of eye movement data points. The designation of specific intervals of eye movement data for each trial is contained in the index field. The index field is formatted as an $m \times 3$ array, in which m is the number of trials and the three columns represent row indices to the data corresponding to the start, stop, and (if needed) target points for each trial. Figure 2A illustrates the eye data organization, and Figure 2B shows the organization of the index variable. Setting up this index field can be done automatically, if the user has instructed the eye-tracking program to delineate the trials. It can also be done manually by designating the start and the end of each trial, using a graphical editor within ILAB.

The third required field in the ILAB variable is the acquisition rate. This is the speed at which data points are acquired. This is generally between 60 and 1000 Hz for typical infrared-based eye trackers and is assumed to be constant throughout an experiment. The inverse of the acquisition rate is the duration of each data point in milliseconds (e.g., $1/60 = 16.67$ msec per data point), providing a built-in time reference for the data.

Coordinate Transformation

Coordinate transformation of the data is necessary because the eye tracker screen coordinates may not map in a one-to-one fashion to the computer screen coordinates. In addition, the central axes of each screen may not be aligned. Thus, the eye tracker and computer screen do not line up precisely at any particular point, such as 0,0 or the center of each screen. This calibration is entirely separate from determining a subject's direction of gaze or point of regard (POR), which maps the subject's eye position as measured by the pupil center and corneal reflection positions to points the subject is fixating. POR calibration is not currently part of the ILAB package.

The user must determine the eye tracker to computer mapping empirically by measuring the location of an object on both screens. Tests of both ISCAN (Burlington, MA) and ASL (Natick, MA) infrared eye-tracking systems show that the computer to eye tracker mapping is well approximated by a set of independent linear equations for the horizontal and vertical directions. These equations are solved using least squares.

MATLAB is very fast at applying the coordinate transformations to a matrix of values and takes only 0.05 sec to transform 100,000 data points on an 800-mHz Pentium III machine. The accuracy of this transformation is, on aver-

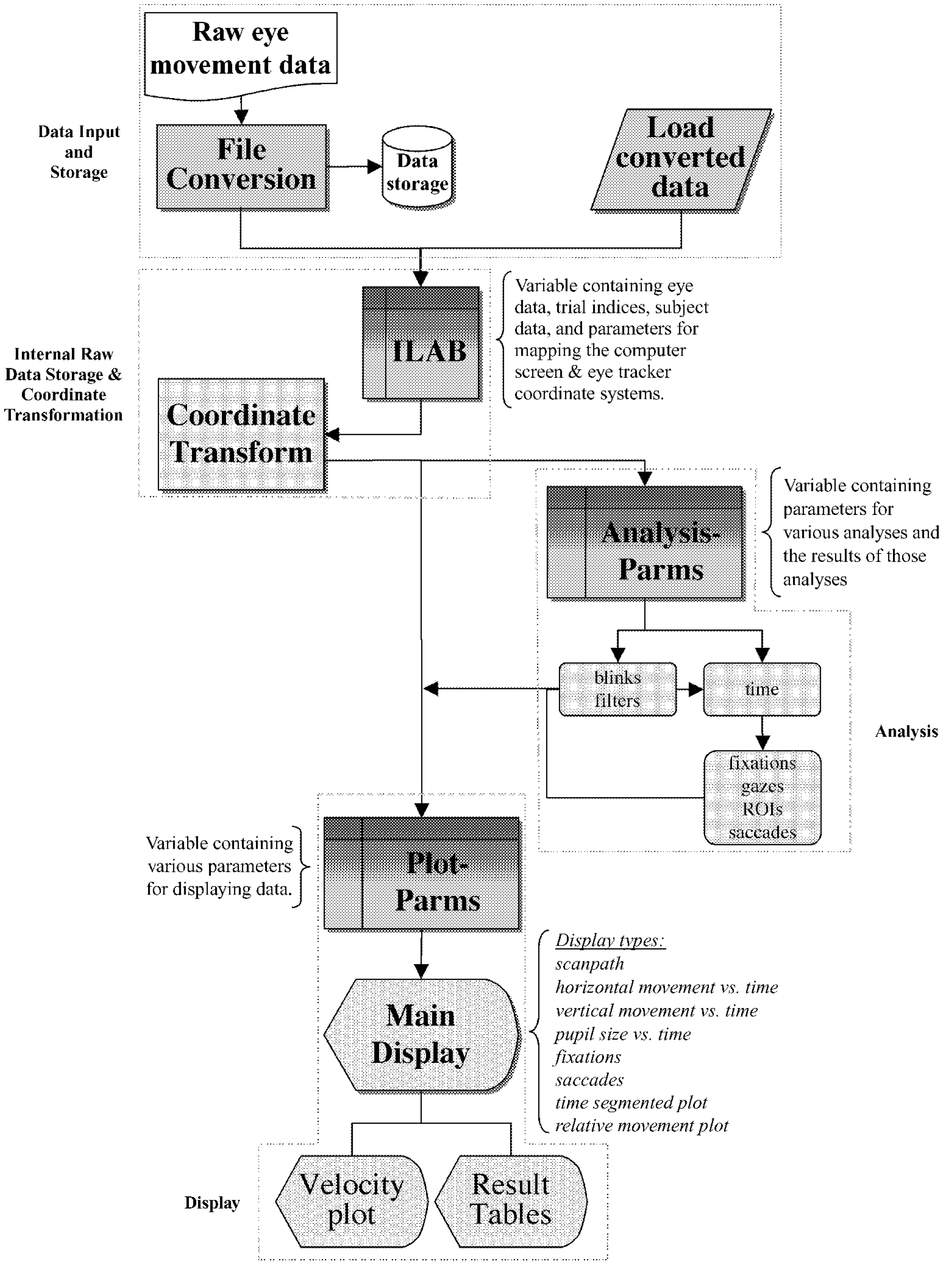
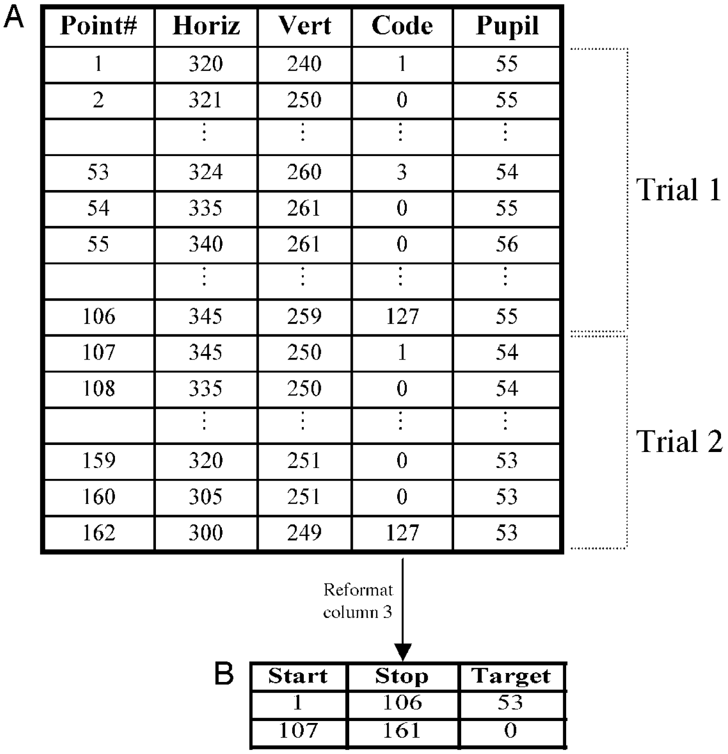


Figure 1. Diagram of variables and methods in ILAB. This figure serves as both a flowchart and a structural depiction of the ILAB program. An analysis always starts with converting the raw data into the native ILAB format—that is, the ILAB variable. Converted datasets can be saved as MATLAB "mat" files and rapidly reloaded at another time. There are four main program sections: data input and storage, coordinate transformation, analysis, and display. Dotted/dashed lines delineate each section. There are three main global data structures in ILAB (shown by the gradient filled icons). The ILAB variable stores raw data; ANALYSISPARMS stores both parameters and results of various analyses; and PLOTPARMS stores information associated with different displays. A summary of information contained within each variable is listed next to it. This organization provides a convenient way for developers to access the data and to store information related to calculations and display. Input methods are indicated by parallelogram icons under data input and storage, calculation methods are indicated by rounded rectangles, and display methods are indicated by angled ovoid icons at the bottom of the figure. The list next to Main Display is the type of information that may be displayed in the main plot area.



Point#	Horiz	Vert	Code	Pupil
1	320	240	1	55
2	321	250	0	55
	⋮	⋮	⋮	⋮
53	324	260	3	54
54	335	261	0	55
55	340	261	0	56
	⋮	⋮	⋮	⋮
106	345	259	127	55
107	345	250	1	54
108	335	250	0	54
	⋮	⋮	⋮	⋮
159	320	251	0	53
160	305	251	0	53
162	300	249	127	53

Start	Stop	Target
1	106	53
107	161	0

Figure 2. Internal organization of the data (A) and index (B) variables. The point# column has been included for illustration purposes. It is not present in the actual variable. The boundaries of two trials are delineated in panel A, and their respective indices are given in panel B. Trial 1 has a target marked at data point 53. This might be an external event (e.g., a target) that should be specifically time-locked to the data stream. The absence of any target event in Trial 2 is coded as a 0.

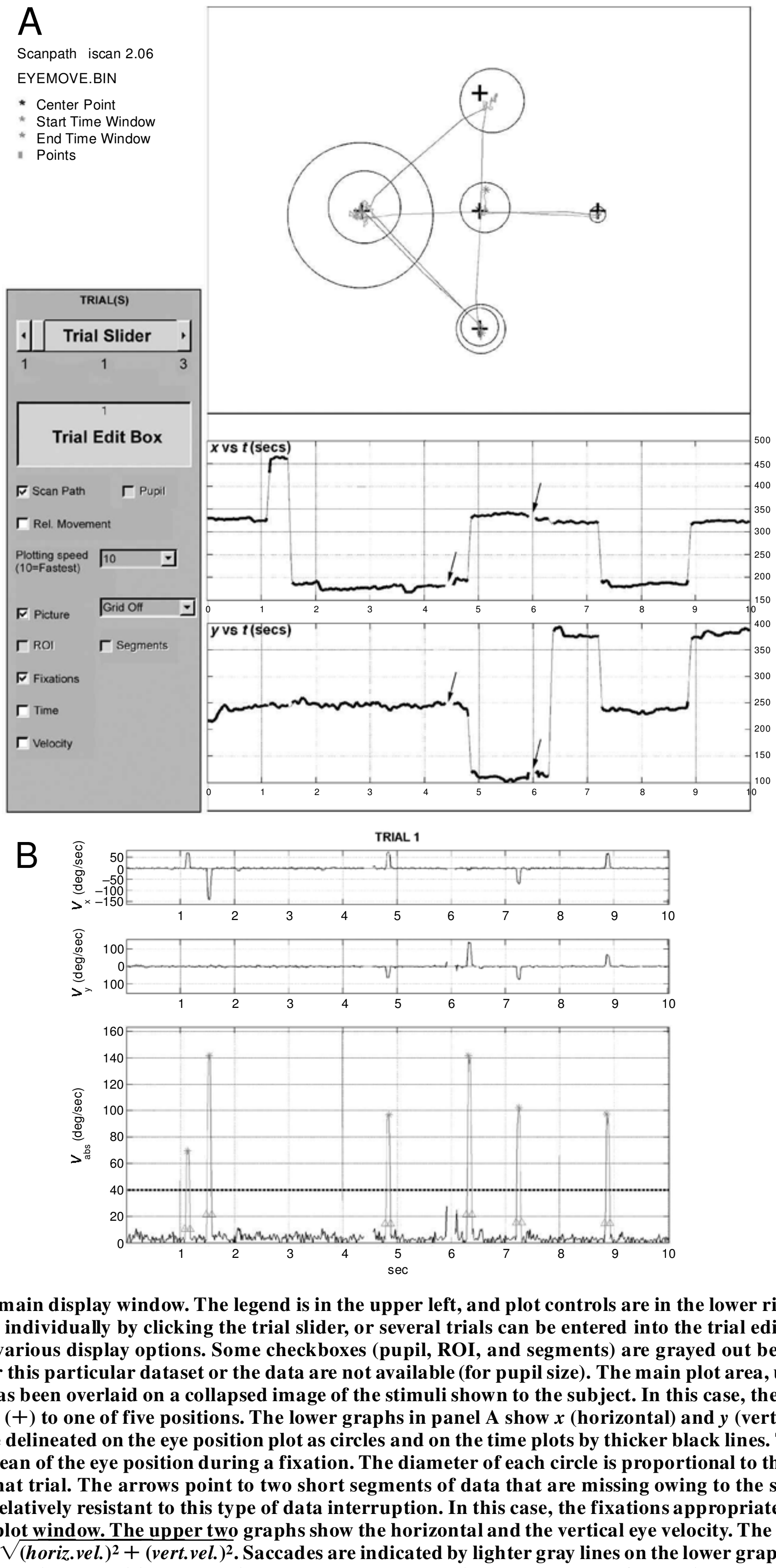
age, in the subpixel range. Accuracy was determined by comparing the results obtained by direct measurement of object locations on the computer screen with object locations determined by the linear equations. For the ISCAN (512 × 512) to PC (640 × 480) transformation, the maximum absolute error was 1.5 pixels horizontally and 1.1 pixels vertically. The mean absolute error (±*SD*) was 0.56 ± 0.29 pixels horizontally and 0.58 ± 0.39 pixels vertically, with a mean absolute total vectorial error of 0.88 ± 0.32 pixels ($\sqrt{(0.56)^2 + (0.58)^2}$). In our setup, 1 pixel corresponds to 0.02° of visual angle; thus, coordinate transformation error is over an order of magnitude less than the measurement error of a standard infrared system. A similar linear coordinate transformation has been proposed previously for this calculation (Duchowski et al., 2000). Of note, this type of coordinate transformation is device independent as long as the video setup does not warp either the computer or the eye tracker screens nonlinearly. Thus, one can transform eye tracker coordinates to nearly any screen resolution.

The PLOTPARMS variable (Figure 1) contains user-settable preferences regarding the display of the data. It also contains a copy of the data following coordinate transformation and following the performance of any analyses on the data. The plotting routines are then able to use the contents of PLOTPARMS for display. ANALYSISPARMS contains preferences for performing various calculations and stores the results of those calculations for rapid redisplay. The display section at the bottom of Figure 1 lists a variety of ways to display the eye movement data.

Graphical Interface

The main ILAB program screen, after loading a dataset, is shown in Figure 3A. The legend in the upper left indicates the name of the dataset, the dataset type, and the key to any colors or symbols that are used in the main plot. The lower left of the main window contains a set of controls allowing the user to display any subset of trials and previously calculated results. Note that some of the checkboxes are grayed out because calculations have not been performed or, in the case of pupil data, are not available in this dataset.

The menus in ILAB (not shown) enable the user to access all of its functions. In brief, the File menu allows converting the raw data files, loading converted data files, loading add-on toolboxes, and printing. The Edit menu allows the user to change various properties of the dataset and to change display preferences. The Analysis menu contains the various types of analyses for manipulating the data. After an analysis has been run, the appropriate sub-



Trial Edit Box: 1

☑ Scan Path ☐ Pupil

☐ Rel. Movement

Plotting speed (10=Fastest): 10

☑ Picture Grid Off

☐ ROI ☐ Segments

☑ Fixations

☐ Time

☐ Velocity

Figure 3. (A) ILAB main display window. The legend is in the upper left, and plot controls are in the lower right of the display. The user can display trials individually by clicking the trial slider, or several trials can be entered into the trial edit box. Checkboxes are used to show or hide various display options. Some checkboxes (pupil, ROI, and segments) are grayed out because the calculations have not been done for this particular dataset or the data are not available (for pupil size). The main plot area, upper right, is a graph of eye position. This has been overlaid on a collapsed image of the stimuli shown to the subject. In this case, the subject followed individual upright crosses (+) to one of five positions. The lower graphs in panel A show *x* (horizontal) and *y* (vertical) eye positions versus time. Fixations are delineated on the eye position plot as circles and on the time plots by thicker black lines. The center of each circle is the arithmetic mean of the eye position during a fixation. The diameter of each circle is proportional to the display time relative to other fixations in that trial. The arrows point to two short segments of data that are missing owing to the subject's blinking. The fixation algorithm is relatively resistant to this type of data interruption. In this case, the fixations appropriately spanned this gap in the data. (B) Velocity plot window. The upper two graphs show the horizontal and the vertical eye velocity. The lower graph shows the vectorial velocity (i.e., $\sqrt{(horiz.vel.)^2 + (vert.vel.)^2}$. Saccades are indicated by lighter gray lines on the lower graph. These saccades correspond to the ones shown in the position versus time eye movement traces in the lower part of panel A. The beginning and the end of each saccade are indicated by triangles, and the peak is indicated by an asterisk. In the actual display, the saccade is colored red.

menus under Windows are enabled so the user can reexamine any calculated results. Annotations can be added to the graphs by using the Tools menu. The ILAB-Help menu brings up a help file covering all aspects of using ILAB. This help file has been integrated with MATLAB's help facility and is written in HTML.

Visualization Tools

The ILAB main window allows a number of simple displays for visualizing data. In the upper right of the main window is the Coordinate Plot Area. Horizontal and vertical POR data are plotted in this section (scanpath in Figure 3A). Data points can be plotted (relatively) instantaneously or at various speeds to allow the user to visualize the eye movements over time within each trial. The user can also set up images to be displayed at the same time as the data, allowing dynamic visualization of both the scanpath and the exact scene that was shown to the subject. Image display is encoded through an image management dialog (not shown), which allows the user to choose an image and assign it a trial and time of display. ILAB is able to read the image types currently supported by MATLAB, including JPEG, TIFF, GIF, BMP, PNG, HDF, PCX, and XWD.

Relative movement visualization (Rel. Movement in Figure 3A) colors the scan path segments red or green on the basis of whether the horizontal vector of the eye movement was directed to the right or the left, respectively. This type of plot has been valuable for visualization of eye movements in patients with neglect, for example. The segmented plot displays the POR data segmented by either absolute or relative time in a trial. Segments are indicated by using different colors for various time intervals in a trial. This display can show where subjects are looking at particular time intervals in a trial (LaBar, Mesulam, Gitelman, & Weintraub, 2000). Additional display options include showing fixations, saccades, and ROIs.

Analyses

ILAB contains algorithms for a variety of basic analyses, such as delineating fixations and saccades, which are fundamental to examining eye movements. Associations between eye movements and one or more user-defined ROIs can also be examined. Gaze analysis is just a special case of ROI analysis that is particularly geared toward checking the maintenance of fixation during a trial. Blink removal and filtering (smoothing) allows the removal of certain types of artifacts in the data. The user has the option of exporting the calculated results or the raw data in order to allow calculations in other programs—for example, statistical software. The various analyses will be discussed further below.

Although the fixation and saccade algorithms had been previously validated (Fischer, Biscaldi, & Otto, 1993; Salvucci & Goldberg, 2000; Widdel, 1984), a dataset was collected to display the results of different analyses. The data were acquired using the RK-426 infrared video eye-tracking system (ISCAN, Burlington, MA) at 60 Hz. The subject was instructed to follow an upright cross (+) as it moved to one of five locations on the screen and, otherwise, to keep her eyes fixed on the cross. The fixation periods lasted from 500 to 3,250 msec. There were 100 fixation periods. The subject was also instructed to blink her eyes whenever the cross changed color from black to red and back, which occurred 56 times during the 3-min 25-sec experiment.

Fixations. Fixations are calculated using the moving window algorithm described by Widdel (1984) and later reviewed by Salvucci and Goldberg (2000) as an example of a dispersion-type algorithm. As compared with a variety of other methods for identifying fixations, including those based on velocity, hidden Markov models, minimum spanning trees, and areas of interest, Widdel's algorithm was found to be accurate, robust, reasonably fast, and fairly easy to implement (Salvucci & Goldberg, 2000; Widdel, 1984). However, this algorithm is sensitive to the choice of parameters defining a fixation, including the maximum horizontal and vertical eye movements and the minimum duration of the fixation (Karsh & Breitenbach, 1983). Thus, it is important for the user to define these parameters on the basis of the experimental goals and type of equipment and to keep these parameters constant across subjects in an experiment (Karsh & Breitenbach, 1983). ILAB allows the user to define and save these parameters in order to standardize this analysis. The algorithm is also relatively resistant to the presence of missing points—for example, owing to eye blinks (see Figure 3A)—by skipping over these intervals as long as the POR remains within the developing fixation cluster before and after the loss of data. If the POR leaves the cluster, a new fixation is searched for.

Figure 3A shows the locations of fixations (gray circles) plotted on 10 sec of a subject's scanpath. This has been overlaid on a composite image of the targets. The fixation circle sizes are proportional to the fixation duration. The user has the option of showing one or more fixation locations on the display. Overall, ILAB found 103 fixations. It identified all 100 predefined fixation periods correctly. The 3 additional fixations were caused by eye blinks' splitting a longer period of fixation. This occurred only three times out of a total of 56 predefined blinks. The performance of the blink algorithm is illustrated further below.

Saccades. Saccades are calculated using an algorithm outlined by Fischer et al. (1993). The algorithm first searches for intervals during which the eye's velocity exceeds a threshold (e.g., 30–40 deg/sec), as a first pass definition for locating a saccade. Within each interval, the peak velocity is located, and the final bounds of the saccade are those points that equal or exceed 15% of the peak velocity. Calculation of saccades at 60 Hz, as shown in Figure 3, is a highly debated topic in the eye movement literature (Karn, 2000), and its usefulness depends on the size of saccades being studied. The algorithm used in ILAB was originally validated at a 1-kHz acquisition rate (Fischer et al., 1993). When used at other acquisition rates, the velocity display (Figure 3B) in ILAB allows the user to assess the performance of the algorithm. In the example shown, with an initial threshold of 40 deg/sec, the saccade algorithm found the exact number of saccades made by the subject (101) after blinks were filtered from the datastream.

Figure 3B illustrates the velocity plot display of ILAB, showing a 10-sec interval of eye movements. The lighter gray lines in the lower trace identify saccades. The upper traces are velocity versus time plots for the horizontal and the vertical directions, respectively.

Eye blinks. During eye blinks, the eye tracker loses track of the pupil center and corneal reflection. Because eye blinks are not instantaneous, the amount of data loss can be variable. Furthermore, when the eye opens and the pupil and corneal positions are reacquired, there are frequently distortions in the eyes' calculated POR as they rotate back to their original positions (Calkins, Katsanis, Hammer, & Iacono, 2001). A particularly dramatic example of these distortions is shown in Figure 4A.

Eye-tracking systems deal with this loss of eye position information in a variety of ways. The ISCAN system (ISCAN, Burlington, MA), for example, tends not to show a pupil size of 0 during the blink, and the eyes' position may or may not be marked as invalid. Although ASL systems (ASL, Natick, MA) usually mark the pupil size as 0 during a blink, incorrect eye position information may still not be marked as invalid. In order to remove eye blink artifacts from the data, two methods have been used: (1) filtering the data by incorrect position information and (2) filtering by 0 pupil size. For the first method, data points are eliminated when they are outside the bounds of the computer screen. The second method eliminates those data points at which the pupil size is 0. The combination of these two strategies can be quite effective at minimizing blink-related artifacts (Figure 4B). When a user selects both methods, the data are examined recursively to fulfill both criteria.

Figure 4C displays a close-up of one eye blink from the dataset used for fixation and saccade analysis. Note the loss of eye data during the blink and the increased apparent velocity (exceeding 50 deg/sec) at the onset and offset of the blink. Thus, the eye blink would have both interrupted an ongoing fixation and have been scored as a saccade if not filtered. The result of filtering is shown in Figure 4D. The parts of the eye blink filtered out are colored gray, and double-headed arrows indicate the limits of the blink. This interval either can be marked as missing data or can be interpolated.

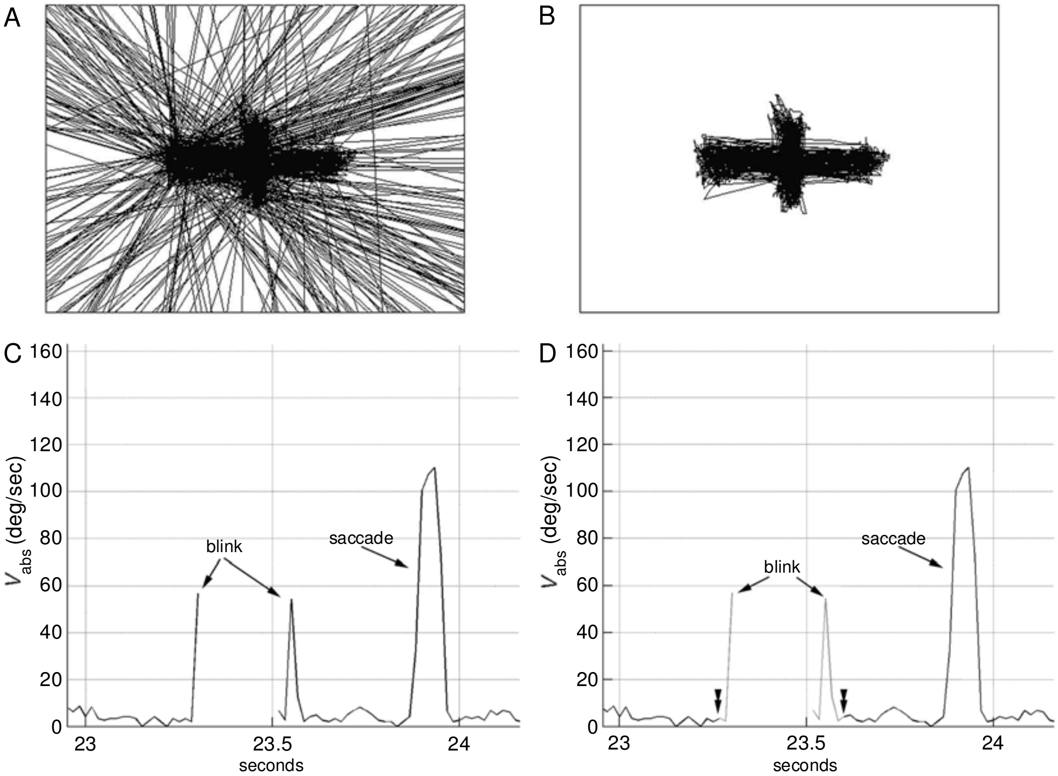


Figure 4. (A) An example of data, from another experiment, heavily contaminated by blink artifacts (data provided by Zina Manjaly, Institute for Medicine, Research Centre, Jülich, Germany). The subject was making horizontal and vertical eye movements on alternate trials. The various lines extending beyond the central dark cross of eye movements all represent blink artifacts. (B) Same data as in panel A, with blink artifacts removed by the blink filter. (C) High-resolution view of a single blink artifact. These data were identical to those used for fixation and saccade analysis. Note the interval of missing data, surrounded by distorted eye position data displaying a high velocity. (D) Same data as those in panel C, showing the artifact to be removed by blink filtering (indicated by the double arrowheads). The gap in eye data can be either left empty or filled in by interpolation. Both options are available in ILAB.

ROI analysis and filtering. Another important analysis included in the ILAB system is the ability to define multiple ROIs and to correlate time, scanpath distance, or fixations with each of the ROIs. Finally, one has the ability to filter (i.e., smooth) the eye movement or pupil size data. Although many infrared eye-tracking systems filter the eye movement data automatically, most do not filter pupil size, even though these data can be quite noisy. Simple filters are included with ILAB, and the user can add additional filters.

Development

One goal of the ILAB system is to enable users to add their own tools. This has been enabled in several ways. (1) ILAB's source is open and modifiable by the user. (2) Gateway functions are provided for properly accessing and setting the program's main global variables (ILAB, PLOTPARMS, and ANALYSISPARMS). (3) Finally, a simple toolbox system is provided wherein folders added under the $ILAB/toolbox[2] directory tree are recognized and added to a toolbox submenu upon starting ILAB. For a toolbox to be recognized an m-file has to have the same name as the toolbox and reside within the toolbox's folder. For example, a toolbox called *test* would have the file test.m located within the folder $ILAB/toolbox/test. Upon selection of the toolbox from the menu, its directory path is added to MATLAB's path, and the test.m file is executed. A similar strategy has proven to be successful for implementing add-ons to the SPM software (Friston et al., 1994–2002).

Conclusion

ILAB provides a flexible and powerful software foundation for analyzing eye movements. The program uses a simple, menu-driven, graphical user interface that allows the user maximum flexibility in defining the parameters for analysis and display. Basic eye movement analyses (fixations, saccades, ROI-based) are included in the package, as well as additional methods for conditioning the data, exporting data and results, and annotating the display. The use of the MATLAB software environment affords the user the ability to read and modify the code and the power of all the standard MATLAB tools. ILAB is in daily use in our laboratory (Gitelman et al., 1999; Gitelman, Parrish, LaBar, & Mesulam, 2000). Since its release to the public in early 2002, it has been downloaded by over 100 other laboratories. It is hoped that the open-source model will contribute to the ongoing improvement of the program and its widespread use. ILAB may be accessed via the URL at http://www.brain.northwestern.edu/ilab.

REFERENCES

Brainard, D. H. (1997). The Psychophysics Toolbox. *Spatial Vision*, **10**, 433-436.

Calkins, M. E., Katsanis, J., Hammer, M. A., & Iacono, W. G. (2001). The misclassification of blinks as saccades: Implications for investigations of eye movement dysfunction in schizophrenia. *Psychophysiology*, **38**, 761-767.

Cox, R. W. (1996). AFNI: Software for analysis and visualization of functional magnetic resonance neuroimages. *Computers & Biomedical Research*, **29**, 162-173.

Duchowski, A., Shivashankaraiah, V., Rawls, T., Gramopadhye, A., Melloy, B., & Kanki, B. (2000). Binocular eye tracking in virtual reality for inspection training. In A. T. Duchowski (Ed.), *Eye Tracking Research & Applications (ETRA) Symposium* (pp. 89-96). New York: ACM Press.

Fischer, B., Biscaldi, M., & Otto, P. (1993). Saccadic eye movements of dyslexic adults. *Neuropsychologia*, **31**, 887-906.

Frackowiak, R. S., Friston, K. J., Frith, C., & Dolan, R. (Eds.) (1997). *Human brain function.* San Diego: Academic Press.

Friston, K., Ashburner, J., Heather, J., Holmes, A., & Poline, J.-B. (1994–2002). *Statistical parametric mapping.* Wellcome Department of Imaging Neuroscience. Retrieved from http://www.fil.ion.ucl.ac.uk/spm.

Gitelman, D. R., Nobre, A. C., Parrish, T., LaBar, K. S., Kim, Y.-H., Meyer, J. R., & Mesulam, M. M. (1999). A large-scale distributed network for covert spatial attention: Further anatomical delineation based on stringent behavioral and cognitive controls. *Brain*, **122**, 1093-1106.

Gitelman, D. R., Parrish, T. B., Friston, K. J., & Mesulam, M.-M. (2002). Functional anatomy of visual search: Regional segregations within the frontal eye fields and effective connectivity of the superior colliculus. *NeuroImage*, **15**, 970-982.

Gitelman, D. R., Parrish, T. B., LaBar, K. S., & Mesulam, M.-M. (2000). Real-time monitoring of eye movements using infrared video-oculography during functional magnetic resonance imaging of the frontal eye fields. *NeuroImage*, **11**, 58-65.

Karn, K. S. (2000). "Saccade pickers" vs. "fixation pickers": The effect of eye tracking instrumentation on research. In A. T. Duchowski (Ed.), *Proceedings of the Eye Tracking Research and Applications Symposium* (pp. 87-88). New York: ACM Press.

Karsh, R., & Breitenbach, F. W. (1983). Looking at looking: The amorphous fixation measure. In R. Groner, C. Menz, D. F. Fisher, & R. A. Monty (Eds.), *Eye movements and psychological functions: International views* (pp. 53-64). Hillsdale, NJ: Erlbaum.

LaBar, K. S., Mesulam, M.-M., Gitelman, D. R., & Weintraub, S. (2000). Emotional curiosity: Modulation of visuospatial attention by arousal is preserved in aging and early-stage Alzheimer's disease. *Neuropsychologia*, **38**, 1734-1740.

Land, M. F., & Lee, D. N. (1994). Where we look when we steer. *Nature*, **369**, 742-744.

Pelli, D. G. (1997). The VideoToolbox software for visual psychophysics: Transforming numbers into movies. *Spatial Vision*, **10**, 437-442.

Rayner, K. (1995). Eye movements and cognitive processes in reading, visual search and scene perception. In J. M. Findlay, R. Walker, & W. Kentridge (Eds.), *Eye movement research: Mechanisms, processes and applications. Studies in visual information processing* (Vol. 6, pp. 3-22). New York: Elsevier.

Salvucci, D. D., & Goldberg, J. H. (2000). Identifying fixations and saccades in eye-tracking protocols. In A. T. Duchowski (Ed.), *Proceedings of the Eye Tracking Research and Applications Symposium* (pp. 71-78). New York: ACM Press.

Suppes, P. (1990). Eye-movement models for arithmetic and reading performance. In E. Kowler (Ed.), *Eye movements and their role in visual and cognitive processes* (pp. 455-477). New York: Elsevier.

Widdel, H. (1984). Operational problems in analysing eye movements. In A. G. Gale & F. Johnson (Eds.), *Theoretical and applied aspects of eye movement research* (pp. 21-29). New York: Elsevier.

Williams, S. (2002). *Free as in freedom*. Cambridge, MA: O'Reilly.

Wooding, D. (1999). *Eye movement equipment database.* Retrieved June 14, 2002, from http://ibs.derby.ac.uk/emed/.

NOTES

1. The notation $n \times 4$ refers to n- rows and 4 columns.

2. $ILAB refers to the location of the main ILAB folder within the directory tree.

(Manuscript received July 15, 2002;
revision accepted for publication October 15, 2002.)