Gazepaths: Using Gaze-based Target-selection Tasks for Tuning Parameters in Scanpath Algorithms

I. Scott MacKenzie
mack@yorku.ca
Dept. of Electrical Engineering and Computer Science,
York University
Toronto, Canada

Maria Francesca Roig-Maimó
Ramon Mas-Sansó
xisca.roig@uib.es,ramon.mas@uib.es
Department of Mathematics and Computer Science
University of the Balearic Islands
Palma, Spain

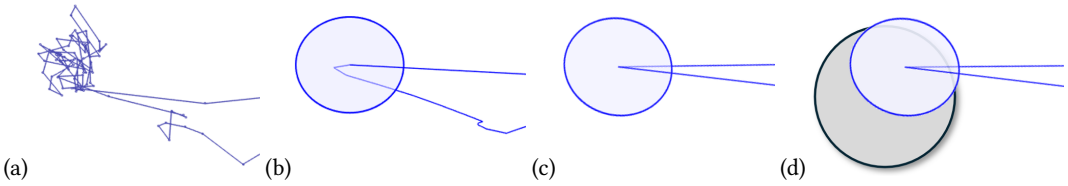

Figure 1: Evolution of a gazepath: (a) raw data from eye tracker, (b) raw data reduced to a fixation and saccade points, (c) saccade points rendered as straight lines between fixations, and (d) gazepath, showing a fixation (with in-going and out-going saccade) superimposed on a UI target the user is selecting via a dwell-time criterion.

Abstract

In eye-tracking research, visualizations of eye movements are commonly rendered as "scanpaths" with movement patterns reduced to fixations and saccades. Although the eyes are crucial for sensory input, some eye-tracking applications use the eyes to both sense and control a user interface, with "eye typing" a typical example. Used in this way, the eyes perform "double-duty," so to speak. We advance the idea that more information is available in double-duty mode since the location of selections (i.e., fixations) is known in advance. This is not so when the eyes are only a sensory input channel, since the eye movements are more casual, even serendipitous. Using data from an eye-tracking experiment using dwell-time target selection, we demonstrate that this added information provides a novel resource to adapt and fine-tune the algorithm that converts raw data to fixations and saccades. The result is a "gazepath" visualization.

Keywords

Eye tracking, scanpath, fixation, saccade, sensitivity analysis

ACM Reference Format:
I. Scott MacKenzie, Maria Francesca Roig-Maimó, and Ramon Mas-Sansó. 2026. Gazepaths: Using Gaze-based Target-selection Tasks for Tuning Parameters in Scanpath Algorithms. In . ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Conference'17, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Most eye-tracking research focuses on the eyes as a sensory input channel, with the user viewing content on a computer display or artifacts in the environment. Data are collected, patterns are analyzed. The research questions, broadly speaking, are *where* is the user looking and *what* are the strategies for scanning [5, p. 11]? However, a subset of eye-tracking research positions the eyes in a double-duty role, serving both as a sensory input channel and as a mechanism for output control [17, Figure 2].[1] What is traditionally an "AOI" or "area of interest" becomes an "SOI" or "selectable object of interest" [18, Figure 3.12]. The user views content on a computer display with the capability to both sense the interface and control the interaction. Buttons, keys, icons, or other GUI components are activated and selected via the eyes as the user performs an interaction task.

A well-researched example with the eyes in double-duty mode is "eye typing" [7, 22]. Using their eyes, the user gazes at keys on a soft keyboard and selects keys—with their eyes—to create a text message (e.g., [8, 20, 21, 25]). Besides eye typing, the eyes can also perform general-purpose point-select operations, in effect, operating as a mouse-emulation device for interactive systems [1, 31].

The present research explores how double-duty eye tasks support the creation of visualizations of eye movements. Such visualizations are commonly called "scanpaths." Here, we introduce "gazepaths" as an alternative term when the data originate from interactions where the eyes are in double-duty mode. To be clear, the visualizations are the same for gazepaths and scanpaths. The difference we advance is that double-duty eye interactions provide information not available in single-duty mode. This information supports a better and more informed use of the algorithms and their parameters to convert

[1] As used here, the terms "input" and "output" are with respect to the human. When describing the human-machine interface in HCI or human factors, the terms "input" and "output" are with respect to the machine.

raw data to fixations and saccades. A process that is similar to discussions herein is calibration, where users are explicitly required to fixate on certain screen locations, typically at the corners [10]. However, fixations for calibrating a system do not move forward into scanpath visualizations.

2 Background

Early and influential work on eye tracking and creating visualizations of eye movements was presented by Yarbus in the 1960s [27, 30]. Yarbus instrumented the user's eyes with small cups resembling contact lenses to monitor eye position. As noted, "a lever or thread was attached to the cup by means of which eye movements were transmitted to the recording system" [30, p. 21]. Users were asked to view a scene, for example, in a painting, and asked to perform a task, such as "estimate the ages of people in the painting." Images were made of the eye movements. He reported differences in the movement patterns between and within participants as well as changes over time and in subsequent viewings. In the images (a.k.a. scanpaths), fixations were shown as points and saccades as lines [27, Figure 5]. Later work changed the depiction of fixations from points to circles with the diameter increasing with longer durations (e.g., [21, Figure 2] [11, Table 7]) or larger diameters (e.g., [16, Figure 3]). Most related work in this vein is motivated to measure eye movement patterns to understand a person's "scanning strategy" [6, p. 203]; that is, where they are looking and why. This is typical for the single-duty use of the eyes noted above.

In double-duty mode, the movement patterns for the eyes are different from the patterns in single-duty "scanning" mode. The user is not simply scanning: There is an intent to interact and control the system—and to do so via the eyes. Thus, new forms of eye movement occur. This particular point has not been explored in previous research. Small adjustments to the gaze point may occur intentionally to "acquire" SOIs. This is equivalent to acquiring "focus" in traditional GUIs. An SOI might be acquired simply to observe, as in a popup tooltip dialog. If a cursor is present, intentional eye movements might seek to adjust the cursor's position in or around an SOI, again, simply to observe. Even acquiring or gaining focus of a GUI component is distinctly different from simple scanning, as in reading online news (perhaps with embedded ads).

Beyond acquiring focus, a common intent is selection. In this case, further engagement following focus is needed and, again, unique patterns of eye movement emerge. Since eye input lacks an inherent confirmation mechanism, a creative method for selection is required. Maintaining focus with an SOI for a prescribed dwell-time is the most common selection method, dating to at least 1990 [14]. Other selection methods are also used, such as blink selection [19, 29], brain signals [12], or using a separate channel for input, such as touch [28] or speech [26].

Some interactive tasks involve dragging or state-2 interaction [2, Figure 1]. These require a transition from pointing (state-1) to dragging (state-2) wherein movements group objects or perhaps create a stroke or gesture. For a touch-sensing surface, a stylus or finger is commonly used; but with eye tracking, the state-1 to state-2 transition requires different actions such as framing the eye-gesture with a wink [4] or a separate multi-modal action [15].

In general, eye movements for controlling a UI are feedback-driven and less ballistic than movements simply to observe. Movements are intentional, guided, and adjusted according to how the UI reacts and presents the system state. The interactions just described fall outside the scope of simple scanning. And so, visualizing the eye movements through gazepaths has different needs and objectives. Fixations are often "hesitations," performed to maintain or slightly shift in the gaze point position while observing the effect. Saccades are "displacements," "adjustments," or "shifts" where the previous action is complete and a new purpose is sought. Gazepoint adjustments within an SOI may spawn a saccade, but the intent is interaction, not scanning.

The work presented herein is analytical, not experimental. An experimental comparison of eye-movement patterns for gazepaths (double-duty) vs. scanpaths (single-duty) is fraught with design problems because the tasks are inherently different—much different. Thus, confounding variables are unavoidable: Are the differences observed due to the single-duty vs. double-duty role of the eyes or to task differences? Because of this, the focus in this short paper is simply to expose and examine eye movements that occur when the eyes are acting both for sensory input and control output. This is done through visualizations for eye movements.

2.1 Generating Scanpaths (or Gazepaths) from Raw Data

Converting raw eye tracking data to scanpath data begins with a series of x-y samples corresponding to the position on a display of the user's eyes while doing a task. Timestamps (t) are also required. The seminal paper on conversion algorithms is by Salvucci and Goldberg [24], where the problem space is explored, caveats are offered, and methods are described [24, Table 7]. The algorithm used herein is loosely based on the velocity threshold identification (I-VT) method [24, Table 2]. Briefly, the steps are

(1) Smooth the data, for example, using a sliding "mean" window of a specified size.
(2) Compute the velocity for each sample based on the previous sample by dividing the Pythagorean distance by the time.
(3) Based on velocity, flag each sample as above or below a specified velocity threshold. This step tentatively distinguishes a fixation sample from a saccade sample.
(4) Scan for consecutive sequences of "fixation" samples of length less than a specified minimum. Change the status of samples in such sequences to "saccade."
(5) Return the scanpath data as groups alternating between consecutive samples for fixations and saccades.

Notice three uses of "specified" above. These are the parameters that control the conversion from raw data to scanpath data. The parameters and nominal values are as follows:

- WS (window size) – size of sliding "mean" window for smoothing data ($\text{WS}_{\text{nominal}} = 7$)
- VT (velocity threshold) – velocity threshold below or above which samples are tentatively flagged as fixations or saccades, respectively ($\text{VT}_{\text{nominal}} = 300$ pixels per second)
- ML (minimum length) – for consecutive samples classified as a fixation ($\text{ML}_{\text{nominal}} = 15$)

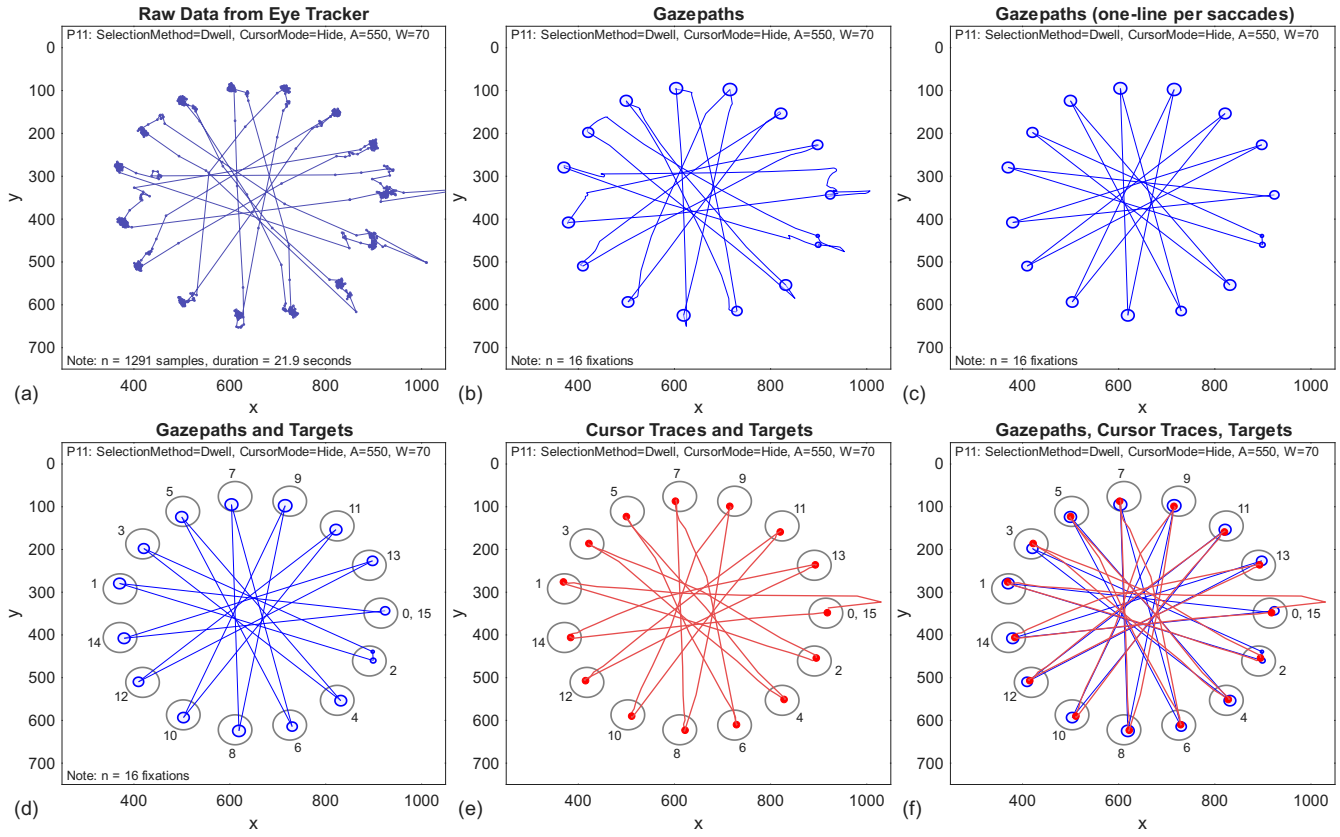

Figure 2: Example from a 2D Fitts' law task with 15 target selections, known as a "sequence." (a) Line plot for eye movements based on raw data from the eye tracker. (b) Raw data reduced to gazepaths showing 16 fixations. (c) Gazepaths with saccades rendered as straight lines between fixations. (d) Gazepaths superimposed on targets as they appeared in the UI. (e) Targets, cursor path, and selection points, as recorded by the experimental app. (f) Gazepaths (blue) with cursor traces (red) and targets.

An alternative to smoothing the data first, then computing the sample-to-sample velocity, is to compute the average velocity over a window of n raw samples and compare that value to a velocity threshold [5, p. 145].

In eye tracking research, velocities are often measured in degrees per second to reflect angular movement of the eyes. Converting from pixels to degrees requires information on the apparatus and setup [9, Figure 2.4]. In our setup with display width = 50.3 cm, display width resolution = 1280 pixels, and a viewing distance = 60 cm, the visual angle for the width of the display is $2 \times \arctan(\frac{50.3}{2}/60)$ = 45.5 degrees, and therefore $\frac{45.5}{1280}$ = 0.0355 degrees per pixel, equivalent to $\frac{1280}{45.5}$ = 28.3 pixels per degree in the reciprocal form. With this, the nominal velocity threshold cited above in pixels per second is $\frac{300}{28.3}$ = 10.7 degrees per second. Salvucci and Goldberg [24] cite 20 degrees per second as an example, while also noting that exploration is needed to find a suitable value [24, p. 73].

The minimum number of consecutive samples per fixation depends on the sampling rate for the raw data. The data analysed herein are from a Tobii Pro Nano sampling at 60 Hz or 0.0167 seconds per sample. Thus, the nominal ML equates to a minimum fixation duration of $15 \times 0.0167 = 0.251$ s = 251 ms, which is in the 200-400 ms range cited by Salvucci and Goldberg [24, p. 72].

If the data are returned as arrays of t-x-y values for each fixation and saccade, the x-y center of each fixation is simply the mean of the x-values and y-values. The duration is computed from the t-values. The diameter of the fixation circle in the visualization is set using a scaling factor multiplied by the number of samples or the duration of the fixation. Saccades are typically reduced to straight lines from the center of one fixation to the next (e.g., [21, Figure 2]).

2.2 Example

An example of the conversion from raw eye tracking data to gazepath data is shown in Fig. 2. The data are from an experiment using eye input in a 2D Fitts' law target-selection task [23]. Fifteen targets were arranged in a layout circle, as per ISO 9241-411 [13]. Target selection required a 600-ms dwell-time.

Note in Fig. 2a that the raw data included 1291 sample points covering 21.9 seconds of interaction. The algorithm reduced these data to 16 fixations, as seen in Fig. 2b. The saccade points are reduced to straight lines between fixations in Fig. 2c. The result is a near-perfect rendering of the expected user behaviour, with the

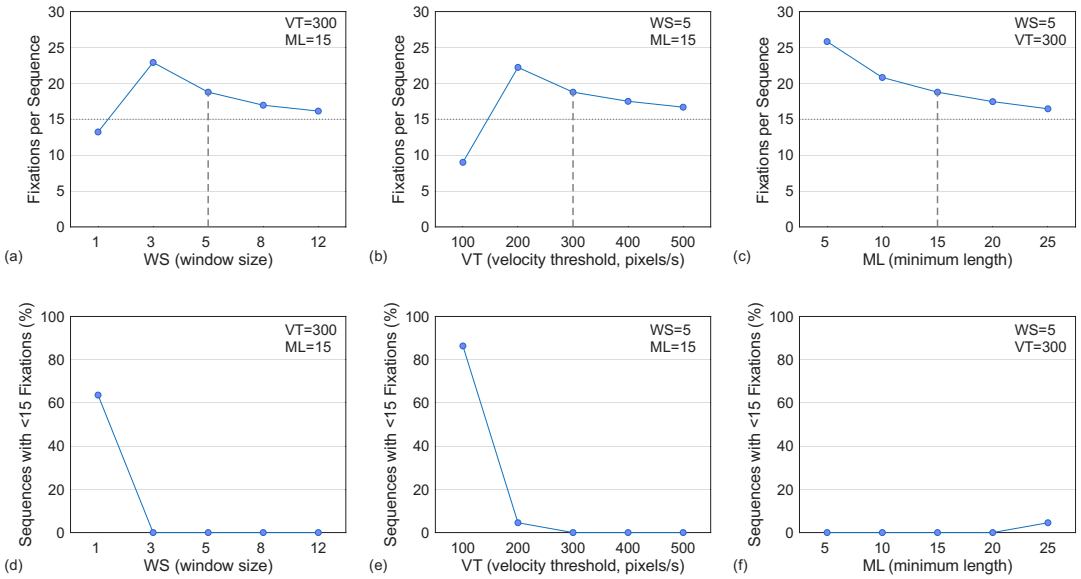

Figure 3: Sensitivity analysis on the algorithm to convert raw data to gazepath data. The algorithm's three parameters appear in columns: window size (left), velocity threshold (middle), and minimum length (right). Five values were tested for each. Two output summary values were calculated: fixations per sequence (top row) and sequences with less than 15 fixations (bottom row). See text for discussion.

minor exception of two fixations for target #2.[2] Since dwell-time selection was used, errors were not possible, so each fixation is on a target, as seen in Fig. 2d. Moreover, there *must be* at least one fixation on each target.[3] Knowledge of this is useful in finding appropriate values for the algorithm's parameters in creating the gazepaths. Such knowledge is not available when the eyes are for sensory input only, since the user's intent is less specific. Fig. 2e is based only on data from the experiment application. Red lines and circles are the cursor path and selection points, respectively, recorded from "mouse events." Fig. 2f combines Fig. 2d and Fig. 2e, showing the gazepath (blue) and cursor path (red).

3 Sensitivity Analysis

How sensitive is the algorithm to parameter values? That is, how do changes in the parameters affect the data returned and the gazepath visualizations of the interaction? This question is critical to both using the algorithm and improving it through refinements. Performing a sensitivity analysis, as presented here, is a practice in HCI dating to the field's inception in the early 1980s and as demonstrated in research on Card, Moran, and Newell's keystroke-level model [3, Section 5.3].

For our analysis, we used 22 data sets (11 participants, 2 sequences each). Each data set contained raw eye tracking data for the selection of 15 targets in sequence (as per Fig. 2a). The data sets ranged in size from 1060 to 1840 samples (M = 1475). With a 60 Hz sampling rate, the mean duration was $1475 \times \frac{1}{60} = 24.5$ seconds. Five values were tested for each parameter in the algorithm: WS = {1, 3, 5, 8, 12}, VT = {100, 200, 300, 400, 500}, and ML = {5, 10, 15, 20, 25}. The middle entry in each set is the nominal value, as used in Fig. 2. With these test values, there were $5 \times 5 \times 5 \times 22 = 2750$ runs of the analysis.

The sensitivity analysis generated two test calculations: (i) the number of fixations per sequence, and (ii) the percentage of sequences with fewer than 15 fixations. These are important for the health of the algorithm, since there is a requirement for exactly 15 target selections in each data set. So, the number of fixations should be about 15, perhaps a few more, but definitely no less.

A summary of the results is in Fig. 3. Note that the test for each parameter included the other two parameters set to their nominal value. The results are organized in two rows, with "fixations per sequence" along the top row and "sequences with fewer than 15 fixations" along the bottom row. The three columns are for the algorithm's parameters: WS, VT, and ML. The nominal values—used in Fig. 2—are indicated with dashed vertical lines.

There are a few observations to make in Fig. 3. First, data smoothing is clearly necessary. With no smoothing (WS = 1), most fixations are missed. The same is true for VT = 100, where the threshold for the velocity of samples is so low that few consecutive sequences meet the criteria for a fixation. Of course, this is with nominal ML = 15. A lower ML would increase the number of fixations, but the effect would likely be minimal. For all remaining parameter values along the top row, the number of fixations per sequence is above 15. So, all is well.

Along the bottom row in Fig. 3 we see that the nominal combination of parameter values (middle value in each chart) yields 0

[2] See Fig. 2d for the target order in a sequence. A sequence began with the user selecting target #0. Selections proceeded across and around the layout circle, finishing with a final selection at target #15 (same as target #0).

[3] Given the size of targets (70 px or 2.47° of visual angle) and the 600-ms dwell-time criterion for selection, it is highly unlikely that a target could be selected in the absence of a fixation.

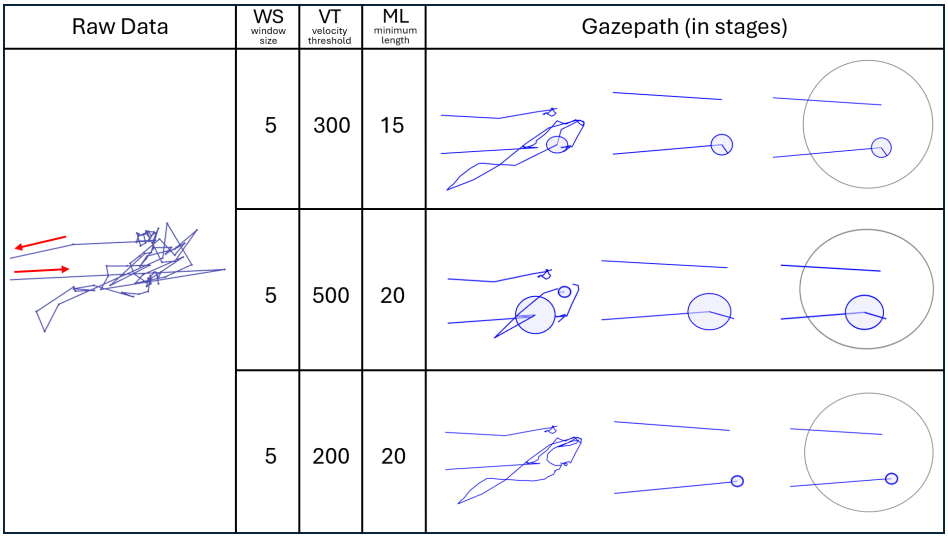

Figure 4: Sensitivity analysis example. The conversion from raw data (left) to a gazepath (right) depends on the algorithm's parameters: WS (window size for sliding 'mean' smoothing), VT (velocity threshold in pixels per second), and ML (minimum length for consecutive samples in a fixation). See text for discussion.

sequences with fewer than 15 fixations. This is an important outcome. Notwithstanding the anomalous results with WS = 1 and VT = 100, slight concerns arise at VT = 200, and ML = 25 (see figure). In these cases, some sequences have trials mis-categorized as lacking a fixation—a result that is simply wrong. Overall, the nominal values seem appropriate.

Next, we offer two examples of how parameter values impact the gazepath visualization. Each begins with raw eye-tracking data for the selection of a single target among the 15 in a sequence. The first example is in Fig. 4. A segment of raw data from the eye tracker is shown on the left. Parameter values are shown next, followed by three results as gazepaths, organized as in Fig. 1. In the first result (top row), the parameters are set to the nominal values used in Fig. 2: WS = 5, VT = 300, and ML = 15. The data here are from a different participant, however. The fixation is for target #15, the last target in the sequence. Above the fixation, is a saccade with eye movement to the left, indicated by the top red arrow. That saccade occurred at the beginning of the sequence, from target #0 (a.k.a. target #15) to target #1 on the left side of the layout (not shown). The bottom red arrow is for the in-coming saccade at the end of the sequence; it ends at the final fixation (shown) to select the target.

The second row sees a small change in the algorithm's parameters. VT is increased to 500 pixels/s. This change allows more sample points to be classified as fixation points, since fixation points are those with a velocity *below* VT. However, ML is increased from 15 to 20. This has an opposing effect, since a longer sequence of consecutive "fixation" points is required for the points to maintain their fixation status. Evidently, in this case, VT has a stronger effect than ML, since the size of the fixation circle is larger in the middle row compared to the top row.

Moving from the middle row to the bottom row in Fig. 4, the only change is VT dropping from 500 to 200 pixels per second. The expected effect is to reduce the number of fixation points since the requirement is more stringent (i.e., the upper limit on velocity is less). Indeed, this occurred. The fixation circle is small, due to re-classifying some points as saccade points.

A second example is shown in Fig. 5, for a participant selecting target #8 in the sequence. Along the rows, three combinations of parameter values are shown. Note in the top row that two fixations were found in the raw data when the user's gaze settled on or near the target. In the middle row, a slight adjustment to VT and ML recast the data to a single fixation. The adjustment in the third row is a problem. Reducing WS to 3 (the window size for smoothing the data) results in the all data points being classified as saccade points. Consequently, no fixation was found ("*empty*" in the figure). Of course, this interpretation is wrong since the user successfully selected the target by maintaining their gaze point within the target for at least 600 ms. This is yet another example of how creating a visualization for gazepaths (with the eyes in double-duty mode) is different from the same process when the eyes serve only as sensory input. In double-duty mode, the user's intent is known. Furthermore, it is known not just in a general sense, as in "selecting targets," but with a specific intent, as in "selecting target *n*." In this example (3rd row in Fig. 5), it is unequivocal that the algorithm failed and adjustments are warranted. The same insight is not available with single-duty eye-tracking, since the user's intent is more fluid.

4 Discussion

The sensitivity analysis and examples above are a sample from a larger set of analyses. By and large, a full account leads to observations and results consistent with those above. Had the analyses used a different algorithm (e.g., as per Salvucci and Goldberg [24]), it is our view that a similar narrative holds. To summarize, the narrative advanced herein is that the observations above (e.g., *most fixations were missed*) are only available for eye-tracking applications if the eyes are in double-duty mode where the task primitives are known.

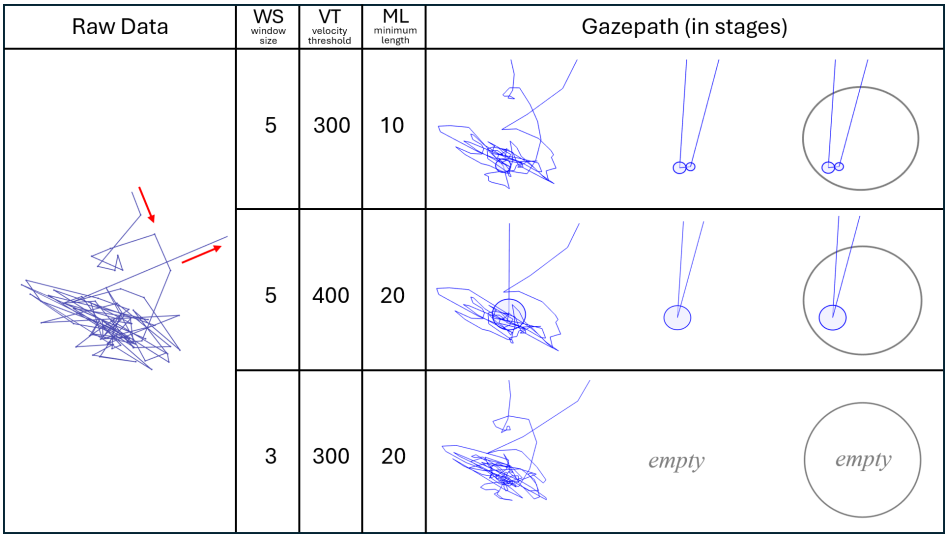

Figure 5: Sensitivity analysis example, showing raw data (left), parameter values (middle), and gazepath generation (right). See text for discussion.

The 2D Fitts' law task is highly scripted. After selecting target #0, the user proceeds expeditiously and sequentially to select all targets in sequence. This predictability facilitates creating the gazepath and verifying the health of the fixation-identification algorithm, as outlined above. However, other interactive tasks are less accommodating. For eye typing, users select "keys" in a predictable sequence to add letters to a message. However, there is also discretion, as the user may gaze off the keyboard to verify the message. Such interactions appear as "read text events" in the gazepath [21, Figure 2]. In view of this, an area for future work on double-duty eye interaction is to analyse the experiment task and to cite and distinguish selections and fixations that *must* occur from those that *might* occur. A gazepath visualization that omits a must-occur fixation is a red flag that the algorithm or its parameters need attention. Of course, this process is not possible for eye-tracking applications limited to simple scanning, since scanning follows the user's intuition rather than task-specific actions.

Guidelines for applying the ideas herein must be qualified, as there is no one-size-fits-all for a scanpath algorithm. Appropriate parameter values will vary with the sampling rate, noise, resolution, and other properties of the host system and eye tracking apparatus. Furthermore, what constitutes a fixation and where it begins and ends depends on the concurrent cognitive processing; that is, what the user is doing and thinking [24, p. 71]. So, guidelines are not rules; they must be tempered for the context of the interaction. Having said that, for dwell-time selection, we recommend using a value between 1× and 2× the expected duration of a fixation. So, to accurately find fixations of, say, 300 ms duration, the dwell-time interval should be at least 300 ms and not more than 600 ms. With this, one, and only one, fixation is possible for a successful target selection. Similarly, if an area-based or dispersion-based algorithm is used [24, Table 1], targets should be sized with 50% to 100% more area than the expected or required area of a fixation. See Fig. 1d for an example.

Finally, we note that the 2D Fitts' law task used herein is just an example, leveraging data from a separate user study [23]. A task with targets evenly distributed on the display might be better, with the proviso that there is a prescribed order of selection. The order should be known the user, with selections proceeding expeditiously and without distractions or side-tasks. This is needed to provide predictability in user behaviour to support the method described herein.

5 Conclusion

In this short paper, we introduced a novel detail in creating visualizations of eye movements—known as scanpaths—from raw eye-tracking data. The algorithms require considerable effort to fine tune the parameters, with the added difficulty that there is no objective way to validate them in contexts where the eyes are simply scanning content on a display. In this work, we provide an approach to verify the health of fixation-detection algorithms when the eyes work in double-duty mode; that is, for sensory input and output control. Eye typing is a well-known example, general-purpose target selection another. When used in this manner, details of where the user is looking are known in advance. This provides additional information not available in simple scanning—information that is useful for fine tuning the algorithm and building an accurate visualization of patterns of eye movement. We use the term "gazepath" for visualizations in this vein.

References

[1] Per Bækgaard, John Paulin Hansen, Katsumi Minakata, and I. Scott MacKenzie. 2019. A Fitts' law study of pupil dilations in a head-mounted display. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications – ETRA '19.* ACM, New York, 32.1–32.5. doi:10.1145/3314111.3319831

[2] William Buxton. 1990. A three-state model of graphical input. In *Proceedings of the IFIP TC13 Third International Conference on Human-Computer Interaction – INTERACT '90.* Elsevier, Amsterdam, 449–456. https://billbuxton.com/3stateModel.pdf

[3] Stuart K. Card, Thomas P. Moran, and Allen Newell. 1980. The keystroke-level model for user performance time with interactive systems. *Commun. ACM* 23, 7

(1980), 396–410. doi:10.1145/358886.358895

[4] Tafadzwa Joseph Dube, Qhelile Ozias Shibanda, I. Scott MacKenzie, and Ahmed Sabbir Arif. In press. WinkSwype: Comparing controller, raycast, and a novel wink-based method for gesture typing in virtual reality. In *Proceedings of Graphics Interface – GI '26*. Canadian Human-Computer Communications Society (CHCCS), Toronto.

[5] Andrew T. Duchowski. 2017. *Eye tracking methodology: Theory and practice* (3rd ed.). Springer, London. doi:10.1007/978-3-319-57883-5

[6] Joseph H. Goldberg and Jonathan I. Helfman. 2010. Visual scanpath representation. In *Proceedings of the 2010 Symposium on Eye Tracking Research & Applications – ETRA '10*. ACM, New York, 203–210. doi:10.1145/1743666.1743717

[7] Aleesha Hamid and Per Ola Kristensson. 2024. 40 years of eye typing: Challenges, gaps, and emergent strategies. In *Proceedings of the 2024 Symposium on Eye Tracking Research & Applications – ETRA '24*. ACM, New York, 222.1–222.19. doi:10.1145/3655596

[8] Katarzyna Harezlak, Pawel Basek, and Pawel Kasprowski. 2022. Side keyboard: The new approach for eye-typing. In *Proceedings of the 26th International Conference on Knowledge-based and Intelligent Information & Engineering Systems – KES '22*. Procedia Computer Science, Amsterdam, 3348–3357. doi:10.1016/j.procs.2022.09.393

[9] Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost van de Weijer. 2011. *Eye tracking: A comprehensive guide to methods and measures*. Oxford University Press, Oxford, UK. https://books.google.ca/books?id=5rIDPV1EoLUC

[10] Anthony J. Hornof and Tim Halverson. 2002. Cleaning up systematic error in eye-tracking data by using required fixation locations. *Behavior Research Methods, Instruments, & Computers* 34 (2002), 592–604. Issue 4. doi:10.3758/BF03195487

[11] Anthony J. Hornof and Tim Halverson. 2003. Cognitive strategies and eye movements for searching hierarchical computer displays. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI '03*. ACM, New York, 249–256. doi:10.1145/642611.642656

[12] Baosheng James Hou, John Paulin Hansen, Cihan Uyanik, Per Bækgaard, Sadasivan Puthusserypady, Jacopo M. Araujo, and I. Scott MacKenzie. 2022. Feasibility of a device for gaze interaction by visually-evoked brain signals. In *Proceedings of the 2022 Symposium on Eye Tracking Research & Applications – ETRA '22*. ACM, New York, 62.1–62.7. doi:10.1145/3517031.3529232

[13] ISO. 2012. *Ergonomics of human-system interaction — Part 411: Evaluation methods for the design of physical input devices*. Report Number ISO/TS 9241-411:2012. International Organisation for Standardisation, Geneva, Switzerland. https://www.iso.org/standard/54106.html Last reviewed and confirmed in 2022.

[14] Robert J. K. Jacob. 1990. What you look at is what you get: Eye movement-based interaction techniques. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI '90*. ACM, New York, 11–18. doi:10.1145/97243.97246

[15] Chandan Kumar, Ramin Hedeshy, I. Scott MacKenzie, and Steffen Staab. 2020. TAGSwipe: Touch assisted gaze swipe for text entry. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI '20*. ACM, New York, 1–12. doi:10.1145/3313831.3376317

[16] Chris Lankford. 2000. Gazetracker: Software designed to facilitate eye movement analysis. In *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications – ETRA '00*. ACM, New York, 51–55. doi:10.1145/355017.355025

[17] I. Scott MacKenzie. 2012. Evaluating eye tracking systems for computer input. In *Gaze interaction and applications of eye tracking: Advances in assistive technologies*, P. Majaranta, H. Aoki, M. Donegan, D. W. Hansen, J. P. Hansen, A. Hyrskykari, and K.-J. Räihä (Eds.). IGI Global, Hershey, PA, 205–225. doi:10.4018/978-1-61350-098-9.ch015

[18] I. Scott MacKenzie. 2024. *Human-computer interaction: An empirical research perspective* (2nd ed.). Morgan Kaufmann (an imprint of Elsevier), Cambridge, MA. doi:10.1016/C2022-0-02755-0

[19] I. Scott MacKenzie and Behrooz Ashtiani. 2011. BlinkWrite: Efficient text entry using eye blinks. *Universal Access in the Information Society (UAIS)* 10 (2011), 69–80. doi:10.1007/s10209-010-0188-6

[20] I. Scott MacKenzie and Xuang Zhang. 2008. Eye typing using word and letter prediction and a fixation algorithm. In *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications – ETRA '08*. ACM, New York, 55–58. doi:10.1145/1344471.1344484

[21] Päivi Majaranta, I. Scott MacKenzie, Anne Aula, and Kari-Juoko Räihä. 2006. Effects of feedback and dwell time on eye typing speed and accuracy. *Universal Access in the Information Society (UAIS)* 5 (2006), 199–208. doi:10.1007/s10209-006-0034-z

[22] Päivi Majaranta and Kari-Jouko Räihä. 2002. Twenty years of eye typing: Systems and design issues. In *Proceedings of the 2002 Symposium on Eye Tracking Research & Applications – ETRA '02*. ACM, New York, 15–22. doi:10.1145/507072.507076

[23] Maria Francesca Roig-Maimó, Ramon Mas-Sansó', and I. Scott MacKenzie. In press. A standardized comparison of blink, wink, and dwell as selection methods for gaze-based interaction. In *Proceedings of the ACM Symposium on Eye Tracking Research & Applications – ETRA '26*. ACM, New York.

[24] Dario D. Salvucci and Joseph H. Goldberg. 2000. Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications – ETRA '20*. ACM, New York, 71–78. doi:10.1145/355017.355028

[25] Sayan Sarcar, Prateek Panwar, and Tuhin Chakraborty. 2013. EyeK: An efficient dwell-free eye gaze-based text entry system. In *Proceedings of the 11th Asia Pacific Conference on Computer Human Interaction – APCHI '13*. ACM, New York, 215–220. doi:10.1145/2525194.2525288

[26] Korok Sengupta, Sabin Bhattarai, Sayan Sarcar, I. Scott MacKenzie, and Steffen Staab. 2020. Leveraging error correction in voice-based text entry by talk-and-gaze. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI '20*. ACM, New York, 1–11. doi:10.1145/3313831.3376579

[27] Benjamin W. Tatler, Nicholas J. Wade, Hoi Kwan, John M. Findlay, and Boris M. Velichkovsky. 2010. Yarbus, eye movements, and vision. *i-Perception* 1, 1 (2010), 7–27. doi:10.1068/i0382

[28] Colin Ware and Harutune H. Mikaelian. 1987. An evaluation of an eye tracker as a device for computer input. In *Proceedings of the CHI+GI Conference on Human Factors in Computing Systems – CHI+GI '87*. ACM, New York, 183–188. doi:10.1145/30851.275627

[29] Yushi Wei, Rongkai Shi, Sen Zhang, Anil Ufuk Batmaz, Pan Hui, and Hai-Ning Liang. 2025. Reevaluating the gaze cursor in virtual reality: A comparative analysis of cursor visibility, confirmation mechanisms, and task paradigms. *IEEE Transactions on Visualization and Computer Graphics* 0, 1 (2025), 1–13. doi:10.1109/TVCG.2025.3622042

[30] Alfred L. Yarbus. 1967. *Eye movements and vision*. Springer, New York. https://books.google.ca/books?id=kRf3BwAAQBAJ&l Translation by Basil Haigh.

[31] Xuan Zhang and I. Scott MacKenzie. 2007. Evaluating eye tracking with ISO 9241 – Part 9. In *Proceedings of HCI International – HCII '07 (LNCS 4552)*. Springer, Heidelberg, 779–788. doi:10.1007/978-3-540-73110-8_85