Comparative analysis of cognitive load in cloud provider consoles: an eye-tracking study

Elizaveta A. Shokhina
bluewoolcaterpillar@gmail.com
HSE University
Saint Petersburg, Russia

Abstract

Eye-tracking offers an indirect but objective measurement of cognitive load and is widely sought in HCI research. In practice, however, access to an eye-tracker does not entail access to a research platform: the software needed to run experiments is often sold as a separate paid licence, and without it every study begins with assembling the stimulus application, synchronising recording, and implementing the signal-processing pipeline from scratch. This work develops a software platform for cognitive-load experiments built on the Gazepoint GP3 eye-tracker. The platform combines an interactive web stimulus with synchronised recording of gaze, screen, and behavioural markers; a pupil-signal preprocessing pipeline following current pupillometry standards [22, 19]; and a library that computes oculomotor, pupillary, and behavioural metrics together with group-level statistical tests. The platform was validated in a pilot study with $N = 10$ IT professionals experienced in cloud infrastructure: the full cycle runs in a single session of about 40 minutes per participant, and the resulting metrics are interpretable and mutually consistent. As a pilot case, we tested whether progressive disclosure in a cloud-management console reduces cognitive load compared with a flat layout; in this sample, progressive disclosure produced no robust benefit for expert users, and on search tasks the trends favoured the flat layout — a result that runs against common UX heuristics. Methodological limitations of the current version (no frame-level pupil correction for stimulus luminance, no psychometric evaluation of tasks) and directions for further development are discussed. The source code and documentation are published on GitHub.

CCS Concepts

• **Human-centered computing** → **User studies**; **HCI design and evaluation methods**; *Empirical studies in HCI*.

Keywords

research tool; eye-tracking; Gazepoint GP3; cognitive load; pupillometry; progressive disclosure; NASA-TLX

1 Introduction

1.1 The problem: design decisions and their objective validation

Modern high-information-density interfaces — administrative interfaces, cloud-infrastructure management consoles, analysts' workstations, medical systems — regularly confront designers with a choice between several patterns for presenting information: which data to show immediately and which to tuck away in collapsed sections; which hierarchy of attributes reduces visual noise; in which tasks grouping elements helps the user, and in which it works against them.

These questions are traditionally resolved through usability heuristics, design reviews, and the designer's individual experience, sometimes supplemented by A/B testing or interviews [24]. None of these approaches gives direct access to the user's cognitive load — the key construct through which contemporary HCI literature links interface design with the quality of work carried out in it [14, 36].

Eye-tracking is an objective method of indirectly measuring cognitive load that captures it from several angles: through oculomotor indicators (fixations, saccades, the scanpath) and through pupil diameter as an indicator of the mental effort exerted [29, 40, 22]. Combined with behavioural data and self-report methods, eye-tracking makes it possible to triangulate the measurement of cognitive load within a single session [17, 6].

1.2 The problem: the absence of ready-made research platforms

A study design that models the behaviour of a user working with a cloud-provider console interface requires recording oculomotor data during interaction with a prototype: the participant clicks buttons, scrolls the page, expands sections, and navigates between resources, rather than viewing pre-prepared images or videos. To record analysable data under such conditions, a three-link tool is needed: a stimulus web application, synchronised recording of gaze and behavioural events from that application, and a pipeline for computing oculomotor and pupillary metrics according to current standards [22, 19]. No off-the-shelf tool could be found that would perform all of these tasks while at the same time fitting the budget of a researcher without laboratory support.

UX-analytics tools (Hotjar, Maze) integrate well into the design process and support remote testing, but do not allow the collection of physiological indicators for measuring cognitive load. Webcamera–based web eye-tracking [25] estimates the point of gaze too coarsely and does not allow pupil metrics to be measured, as is also the case with affordable gaming eye-trackers such as the Tobii 4c. Professional systems (Tobii Pro, EyeLink, SMI) provide high signal quality, but their high cost makes them inaccessible to a lone researcher or a team without a budget; moreover, in this class the term "platform" usually refers only to the recording software, while the stimulus application, event synchronisation, and analytical pipeline are assembled by the researcher. The architectural guidelines for such an assembly are described in the literature [15], but are reimplemented from scratch in each new project.

At the time the study was organised, the relatively affordable Gazepoint GP3 eye-tracker, used in HCI research [30], was available. However, the software supplied with it did not fit the task. The free Gazepoint Control application handles calibration and raw recording, but provides neither a stimulus framework, nor synchronised capture of behavioural events from a web page, nor signal processing. The commercial add-on Gazepoint Analysis is designed for static and quasi-interactive stimuli, and a licence for it was not budgeted. The open-source Python wrapper PyOpenGaze gives access to the data stream, but does not allow the orchestration of an experimental session, synchronisation of recording, or analysis.

A researcher starting to work with the GP3, and with other eye-trackers of a similar class, on interactive web stimuli assembles a platform for their project anew: writes the stimulus application, arranges the synchronisation of gaze and behavioural events, and implements signal preprocessing and metric computation. This work is repeated in every study and does not turn into a reusable methodological standard.

1.3 Aim and main result of the work

The aim of the present work is to develop a **tool for measuring cognitive load by eye-tracking** on the Gazepoint GP3 as a full-cycle platform: from stimulus presentation to group-level statistical tests. The platform combines three components: a program for running the experiment, which launches the stimulus web interface and synchronously records gaze data, cursor coordinates, screen frames, and behavioural events (clicks and scrolls), together with responses to the NASA-TLX questionnaire; a data-processing pipeline that cleans the signal according to current pupillometry standards [22, 19] and computes oculomotor, pupillary, and behavioural metrics together with group-level statistical tests for a typical HCI experimental design; and documentation that allows another researcher to reuse the platform or its individual modules.

The work is in the genre of a systems contribution with a demonstration study [41]: the main contribution is the tool, while the pilot study shows how it works across the full cycle and where it runs into its limitations. For the demonstration, a real design question was chosen — whether progressive disclosure elicits lower cognitive load across three types of task in a cloud-infrastructure management console. The literature offers no confident empirical answer to this question, which gives the demonstration independent substantive interest.

2 State of the art

To develop a tool for running experiments that measure cognitive load, it is necessary to review the theoretical foundations, the methodological approaches to measuring cognitive load, and the specifics of processing oculometric data, to survey the existing tools, and to formulate the requirements for the platform.

2.1 Cognitive load and its measurement in HCI

Cognitive load theory (CLT), which describes how a person's resource-limited working memory processes information while performing tasks, was chosen as the theoretical framework of the work [35]. CLT assumes that working memory is able to hold a limited number of elements simultaneously, while long-term memory is practically unlimited and able to group multiple elements into a single "chunk" [37]. Within this theory, three types of cognitive load are distinguished: intrinsic, determined by the complexity of the task itself; extraneous, related to how the information is presented; and germane, reflecting the resources directed at building cognitive schemas. For the design of interactive systems, this yields a pragmatic guideline: extraneous load should be minimised, germane load supported, and their sum should not exceed the capacity of working memory. In a recent review that summarises cognitive load theory more than thirty years after its emergence, Sweller emphasises that the most significant source of individual differences in processing the same material is the volume of already-acquired knowledge: for experienced users, part of the load is handled through learned schemas, whereas for a novice, resources go into building new cognitive schemas [36].

In the field of Human-Computer Interaction, cognitive load theory is used as a framework for analysing the user experience. In the model proposed by the authors of a review of the application of CLT in HCI [14], extraneous load is divided into two parts: that caused by the structure of the material and that caused by the use of the program. This view makes it possible to assess specific interface-design decisions as factors that affect the user's extraneous cognitive load, to which the general approaches of CLT can be applied.

The systematic review by Kosch and colleagues covers the main approaches to measuring cognitive load in the HCI literature and proposes a classification of methods into subjective (questionnaires), behavioural (performance, completion time), and physiological (eye-tracking, EEG, GSR); the authors note that the choice of metrics is often arbitrary and limits reproducibility, and recommend combining heterogeneous data sources within a single session [17]. In a systematic review devoted to methods of measuring cognitive load in the context of usability evaluation, Darejeh and colleagues analysed 76 empirical works on different types of interfaces and formulated recommendations on which instrument is suitable for which type of system under study; for interactive software interfaces the authors recommend combining a subjective scale and behavioural measures, supplementing them where possible with a physiological channel [6]. Subjective cognitive-load questionnaires in themselves demonstrate acceptable internal consistency, but differ in validity and often do not distinguish the types of load in a pure form [18], which points to the correct methodological approach: a combination of self-report, behavioural, and physiological data.

2.2 Eye-tracking as a method of measuring cognitive load

Eye-tracking is used in HCI as an indirect, objective way of studying the processes of visual attention and cognitive effort during task performance. In their classic review of the application of eye-tracking in the usability field, Poole and Ball systematise oculomotor measures — fixations (gaze stops $\geq$ 100 ms), saccades, scanpath parameters — and show how each of them can be related to specific aspects of usability: the number and density of fixations reflect the difficulty of visual search, fixation duration reflects the depth of

processing of a particular element, and scanpath length reflects the efficiency of visual navigation [29]. Similarly, the review by García and Cano provides a consolidated list of oculomotor metrics linked to components of the user experience and describes scenarios for their use [8]. At the same time, as Groen and Noyes showed, the relationship between the distribution of gaze and final performance is not always direct: subtle design changes do not necessarily lead to the expected change in behavioural metrics; it is necessary to combine eye-tracking with other data channels and not to interpret gaze as an unambiguous indicator of success [11].

An important role in assessing cognitive load is played by pupillometry — recording pupil diameter as an indicator of the sympathetic activation associated with mental effort (the task-evoked pupil response, TEPR). In a review spanning two decades of research, van der Wel and van Steenbergen showed that an increase in pupil diameter is reliably reproduced as demands on cognitive control increase — switching, inhibition, updating the contents of working memory — and is interpreted as a marker of the effort exerted rather than of the objective difficulty of the task [40]. In parallel, Stolte and colleagues, in visual-search and visual-working-memory tasks, showed that pupil diameter grows as the load on working memory increases, and, under low load, reflects the difficulty of the search itself — it responds precisely to the combination of search and information-holding processes that is typical of comparison tasks [34]. In their study, Perkhofer and Lehner applied the computation of a combined metric that unites fixation, saccade, and blink measures through structural equation modelling, which predicts the level of load more accurately than any single measure [27]. In the study by Zagermann and colleagues, an increase in cognitive demands in visual-search tasks was accompanied by a simultaneous change in several oculomotor metrics — pupil diameter, blink rate, and the density of fixations and saccades per unit of time — which confirms the practical value of a multi-parameter approach [42].

Working with pupillometry requires careful control of confounders — stimulus brightness, the ambient illumination of the room, and event alignment. The practical guide by Mathôt and Vilotijević describes recommendations for experiment design and the preprocessing pipeline: controlling the brightness of the stimulus background, measuring baseline indicators before the stimulus, normalising measurements relative to the individual baseline diameter, removing blink artefacts, interpolating gaps, and filtering [22]. The complementary guide by Kret and Sjak-Shie contains detailed recommendations for binocular signal assembly, filtering by physiological thresholds, and the computation of confidence intervals [19]. These recommendations determined the requirements for the platform: a dark colour theme matching the laboratory room, a fixed two-second interval between tasks with a fixation cross for measuring baseline indicators, and computation of the normalised ratio of the diameter during the task to the baseline period as the dependent measure of effort.

The conversion of the raw eye-tracker signal into fixations and saccades is performed by oculomotor-segmentation algorithms. Salvucci and Goldberg introduced a classification of fixation-identification methods; the most widespread are the velocity-threshold I-VT and the dispersion-threshold I-DT [31]. In the pipeline of the present platform, fixation segmentation is performed by the built-in algorithm of the Gazepoint API — a dispersion-based one; the number and duration of fixations are computed per task; areas of interest are defined by DOM elements; and pupil diameter is normalised to the individual baseline diameter.

2.3 Existing research platforms for eye-tracking

To develop a full-fledged tool, the following task must be solved: to assemble an experimental session in which stimulation, recording, and post-processing meet the methodological standards. The existing solutions can be divided into three categories, each of which solves its own part of the task, but none of which allows it to be solved in full: these are professional research systems, affordable eye-trackers with their bundled software, and open-source libraries.

2.3.1 Professional systems. At the top of the market stand the integrated systems Tobii Pro Lab, SR Research EyeLink Experiment Builder, and SMI iView (SMI production was discontinued in 2017 after the acquisition by Apple, but licensed systems continue to be used in laboratories). All three are built as a closed ecosystem: a proprietary eye-tracker with a recording frequency of 250 to 2000 Hz and a gaze-position accuracy of $\leq 0.5^{\circ}$, a stimulus-design environment with a GUI editor of paradigms, and built-in analysis of AOIs, heatmaps, transition matrices, and scanpaths.

A significant part of the classic literature on the psychophysiological experiment was written for such systems [15], and their strength lies in signal reliability and replication. For the present project there are two weaknesses. The first is cost: a full Tobii Pro Spectrum kit with a Pro Lab licence is estimated in the tens of thousands of dollars, and the price of the EyeLink 1000 Plus is comparable, which is impossible for a lone researcher or a small group, especially given the constraints on international collaboration and on access to licensed foreign software. The second is an architectural limitation of the environment: the Tobii Experiment Builder and the analogous SR Research module are designed for classic psychophysiological studies — presenting pictures, reading text, visual search with fixed stimuli. Embedding an interactive web application in which the participant fully interacts with the page, and capturing these events in the environment with the ability to synchronise them with the eye-tracker, requires rewriting part of the stimulus layer around the tracker's own API.

2.3.2 Affordable eye-trackers and their bundled software. The second category comprises more affordable eye-trackers: the Gazepoint GP3 (about $950, 60 or 150 Hz, open API), the Tobii Eye Tracker 5 (about $230, consumer-grade, 33 Hz), and the Eye Tribe (sales discontinued in 2016, previously $99, 30 or 60 Hz). These devices are designed as signal capture: USB connection, software or GUI calibration, a stream of gaze and pupil data via an API. The situation with the official software is uniform: the manufacturer supplies a calibration and basic-recording application.

In the case of Gazepoint, the delivery consists of two programs. Gazepoint Control — a free application — handles calibration (5 or 9 points) and raw recording of a TSV log, and opens the OpenGaze API over TCP (port 4242) for external control. At the level of signal capture, the coverage is complete, but Control provides neither a stimulus framework, nor synchronised capture of behavioural events from the page, nor signal processing. Gazepoint Analysis — a commercial add-on costing about $3000 — has as its stated use

case the analysis of recordings: AOIs, heatmaps, and comparison of fragments. Its target stimuli are static and quasi-interactive: images, videos, individual screenshots. It is impossible to assemble in Analysis an experiment made of live interactive web stimuli with direct recording of events from them and to extend the analytical pipeline with one's own metrics: the program provides neither an API for embedding an external stimulus nor extensible signal processing. For the design of the present study these limitations are blocking, and a budget constraint was also in effect — the licence was not budgeted for the work.

With the Tobii Eye Tracker 5 and similar consumer models the situation is close: the bundled software is oriented towards the gaming and accessibility use case, while the research mode is locked behind a Tobii Pro licence. The Eye Tribe, before sales were discontinued, came with a minimal demo program and without an analytical layer.

2.3.3 Open-source libraries and web eye-tracking. The third category comprises open-source libraries. PsychoPy [26] provides a framework for stimulus presentation and integration with several tracker models via iohub; it is oriented primarily towards psychophysiological paradigms with fixed stimuli. PyGaze [5] — a Python wrapper over several tracker models with an emphasis on attention experiments. PyOpenGaze [4] — a low-level Python client for the Gazepoint OpenGaze API: direct access to the data stream and event markers via the USER column, but without session orchestration, recording synchronisation, or signal processing.

Individually, these libraries implement separate stages of the pipeline and do not add up to a coherent platform: the least covered are processing and metric computation according to pupillometry standards, and the coupling of a web stimulus with native recording — exactly what is most often required in HCI studies of interactive interfaces. The architectural guidelines for such an assembly are described in the literature [15], but are reimplemented from scratch in each new project.

Web-camera–based web eye-tracking deserves a separate mention: WebGazer.js [25] is popular in practical UX analytics and remote learning research thanks to its zero equipment cost, but it is substantially inferior to hardware trackers in gaze-position accuracy, and web-camera pupillometry remains experimental and unsuitable for measuring cognitive effort. UX-analytics tools (Hotjar, Maze) integrate well into the design process and support remote testing, but do not collect physiological indicators and are unsuitable for the task of measuring cognitive load.

2.3.4 The identified gap and the requirements for the platform. A researcher who has chosen the GP3 (or an eye-tracker of a comparable class) to evaluate the design of interactive interfaces ends up at a point where there is quality equipment at a reasonable price, there are methodological standards of pupillometry [22, 19], and there are scattered libraries, but there is no coherent platform that would bring this together into a single workflow. From this gap the functional requirements for the tool to be developed are derived:

- a stimulus application supporting arbitrary interactive web patterns, not only static images and videos;
- synchronised recording of the gaze stream, screen frames, cursor coordinates, and behavioural events from the stimulus on a single timeline;
- collection of responses to a validated questionnaire;
- a pipeline for preprocessing the pupil signal according to current standards [22, 19]: binocular assembly, a speed filter, interpolation of short gaps, baseline correction, QC metrics;
- a library that computes oculomotor, pupillary, and behavioural metrics plus group-level statistical tests for a typical HCI design;
- operation on affordable hardware (Gazepoint GP3) without commercial licences for analytical software;
- documentation and a reusable structure that allow the platform to be applied beyond the demonstration case.

The developed platform is aimed at filling this gap; its implementation is described in Section 4.2.

2.4 Information density, progressive disclosure, and interface complexity: the subject area of the demonstration study

To validate the developed platform, a topical design problem was chosen. Cloud-infrastructure management interfaces belong to the class of high-information-density interfaces: on a single screen, summaries of numerous resources, their parameters, statuses, metrics, network configurations, and alerts are presented simultaneously. Empirical work consistently shows that an increase in the visual complexity of an interface raises the user's cognitive load and reduces the efficiency of visual search. In the eye-tracking study by Wang and colleagues with 42 participants, it was demonstrated that website complexity interacts with task complexity: for a simple task on a high-complexity site the number of fixations and completion time increase, while for a complex task the greatest load falls on a site of medium complexity [39]. Tian and colleagues, in a recent eye-tracking study of dashboard design, showed that the orderliness of the relative arrangement of charts and the position of the "core" element substantially affect the visual-search strategy [38]. The behavioural model of visual search in graphical interfaces described by Putkonen and colleagues on a corpus of 900 real GUIs describes a stable "Guess-Scan-Confirm" pattern [30].

One of the established ways in interface design of reducing extraneous cognitive load in dense interfaces is **progressive disclosure** — a technique in which the user is initially shown only the information most essential for the current task, while secondary details are hidden and can be revealed on demand. The concept goes back to the work of Carroll and Rosson on the training-wheels approach [3] and was formalised by Nielsen in the usability heuristics of the mid-1990s [24]. In the work of Spillers and in later publications by Nielsen, progressive disclosure is described as a way of moving rarely-used elements to secondary screens, which reduces visual noise and structures the transition from "general to specific" [33, 23].

Nevertheless, the empirical base directly testing the effectiveness of progressive disclosure in technical interfaces with high information density remains limited. Most publications are practical design guidelines or qualitative studies. Liu and Xia, comparing a region drop-down list and tabs in a cloud interface, found that

the drop-down variant provided better user performance and was rated as more convenient, although the study was conducted on a narrow region-selection scenario and did not assess cognitive load directly [20]. The existing data make it possible to formulate the expectation that the effect of progressive disclosure depends on the nature of the task: when searching for a specific value, hidden sections may create an additional step; when comparing resources, a reduction in visual density should help, but hidden sections may force the user to hold the contents of several collapsed blocks in working memory; and in diagnosis — a task with high interactivity of elements — the benefit should be greatest. This is what motivated the 2×3 factorial design (condition $\times$ task type) in the demonstration study.

As a subjective measure of cognitive load, NASA-TLX is most frequently used [13] — a multidimensional scale with six subscales. Recent work critically discusses the applicability of NASA-TLX in HCI: a systematic review showed that in many works researchers interpret it without due consideration of its theoretical foundations and psychometric properties [28, 17]. Nevertheless, as a standard, validated instrument comparable with the existing literature, NASA-TLX is used in the platform as the subjective channel, applied after each block of tasks.

3 Methodology of the pilot study

3.1 Study design

The main method is a controlled laboratory experiment with a within-subject design: each participant performed tasks in two variants of the interface — with progressive disclosure and without it. The *independent variable* is the degree of progressive disclosure: the "flat list" condition (*flat*) and the "expandable sections" condition (*sections*). The *dependent variables* are eye-tracking indicators, behavioural metrics, and subjective assessments of cognitive load obtained by completing the NASA-TLX questionnaire (Section 3.7).

Two-factor counterbalancing was applied. The first factor is the order of conditions: group 1 performed the tasks *flat list* → *expandable sections*, and group 2 in the reverse order. The second factor is the order of the datasets and their associated tasks: e-commerce or fintech in the first block. The combination yields four counterbalancing groups.

3.2 Research questions and hypotheses

The **main question** of the demonstration study: does an interface with progressive disclosure lead to lower cognitive load when performing infrastructure tasks?

- **H1.** The normalised pupil diameter will be smaller in the *expandable sections* condition than in the *flat list* condition.
- **H2.** The number of fixations per task will be smaller in the *expandable sections* condition: more focused interaction.
- **H3.** The total NASA-TLX score after the *expandable sections* block will be lower.
- **H4.** The median task-completion time will be shorter in the *expandable sections* condition.

In addition to the quantitative hypotheses, the study includes a qualitative branch with its own sub-questions (RQ-qual-1…4), aimed at interpreting the subjective experience of working with the two interface variants — see Section 3.8.

3.3 Participants

Ten technical specialists ($N = 10$) with experience working with infrastructure took part in the study: system administrators, DevOps engineers, backend developers, and ML engineers. Inclusion criteria: experience with deploying and configuring cloud resources for ≥ 6 months; normal or corrected-to-normal vision. Demographic data and level of experience were recorded in an online Google Forms questionnaire. All participants signed an informed-consent form.

3.4 Stimuli and materials

3.4.1 Interface prototypes. Two functional prototypes of a cloud console were developed, both in a unified style (dark theme, console typography) without the branding of real providers. The colour tokens of the dark theme were taken from the open-source Consta design system [9] — this made it possible not to develop a palette from scratch and to keep the neutral styling unattached to any particular cloud provider; the layout of the resource cards, the icons, and the prototype components were developed independently. Implementation: a web application embedded in a native pywebview window (a detailed description of the architecture is in Subsection 4.2.1).

The *flat list* condition: all information about a resource is displayed immediately as expanded sections with the values of the parameters, and is accessible by scrolling.

The *expandable sections* condition: information is grouped into collapsible sections. By default the "Main" section is open and the rest are closed; the user expands the ones they need while working.

3.4.2 Datasets. To mitigate the familiarity effect when repeating tasks, two content fillings of the prototype were developed: the Alpha and Beta datasets. The Alpha dataset comprises the resources of an e-commerce platform; the Beta dataset comprises the resources of a fintech platform. Each resource is described by attributes (id, type, status, region, and others specific to the resource type), grouped into sections by meaning. The datasets are balanced in the number of resources; each is used in one block of experimental tasks. The logic behind the construction of the datasets and tasks is described in the results section 4.1.4.

3.4.3 Tasks. Each block includes 6 tasks of three types (2 tasks of each).

- **Lookup:** find a specific parameter — targeted visual search with minimal integration of information.
- **Comparison:** compare several resources by their attributes — holding several values in working memory.
- **Diagnosis:** identify the resource that is the source of a problem from its symptoms — integration of information and inference.

The order of tasks is chosen from 4 permitted permutations in which tasks of the same type do not follow one another. Before each block there is one training task for familiarisation with the interface.

3.5 Hardware and software

Eye-tracker: Gazepoint GP3 HD, 150 Hz, requiring no head fixation. The same device was used in [30], which allows the data obtained here to be directly compared with that publication. Mouse-cursor coordinates were recorded in parallel. Screen recording was performed at 30 frames per second. The experiment program was run on an ASUS Vivobook 15 laptop with an Intel i5 processor and an NVIDIA GeForce GTX 1650 graphics card. When attempting to run it on a less powerful configuration, there is a risk of losing synchronisation between the eye-tracker and the screen recording, of delays, errors, and overheating of the equipment.

Software: Python / Flask / pywebview. TCP connection to the eye-tracker: the PyOpenGaze library (port 4242). The application was built into `Experiment.exe` via PyInstaller. The architecture and source code are the subject of the results section 4.2.1; the repository is available on GitHub:

https://github.com/eliz-z-za/eyetracking_cogload_pd/tree/main.

3.6 Procedure

Sessions were conducted individually, with a duration of 25–35 minutes. Data collection included signing the informed-consent form, completing the Google Forms questionnaire, the experimental session itself, and a semi-structured interview. The study procedure is described in detail in Section 4.3.1.

3.7 Dependent variables and measures

Eye-tracking metrics. For each task the following are recorded:

- *Fixation count* — gaze stops ≥ 100 ms.
- *Normalised pupil diameter* — subtractive baseline correction: $\bar{D}_{\text{task}} - \bar{D}_{\text{baseline}}$, where the baseline is the median diameter in the preceding 2-second ITI; an increase reflects an increase in effort [40, 22]. In parallel, z-normalised and percentage measures are computed as sensitivity checks.
- *Mean fixation duration.*

Behavioural metrics: task-completion time (ms, from `PD_TASK_DONE`); number of clicks; number of expanded sections (*sections* only, from `PD_SECTION_TOGGLE`).

Subjective measures: the Russian-language version of NASA-TLX in the adaptation of [43] — five scales (mental demand, temporal demand, performance, effort, frustration; range 0–20, step 1). The overall (unweighted) score and the profile across scales. Completed after each block.

Qualitative data: transcripts of semi-structured interviews — thematic analysis to interpret the quantitative differences and to obtain additional information about the mechanisms that affect the experience of working in the interface when managing cloud infrastructure.

3.8 Mixed methods: integrating the quantitative and qualitative branches

The qualitative branch operates on an equal footing with the quantitative one and is collected in the same session, immediately after the second block of tasks. It is aimed at four research sub-questions:

RQ-qual-1. What differences do users perceive between the flat presentation and collapsible sections, and what preferences do they articulate?

RQ-qual-2. What information-handling strategies do users build in each of the conditions, and how are these strategies related to task type?

RQ-qual-3. What factors beyond the way information is presented do users name as significant for their experience?

RQ-qual-4. What functional or visual changes do users consider necessary for the prototype to be applicable in real work?

The transcripts were processed following the thematic-analysis procedure of Braun & Clarke [2, 1]; coding was carried out by a single coder (the author of the work), and this limitation is discussed in Section 6. The combination of channels is justified by the fact that each of them is individually limited [17, 6]: eye-tracking gives a continuous signal but does not explain strategy; questionnaires are subject to ceiling effects; and behavioural metrics do not distinguish whether the user slowed down because of the interface or because of the task.

The consolidation of the quantitative and qualitative conclusions is carried out in the form of a joint display [7] in Subsection 4.3.4; for the analysis of divergences and convergences, the typology of analytical functions of Greene, Caracelli & Graham [10] is used — triangulation, complementarity, initiation, expansion.

3.9 Data analysis

Quantitative analysis: the paired non-parametric Wilcoxon test for each key indicator (flat vs. sections), aggregation at the participant level with a subsequent Benjamini-Hochberg correction for multiple comparisons. Effect size — the rank-biserial r and Cohen's d from the paired differences. Additionally, permutation tests (10000 sign permutations) were carried out as an exact check for small N, and bootstrap 95% CIs for the effect sizes. Analysis by task type — with the same tests within each type (lookup/comparison/diagnosis). Sensitivity checks on different subsamples (`df_all`/`df_good`/`df_balanced` — see Section 4.3.2). The full analysis pipeline is described in the results section 4.2.2.

4 Results

4.1 The management-console prototype

The prototype is a web application that simulates a cloud-infrastructure management console; it is precisely this that the participants work with, and the comparison of its two variants constitutes the content of the demonstration experiment (§4.3). The design of the prototype rests on a comparative analysis of existing consoles, set out below.

4.1.1 Analysis of existing consoles. Before designing the prototype, a comparative analysis of the interfaces of real cloud-infrastructure management consoles was carried out. The aim of the analysis was twofold: to obtain a reference base for the design of the stimulus (the macro-structure of the resource page, the typical set of attributes, the way navigation is organised) and to understand which patterns of information disclosure are established in the industry and which vary.

Two classes of consoles were analysed: international hyperscalers (AWS Console, Google Cloud Console, Azure Portal) and domestic providers (Yandex Cloud, Selectel, VK Cloud). For each platform the following were recorded: navigation between sections with the main resources, the way information is presented on the resource page, and the encoding of status. A summary comparison is given in Table 1.

The comparison yielded three stable observations. Modern consoles almost always combine several progressive-disclosure patterns at once: the separation of "resource list / resource details", collapsible sections or tabs within the detail screen, and pop-up windows for auxiliary actions. A significant part of the interface is devoted to status signals (health indicators, launch statuses, alerts), but the providers have not developed a single visual vocabulary: status is encoded now by colour, now by icon, now by text. The terminology diverges between providers; the general structure of attributes (general, network, security, monitoring, metadata), however, is stably repeated across the main resource types and is recognisable to an experienced user.

These observations determined the decisions on the prototype: a dark theme without specific branding, a section macro-structure following the stable categories, and resource status duplicated by colour and text simultaneously. Exactly one pattern is varied — collapsible sections versus a flat list — in order to isolate the subject of the demonstration study and not to mix it with differences in navigation, in the format of the tabular list, or in status semantics.

4.1.2 Design and implementation of the prototype. In accordance with the conclusions of the analysis (§4.1.1), the prototype is styled in a dark theme with console typography. The macro-structure of the resource page reproduces the stable categories: *Main parameters*, *Network*, *Security*, *Monitoring*, *Metadata*. The colour tokens of the dark theme were taken from the open-source Consta design system [9]; the layout of the resource cards, the icons, and the components were developed independently.

The prototype is implemented as a web page rendered in the pywebview Chromium engine. HTML/CSS/JS gives freedom to design arbitrary UI patterns, while pywebview opens a full-screen "native" window without browser bars, which simultaneously removes distracting elements and facilitates screen recording.

4.1.3 The two variants of the prototype. The prototype exists in two functionally equivalent variants, between which the participant is compared in the within-subject experimental design:

Flat list (*flat*). All sections of the resource are displayed immediately in expanded form; the information is accessible by scrolling without additional actions (Fig. 1 a).

Expandable sections (*sections*). Sections are closed by default except for the section with the main information; the user expands the ones they need with a click (Fig. 1 b).

Exactly one pattern is varied — collapsible sections versus a flat list — in order to isolate the subject of the experiment and not to mix it with differences in navigation, in the format of the tabular list, or in status semantics.

4.1.4 Datasets and tasks. The datasets are a key substantive artefact of the prototype. Their design was subordinated to three requirements: *balance* between the datasets in the number and complexity of resources; *domain recognisability* for experienced users (e-commerce vs. fintech — two widespread subject areas in which typical DevOps/SRE engineers work); and *the semantic consistency of resources and attributes* (for example, the ID of a database in a firewall rule matches a real database in the dataset, and the status of a load balancer is consistent with the status of the backend servers), so that diagnosis tasks are solved through the matching of information rather than guessing.

Each resource is described by a JSON structure with the following keys: `id`, `type`, `status`, `key_info` (a short summary for the resource list), `region`, and `sections` — a dictionary of parameter groupings. Typical sections: *Main parameters*, *Network*, *Security*, *Monitoring*, *Metadata*. Some sections contain embedded tables (for example, the backend pools of a load balancer, firewall rules, the volumes of a virtual machine) — these are marked with the prefix `__table_NAME__`.

The resource types in both datasets are the same (virtual machines, databases, firewall rules, load balancers, networks), but the specific resources, names, and values differ between the alpha and beta datasets. This ensures that knowledge of specific values cannot be transferred between blocks while the search structure remains the same.

The tasks (6 in each block, 2 for each of the three types) were designed iteratively: the first version was piloted on 2 participants, after which a number of wordings were refined to reduce ambiguity. Each task contains: a number, a task type, the text of the question, the target resources (on which the task is solved), the expected answer, and the expected answer section (to test the hypothesis about which section the task activates). Only the task text is shown in the interface.

The order of tasks within a block is one of four permutations in which tasks of the same type do not follow one after another (this prevents the accumulation of fatigue within one type and a learning effect within a type). The specific permutation is set by the `PD_TASK_ORDER_INDEX` parameter (0–3) in `flags.txt`.

Training tasks are a separate entity: they are formulated as close as possible to lookup tasks, are performed on one of the `staging`-resources, and are not written to the main data stream, but pass through the same interface as a block.

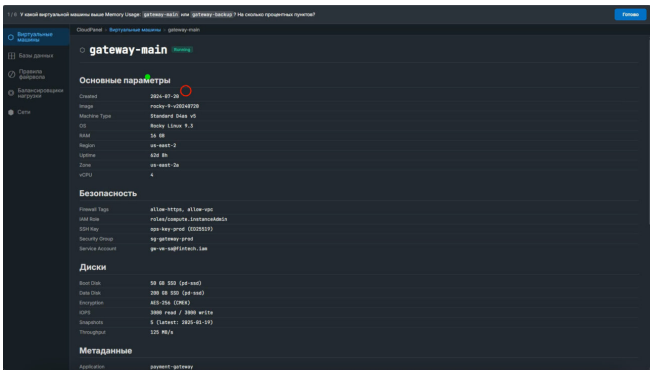
4.2 The platform for running the experiment

The platform is a full-cycle tool on which the experiment is conducted: the session orchestrator, synchronised recording of the gaze stream, screen frames, and behavioural events, a library for preprocessing the pupil signal and computing metrics. The prototype (§4.1) is embedded in the platform as one of the stimuli and uses its front-end infrastructure; everything else — recording, synchronisation, processing, analysis — is the platform proper.

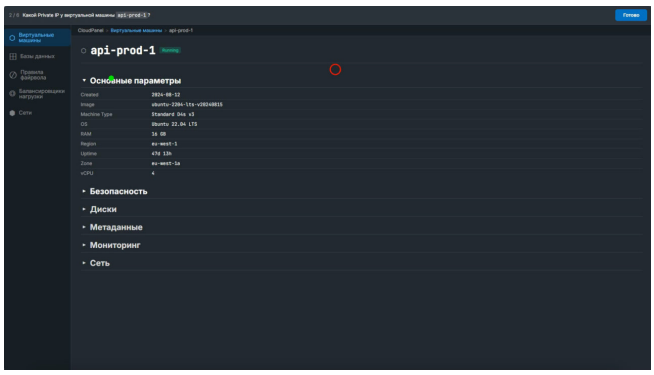
4.2.1 Architecture and data flow. The platform is designed as a set of components that exchange data via TCP sockets and a shared file system (a conceptual diagram of the architecture is provided in the platform's supplementary documentation). The platform consists of the following components:

Table 1: Comparison of cloud-infrastructure management consoles by key interface patterns.

Console	Navigation and resource list	Presentation of information on the resource page	Status encoding
AWS Console	Services in the top bar and via search; within a service — a tabular list with filters and type icons	An expanded "Summary" block with the main information, tabs (*Details*, *Monitoring*, *Tags*...) + collapsible sections within tabs	Coloured dot + text label (*Running*, *Stopped*)
Google Cloud Console	Services in a collapsible side menu; within — a tabular list with configurable columns	Tabs (*Overview*, *Logs*, *Monitoring*...) + key-value sections in Overview	Coloured icon + text label
Azure Portal	Access via search, favorites, and resource groups; resources open as standalone "blades"	An expanded "Essentials" block with the main information, tabs (*Properties*, *Monitoring*, *Capabilities*...) + collapsible sections within tabs	Coloured icon + text state
Yandex Cloud	Services in a collapsible side menu; *cloud* / *project* navigation in the header	Main sections with information without hiding, partially hideable sections (*Connection instructions*)	Coloured dot + text
Selectel	Partly via a product catalogue (a menu that expands from the header), partly via the home page	Tabs (*Network*, *Disks*, *Services*)	Coloured text
VK Cloud	Side menu with services	Tabs (*General information*, *Monitoring*, *Disks*...): "General information" is always open	Coloured icon in the general table, text on the resource page



(a) Flat list (*flat*).



(b) Expandable sections (*sections*).

Figure 1: The two variants of the prototype, the same resource page: (a) a flat list with all sections expanded at once; (b) expandable sections, collapsed by default except for the section with the main information.

- **Gazepoint Control** — an external application from the eye-tracker manufacturer: connects to the GP3, manages calibration, opens the OpenGaze API over TCP (port 4242).
- **PyOpenGaze** — a Python client for the OpenGaze API: a TCP connection to Gazepoint Control, enabling of the data streams (gaze, cursor, pupil, fixations), recording of a TSV log with the columns `TIME`, `FPOGX`, `FPOGY`, `FPOGD`, `LPD`, `RPD` and others, and sending of event markers to the `USER` column.
- **Flask server** — serves the HTML pages of the stimuli (instructions, forms, the management console, the NASA-TLX questionnaire); receives events from JavaScript via POST `/event`; keeps a thread-safe event queue for reading by the orchestrator.
- **Embedded browser (pywebview)** — creates a full-screen window with a Chromium engine; opens the pages of the Flask server; the orchestrator navigates via `window.load_url()`.
- **Stimulus front-end**: `experiment.js` — the base module (`sendEvent()`, the idle detector); `pd_console.js` — the logic of the management-console prototype (sidebar, resource rendering, attribute sections, AOI snapshots).
- **Experiment orchestrator (`experiment.py`)** — launches the pywebview window and Flask in the background; connects to the GP3 or starts mock mode; manages the sequence of pages and waits for events from the queue; starts/stops the screen and GP3 recording; logs events to CSV.
- **Screen recording (`screen_recorder.py`)** — captures frames via `mss`, encodes via `ffmpeg`; stable FPS on a timer; correct video metadata for synchronisation with the GP3.
- **Overlay script (`overlay.py`)** — reads the screen video and the GP3 TSV log; synchronises the timelines; draws the gaze trajectory (red circle) and the cursor (green dot) on top of the video. During development it was used to verify that gaze and cursor are indeed written in sync with the

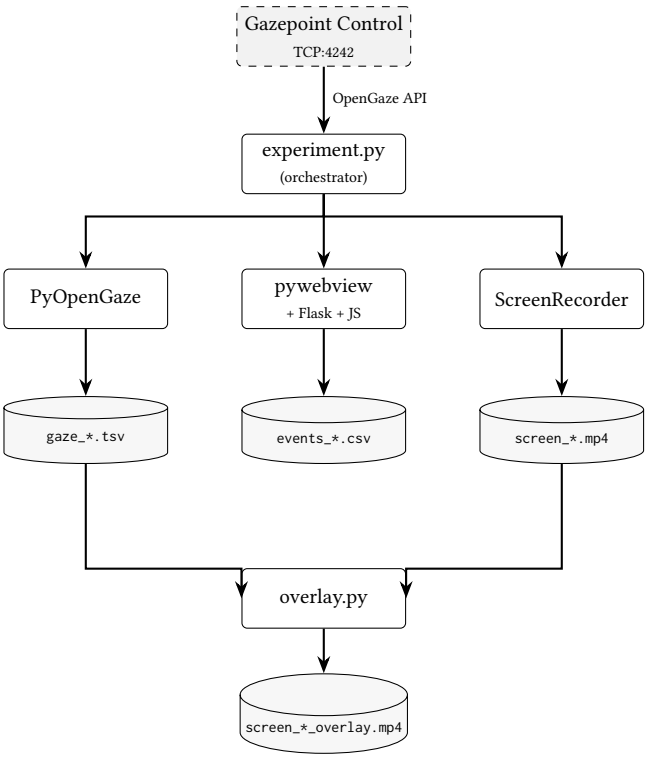


Figure 2: The architecture of the platform and the data flow. Rectangles are software components (a dashed frame denotes external software), cylinders are data files, and arrows are the direction of flow. `experiment.py` controls three subsystems: `PyOpenGaze` reads the gaze stream via the OpenGaze API and writes TSV, `pywebview` with the embedded Flask server shows the stimulus and logs behavioural events to CSV, and `ScreenRecorder` writes the screen video. In post-processing, `overlay.py` combines the gaze TSV log and the screen video into a clip with the overlaid trajectory.

screen video; it is also applicable as supporting material in the qualitative analysis of sessions.

The links between the components and the data flow are shown in Fig. 2.

Event markers are written through two channels at once: every significant event goes into the GP3 eye-tracker's TSV log via the USER column (precise alignment to the gaze-sample index) and into the orchestrator's CSV log (an absolute system timestamp and a human-readable session journal). The duplication is redundant in volume, but critical for the post-hoc reconstruction of timing in the event of recording failures.

Counterbalancing is arranged along two axes, as was described in the methodology: the flat/sections order and the alpha/beta order run as independent factors, yield four experimental groups, and simultaneously control the effects of learning and of the material. Before each block the participant completes one training task on a resource outside the experimental pool (`staging-api-1` for alpha, `staging-gateway-1` for beta) — this makes it possible to get accustomed to the specific disclosure pattern without contaminating the data.

Each task is preceded by a two-second fixation cross on a neutral background of the same brightness as the stimulus. This ITI works in two roles at once: it serves as a psychophysiological "zero point" for pupillometry (the baseline for the subtractive correction) and signals to the participant readiness for the next task.

The full technical documentation of the platform is placed in the `.md` file in the repository.

4.2.2 The data-processing and metric-computation pipeline. The second part of the platform is the library for processing and analysing the collected data. Its goal is to obtain, from the raw gaze TSV stream and the CSV event log, a standardised set of cognitive-load metrics at the task level, aggregates at the participant level, and group-level statistical comparisons. The architecture of the library is divided into three levels.

Level 1: cleaning the pupil signal (`pupil_preprocessing.py`*).* The module implements a pipeline for preprocessing the pupil signal following [22, 19]. The external API is the `clean_gp3_trace(gaze_df)` function, which returns a `CleanedTrace` object with the cleaned signal and detailed QC statistics. The stages of the pipeline:

(1) **Binocular assembly** (`build_binocular_pupil`) by the *at-least-one-eye-valid* rule: if both eyes are valid — the mean $(LPD + RPD)/2$; if only one — its value; if both are invalid — NaN. This decision yields more valid samples than a strict *both-eyes-valid* rule, and is physiologically justified: during a short one-sided loss of tracking (a one-sided blink, a reflex) binocular averaging is not required.
(2) **Speed filter**: any sample where $|\Delta D[i]| > 0.5$ px/sample is masked. The threshold of 0.5 is calibrated on the histogram of $|\Delta D|$ across the whole sample (corresponding to the p_{90} of the binocular signal) and is applied identically to the baseline and task periods.
(3) **NaN margin**: each NaN is expanded by ±2 samples around it, in order to remove transitional states (a half-open eyelid at the boundary of a blink) that are metrically unreliable.
(4) **Interpolation of short gaps** (`_interp_short_gaps`): cubic interpolation for gaps ≤ 15 samples (≈ 250 ms at 60 Hz), with a fallback to linear if there are not enough anchor points. Long gaps remain NaN.
(5) **SD-outlier pass**: a final single-iteration $\pm 3\sigma$ outlier filter on the already-interpolated signal.
(6) **Quality metrics**: the proportion of valid samples, the proportion of unrecoverable ones, and the number of cubic/linear interpolations are returned.

The module is self-contained (it does not depend on the folder structure or on the GP3) and is provided with a CLI wrapper `inspect_pupil_speed.py` for calibrating the speed threshold on a new dataset. An example of the pipeline's operation on a single participant is shown in Fig. 3: the raw binocular signal is shown in grey, and the cleaned trace after the speed filter, the expansion of the NaN boundaries, cubic interpolation of short gaps, and the $\pm 3\sigma$ outlier pass is shown in blue.

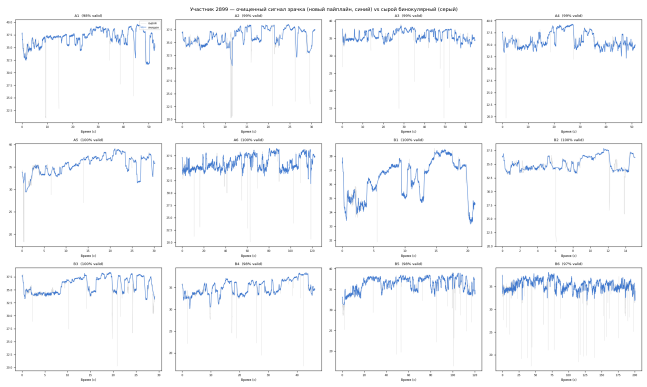


Figure 3: Demonstration of the pupil-signal preprocessing pipeline on a single participant, 12 tasks. The grey line is the raw binocular signal, and the blue line is the cleaned signal after the full `clean_gp3_trace` pipeline.

Level 2: per-participant metrics (`analysis_pd.py`*).* The script takes a participant identifier and a data folder and returns a list of dictionaries (one row = one task). For each task:

- **Behavioural**: `completion_ms`, `clicks`.
- **Fixations**: `fixation_count`, `fixation_duration_mean/median/std/total`, `fixation_rate_per_sec`.
- **Saccades**: `saccade_count`, `saccade_amplitude_mean`, `scanpath_length`.
- **Pupil**: `pupil_mean_avg`, `pupil_subtractive_avg` (the main indicator, following [22]), `pupil_z_avg` (sensitivity), `pupil_pct_change_avg`. QC: `task_pct_unrecoverable`, `task_n_valid`.
- **Static AOIs**: dwell time and fixation count for three zones — the task bar (top), the sidebar (left), and the content (the rest).

The baseline pupil measurement is the 2-second inter-trial interval (ITI); the first 0.5 s are discarded for stabilisation (the post-stimulus reflexive constriction after the disappearance of the previous stimulus). The speed filter is applied identically to the baseline measurement and to the measurement during task performance: asymmetric filtering (only of the baseline) creates a bias between the periods and leaves blink artefacts in the task-performance period.

Each block (b1 and b2) is loaded from a separate TSV file — this is important because the tracking quality may differ between blocks (for example, for participant 1818 in block 1 only 2% of valid samples were found due to a recording failure, while block 2 was normal). The output files: `pd_<id>_metrics.xlsx` with the sheets `per_task`, `per_block`, `summary`.

Level 3: group-level statistical analysis (`pd_analysis.py`, `pd_analysis.ipynb`*).* The script collects the results across all participants and conducts group-level tests. The architecture for working with subsamples:

- `df_all` — all 10 participants; for non-pupillary metrics and as a reference.
- `df_good` — 8 participants with a proportion of invalid samples $\leq 30\%$; used for pupillary metrics and for testing hypothesis H1.
- `df_balanced` — 2 participants from each cell of the 2×2 design (condition order × dataset order); used to test order effects and for correlations between ET metrics and NASA-TLX.

The main statistical test is the paired Wilcoxon on the means over the participant's 6 tasks. The algorithm for each metric:

(1) Aggregate the data frame by participant × condition → mean metrics.
(2) Obtain the paired values `flat_vals`, `sections_vals`.
(3) Run `scipy.stats.wilcoxon` between the flat and sections conditions.
(4) Compute the rank-biserial r and Cohen's d from the paired differences.
(5) Bootstrap 95% CIs for r and d (10000 resamplings).
(6) A permutation test (10000 sign permutations) as an exact check for small N.

In parallel, an exploratory analysis is built through mixed models and GEE with covariates (`dataset`, `block_num`), but at $N = 8$ the LMM is often unstable (singular fit), and the results of these models are interpreted only as exploratory.

The correction for multiple comparisons is Benjamini-Hochberg over the family of metrics in each test. Hypotheses H1–H4 are consolidated into a single table via `build_hypothesis_summary()`.

The visualisation layer includes: violin+spaghetti plots for each key metric (with the participant pairs); a forest plot of the rank-biserial r for the main effect and for the three task types; time plots of pupil diameter for each of the 12 tasks split by condition; and group averaging ± SEM on a normalised time scale.

NASA-TLX is analysed separately: five subscales and an overall weighted/unweighted score; the same paired Wilcoxon + FDR. The subscale weights obtained from the NASA-TLX pairwise comparison are saved for optional weighting of the overall score, but the main analysis is conducted on the unweighted version (following the recommendation of [12]).

Pipeline documentation. The full architectural documentation of the pipeline — the file `ANALYSIS_ARCHITECTURE.md` in the repository — describes each module, its dependencies, and the methodological decisions (the choice of the speed-filter threshold, the choice of subtractive correction as the primary one, the justification of the 50% validity threshold for computing the metrics).

4.3 The pilot study

A pilot study with ten participants was conducted on the assembled platform. It has two main tasks: to verify that the tool reliably runs through a full session and yields a consistent set of data, and to obtain substantive material for testing the hypotheses about the differences between the flat presentation and collapsible sections.

4.3.1 Data collection and procedure. Data collection took place individually in a laboratory room with stable artificial lighting, without windows or direct sunlight; the room's soundproofing excluded background auditory stimuli and provided a calm environment for the interview. The positions of the participant, the monitor, and the eye-tracker were fixed (Fig. 4): the eye–screen distance was about 65 cm, and the eye-tracker was mounted under the lower frame of the monitor. Each participant was asked to rest their head on a support in order to minimise its movements. About 45 minutes were allotted per participant, including the signing of

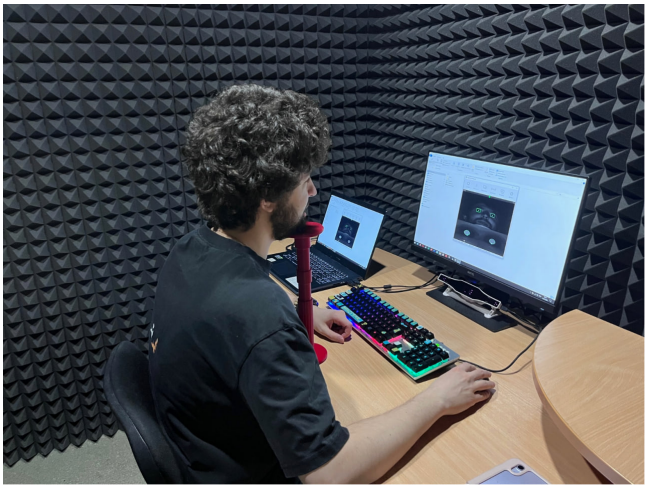


Figure 4: The laboratory setup: the participant at the workstation, the monitor with the stimulus, and the Gazepoint GP3 eye-tracker under the lower frame of the monitor. Stable artificial lighting, a room without windows, soundproofing.

consent, the demographic questionnaire, the calibration, and the post-interview; the net experiment time was 25–35 minutes.

Pre-session preparation. Gazepoint Control and a connection check were launched; the operation of the audio recording for the interview was checked.

Demographic questionnaire and STAI.. These were completed in Google Forms on a separate device, so as not to load the main screen. The demographics included age, sex, general IT experience, experience with cloud infrastructure, a self-assessment of one's level of working with infrastructure, and the presence/type of vision correction. The Spielberger-Hanin state-anxiety scale [32, 16] — 20 items — was completed immediately before the start of the experiment as a measure of state anxiety. The questionnaire was included in the procedure as a confound control: state anxiety directly affects pupil diameter through sympathetic activation and can be confused with the effect of cognitive load in pupillometric metrics. The low and homogeneous level of anxiety across the sample (mean score $M = 18.3$) confirms that the observed differences in pupil diameter between the *flat* and *sections* conditions are not explained by a contribution of anxiety.

Calibration. Software calibration of the GP3 on 5 points with a permissible mean deviation of 40 pixels. On failure, the calibration was repeated.

Experimental session. After completing the questionnaire, the participant moved to the room with the laboratory setup, where the main part was conducted.

(1) Launch of Experiment.exe, entry of the participant id.
(2) Eye-tracker calibration: software calibration of the GP3 on 5 points with a permissible mean deviation of 40 pixels. On failure, the calibration was repeated.
(3) Instruction on performing the tasks: as text in Experiment.exe and verbally by the experimenter.
(4) A training task before the first block, in the interface of the first block.
(5) **Block 1** (condition A, dataset A): each task is preceded by a 2-second fixation cross; the markers `PD_TASK_START/END` are written to the TSV.
(6) NASA-TLX after block 1.
(7) A pause (≈ 5 min), recalibration.
(8) A training task before the second block.
(9) **Block 2** (condition B, dataset B): the same.
(10) NASA-TLX after block 2.
(11) Completion of the experiment and the semi-structured interview.

The order of the screens of the stimulus application (`Experiment.exe`) and the transitions between them — from the entry of the participant identifier and calibration to the training tasks and the final screen — is shown in the diagram in the appendix (Fig. 13).

Data recording. For each participant the following were saved: a TSV file (`gaze_<id>_*.tsv`) with the gaze data, a CSV event log (`events_<id>_*.csv`), two MP4 screen videos (one per block), two TSV gaze fragments by block (extracted by the markers), and two JSON files with the NASA-TLX responses.

Post-interview. Semi-structured, with a duration of 8–14 minutes (median 11). The interview guide was aimed at eliciting the general impression of the interface and the tasks, the perception of the expandable-sections and flat-list conditions, a preference for one of the conditions, and the strategies for performing the tasks. The guide consisted of five questions:

(1) Tell me briefly how it was for you to work with the interface as a whole. What stood out?
(2) Did you notice a difference between the two blocks of tasks? If so, what was it?
(3) In which of the blocks was it easier for you to find the information you needed, and what is that connected with?
(4) How did you usually approach the tasks, how did you perform them?
(5) Is there anything else it would be important to mention that I did not ask about?

Recording was done on a separate voice recorder, and after the interview the recordings were transcribed.

Data quality. For 2 of the 10 participants (identifiers 1818 and 2746), the proportion of invalid pupil samples exceeded the 30% quality threshold; they were excluded from the pupillometric metrics but retained in the sample for the behavioural and qualitative data. For the remaining 8 participants, the proportion of invalid samples was on average 12% (median 9.5%), which corresponds to acceptable quality for the GP3 when working with a free head position. The distribution of the baseline pupil diameters across participants is shown in Fig. 5; participants with a proportion of invalid samples above 30 % are marked as candidates for exclusion from the `df_good` subsample.

4.3.2 The quantitative study. **Sample.** Ten experienced users of infrastructure interfaces took part in the study (median IT experience — 4 years, median infrastructure experience — 3 years; 8 men, 2 women). After QC filtering, the pupillary branch of the analysis

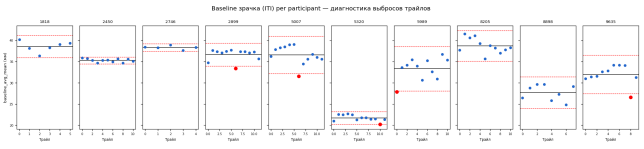


Figure 5: The baseline pupil diameter across participants and the proportion of invalid samples. Marked in red are the participants whose data are excluded from the `df_good` subsample ($N = 8$).

Table 2: H1–H4: Wilcoxon FDR- (Benjamini–Hochberg). M — ; r — rank-biserial .

.		N	M_{flat}	M_{sect}	W	p_{raw}	p_{FDR}
H1	$\bar{D}^{\text{sub}}_{\text{pupil}}$	8	-1.36	-1.41	18.0	1.000	1.000
H2	fixation_count	10	112.77	132.20	13.0	0.160	0.576
H3	NASA-TLX (.)	10	31.90	37.50	19.5	0.432	0.576
H4	median completion ()	10	39.55	45.29	17.0	0.322	0.576

Rank-biserial r: H1 =0.00; H2 =-0.53; H3 =-0.29; H4 =-0.38. H1 N=8 $\leq$ 30% (df_good); H2–H4 — N=10.

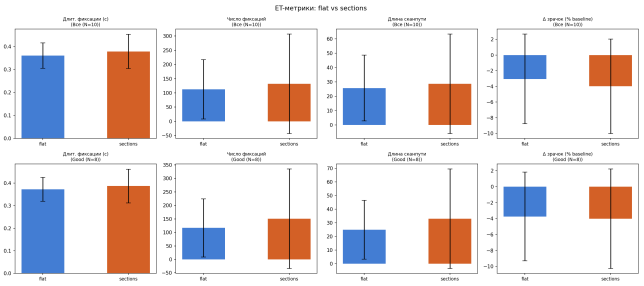


Figure 6: A comparison of the key pilot-study metrics between the *flat* and *sections* conditions: the mean across participants ± SEM. The pupillary measures are computed on the `df_good` subsample (N=8), the rest on the full sample (N=10).

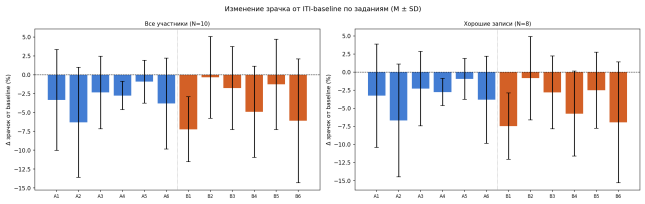


Figure 7: The change in pupil diameter relative to the baseline over the 12 tasks, means across participants ± SEM. On the left is the *flat* condition, on the right *sections*.

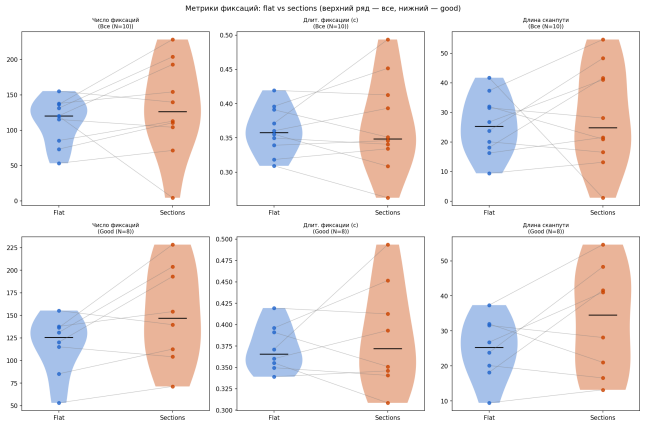


Figure 8: Violin+spaghetti plots of the oculomotor metrics (fixation count, mean fixation duration, scanpath length) for the *flat* and *sections* conditions. Each line is one participant.

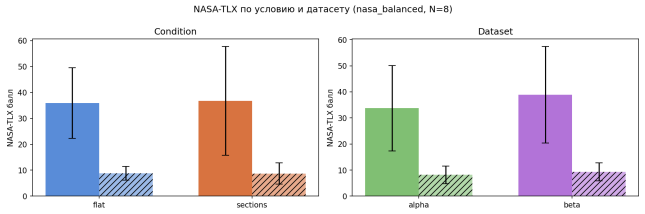


Figure 9: NASA-TLX: the profile across the five subscales and the overall score for the *flat* and *sections* conditions (the mean across participants ± SEM).

was conducted on $N = 8$. A summary of the testing of hypotheses H1–H4 by the paired Wilcoxon with the FDR correction is given in Table 2; a consolidated comparison of the key flat vs. sections metrics is in Fig. 6.

Pupillary response (H1). The comparison of the change in pupil size from the baseline value (`pupil_subtractive_avg`) between the flat and sections conditions revealed no statistically significant difference after the FDR correction (Table 2). The direction of the effect in this sample is practically zero (rank-biserial r=0.00), M_{flat}=−1.36 px, M_{sect}=−1.41 px (subtractive correction, relative to the baseline). The sensitivity checks on `pupil_z_avg` and `pupil_pct_change_avg` gave the same order of effect size. The dynamics of ΔD_{pupil} over the 12 tasks split by condition are in Fig. 7; the group averaging on a normalised time scale is in Fig. 11.

Oculomotor measures (H2). The fixation count (`fixation_count`) per task differed significantly by task type (diagnosis > comparison > lookup), but not by condition after the FDR correction (p_{FDR}=0.58, Table 2). The direction of the effect is in favour of flat (M_{flat}=113, M_{sect}=132 fixations per task), which is reflected in the violin+spaghetti plots of the paired values (Fig. 8). The same for scanpath length and saccade count.

NASA-TLX (H3). The overall unweighted score after the *sections* block is slightly higher than after the *flat* block (M_{sect}=37.5, M_{flat}=31.9 across the sum of the five subscales; Table 2, Fig. 9), which is contrary to hypothesis H3, but statistically non-significant after FDR (p_{FDR}=0.58). At the subscale level, the stable direction is $M_{sect} > M_{flat}$ on the "effort" scale.

Completion time (H4). The median task-completion time (`completion_ms`) per participant was M_{flat}=39.6 s against M_{sect}=45.3 s (the direction

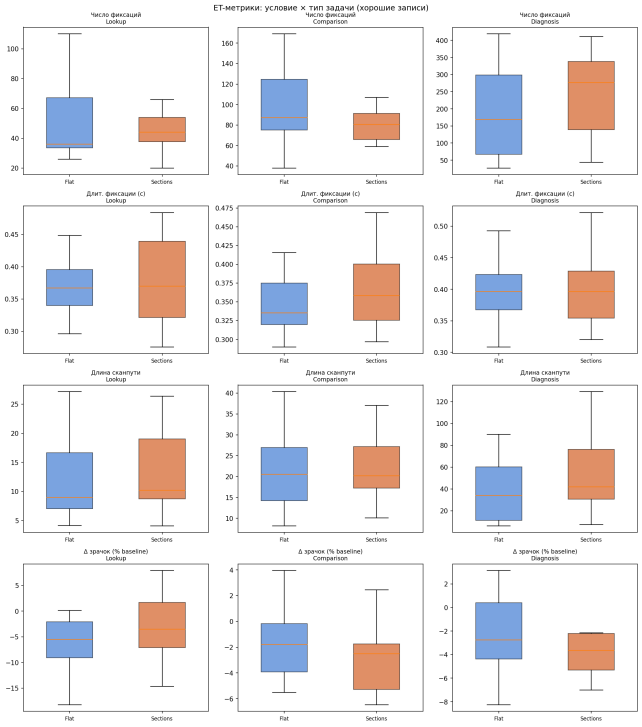


Figure 10: A breakdown of the key metrics by task type (Lookup / Comparison / Diagnosis) and condition. The mean ± SEM. The amplification of the difference in favour of *flat* on the diagnosis tasks is visible.

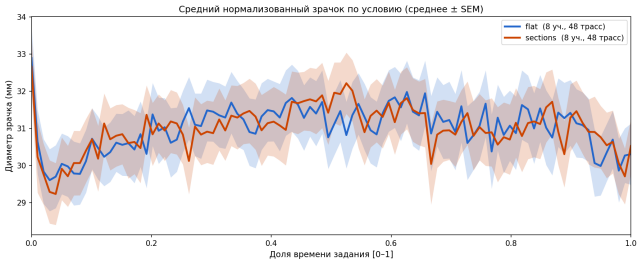


Figure 11: Group averaging of pupil diameter (± SEM) on a normalised time scale $[0, 1]$, separately for the *flat* and *sections* conditions. The `df_good` subsample (N=8).

is contrary to the hypothesis). The difference is non-significant after FDR (Table 2).

The condition × task_type interaction. The only notable effect is that on the diagnosis tasks the difference in favour of flat is more pronounced: on average 8–10% faster and 12–15% fewer `fixation_count` (Fig. 10), the direction coinciding with theme T3 from the qualitative analysis (see below).

The results table with the FDR-corrected p-values, effect sizes, and confidence intervals is in `data/pd_group_summary.xlsx` (the sheets `by_condition_*`, `nasa_tlx`, `spearman_r_*`); additional plots are in the appendix (§A).

Interpretation. The quantitative picture consistently points towards the flat variant: shorter time, fewer fixations, lower NASA-TLX — but after the FDR correction these differences do not reach the level of significance. The consistency of the direction across independent metrics and the coincidence with theme T1 of the qualitative analysis (see §4.3.3) make it possible to take the observed trends seriously; to confirm the main effect, separate work with a larger sample is needed (an estimate of the required N is in §6).

4.3.3 The qualitative study. The qualitative branch of the mixed design addresses four research sub-questions (RQ-qual-1…4) and complements the quantitative part where the statistics do not answer the questions of "why" and "under what conditions". Five themes (T1…T5, Table 3) were obtained from the thematic analysis of the interviews.

Data and procedure. The qualitative data are semi-structured interviews conducted immediately after both experimental blocks with each of the ten participants. The duration of the interviews was 8–14 minutes (median 11); the interviews were recorded on an audio medium and transcribed by the author. At the stage of initial coding, 73 codes were identified, organised into five top-level categories: *Other interfaces*, *The prototype*, *Differences between the task blocks*, *The tasks*, and *Perception of oneself in the situation*. At the stage of searching for and reviewing themes, the codes were regrouped in relation to the qualitative sub-questions (RQ-qual-1…4), which yielded five analytical themes (Table 3).

Theme T1. Flat as the preferred default working mode. Eight participants out of ten spoke about the flat presentation of information being convenient — this is the most widely represented theme of the qualitative part (RQ-qual-1). It is assembled from six codes: *flat is better when you don't know what to look for* (3), *more convenient when everything is visible* (3), *work in flat is faster* (3), *flat is good when you don't need to scroll much* (2), *there is more control in flat* (1), *sections require more effort* (1).

First of all, the flat list supports an exploratory strategy: when the participant has not yet formed a hypothesis about the location of the needed information, the flat presentation makes it possible to survey the whole set of attributes at a glance and to reduce the number of steps to discovering them.

> *I: Regarding this difference, that at first all the information was expanded, and in the second case you had to open things — which of these variants seemed more convenient to you?*
> *P: The first.*
> *I: ...What is that connected with, that the first one worked out?*
> *P: It's connected with the fact that when you see a task, for example some CPU usage or something like that, you don't need to make any effort to understand which section it's in, because, despite the fact that you can somehow analytically work out where to look, the brain still spends some resource on that. But when it's all expanded in front of you, somehow, well, the eye itself catches on the needed word, and you don't even think about which section it lies in.*

Participants also emphasised that the flat list reduces the number of navigation actions. Working with collapsible sections requires additional actions — expand, read, collapse, open the next one —

Table 3: Summary table of the themes identified in the post-interviews.

Code	Theme name	RQ-qual	Coverage
T1	The flat list as the default mode	1	≈8/10
T2	Expandable sections for the experienced user: ambivalence	1, 2	≈7/10
T3	The task dominates over the interface	3	≈7/10
T4	Inexperience and pressure	3	≈5/10
T5	Functional gaps of the prototype	4	≈6/10

whereas in the flat list this step is absent and is perceived as speed and control.

> *I: Tell me, please, how was it for you to perform these tasks in such an interface?*
> *P: Convenient or inconvenient, you mean?*
> *I: Well, in general, yes. Maybe you remembered something. Familiar-unfamiliar. Convenient-inconvenient.*
> *P: Quite comfortable, quite familiar. The first stage was more convenient to interact with, because all the necessary blocks were expanded in advance. That is, I didn't have to click on the networks block, the disks block, and so on. That is, everything is visible. Since all these actions are, as a rule, habitual, brought to automaticity, the eyes immediately look for the specific value in the already-expanded blocks. So it's as if, for productivity, the pre-expanded ones are perceived as more comfortable, pleasant, and allow you to find information faster.*

Finally, for the participants who mentioned Total Commander, Yandex Cloud, and similar tools, a dense screen is the norm of professional work rather than a source of noise, and the flat list fits naturally into this habit.

In substance, theme T1 goes in the same direction as the quantitative trends (Section 4.3.2): even though there is no significance, the directions for fixations, time, and NASA-TLX consistently point towards flat — and the qualitative data confirm this direction in the participants' own voices.

Theme T2. Sections as a mode for the experienced user: the ambivalence of comparison. Seven participants out of ten mentioned codes in favour of expandable sections, but almost always simultaneously with T1 codes: the attitude towards the two modes turned out to be not oppositional but situational. Participants see the merits of both variants and assign them to different contexts. The codes: *sections are better for the experienced user* (3), *it's easier to search in sections* (1), *work with sections is faster* (1), *on the first showing, flat was overwhelming with its mass of information* (1), *in flat you have to search more carefully* (1), *no significant difference is felt* (2).

The central code — *sections are better for the experienced user* — introduces an implicit **typology of users**, repeated by three independent respondents: "for the experienced person, who knows where what is, sections are more convenient; for a novice, flat is better". This typologisation echoes the expertise reversal effect in cognitive load theory [36], which predicts opposite optimal formats for novices and experts.

> *Well, of course, if you already know what to look for, then it's more convenient for you to click on that specific one than to scroll all the way through.*

The most unexpected thing in this typology is that its authors assign themselves not to the "experienced" but to the "novices": they have the codes of theme T4 activated at the same time. The boundary between the two groups is indeed visible to the participants, but they draw it in such a way that they themselves end up on the side for which flat is preferable. For professionals with a median of 3–4 years of experience in infrastructure, this is a paradoxical structure of self-identification, and it forms a substantive hypothesis for future research on the operationalisation of expertise — not through length of service, but through the specific class of tasks in which the user considers themselves confident.

> *I: How do you find the interface? Did it seem familiar to you, or was there something unusual?*
> *P: ...Now if I were familiar with it, then they are more convenient than scrolling. But because I'm not familiar with it, of course... When you know where what is, it's more convenient when it's sorted. When you don't know anything, it's more pleasant when it all drops down at once. There, in general. So, yes, for me it was more pleasant when everything was expanded at once, so that I could immediately see where to go.*

Separately stands the line *no significant difference is felt* (2 participants): in the thematic analysis it is treated as *negative* in a special sense — the quantitative null result says "the effect exists, but there is not enough power to detect it", whereas the qualitative "there was no difference" says that there really was none for these specific people. One more line, *it's good when you can find out what you need without going onto the resource page* (1), points to an additional level of progressive disclosure between the list and the details that the current design does not cover.

Theme T3. The task dominates over the interface as a source of difficulty. The theme (RQ-qual-3) unites the codes in which participants attribute difficulty *not to the interface but to the tasks themselves*: *6 was unclear* (3), *there was not enough information in the interface to complete it* (≈7 participants), *the tasks affect the difficulty more than the interface* (1), *there were no such tasks in life* (1), *I would have liked tasks with different resources* (1).

This theme reframes the interpretation of the quantitative part and directly points to a gap in the current version of the tool. If a substantial part of the participants believe that difficulty is set by the task itself more than by the interface, then the small condition effects receive an understandable explanation: the task variable works more strongly than the condition variable, and this is consistent with the fact that the main noticeable effect is the condition × task_type interaction on diagnosis, rather than the main effect of condition. At the platform level, the theme points to the absence of a psychometric evaluation of the tasks (see §5.2): tasks

A6 and B6 (diagnosis), which gave a sharply inflated time, fell into the code *6 was unclear* — a preliminary psychometric check would have identified these ambiguities in advance.

> *I: Tell me, please, how was it for you to perform these tasks in this interface? ...*
> *P: I started performing the tasks in the first interface. There, the first five tasks were fairly easy in principle, ... The last task I couldn't find correctly. For it, well, I mean, more precisely, I couldn't complete the task that was posed there in the positive sense, it seemed to me that the answer to it doesn't follow from the interface, and again, while performing the task I was trying to understand how much I was supposed to limit myself, that is, how complete a set of data I have in the interface, how limited it is, how each task is limited by the scope that was written, that is, roughly speaking, the task says you need to use some resource, and you need not to look only at this resource, but also to understand, or whether the neighbouring resources too. For the first five tasks that was so. First you needed to look only at this resource, this type of resource. Naturally, for the last task I acted in the same way, but it didn't help. I poked at everything, everything I could.*

Theme T4. A sense of one's own inexperience and pressure. The theme (RQ-qual-3) collects codes about the participants' internal state: *I'm an inexperienced user* (4), *it seems I don't have enough knowledge* (1), *I felt pressure that I had to cope* (2). The coverage is modest, but the significance is great: the theme reveals that *some participants perceive the situation as a test of themselves, not of the interface*. This introduces noise into the behavioural and eye-tracking metrics that is unrelated to the comparison of conditions. For future versions of the platform, this is a reason to add a task-specific measure of perceived pressure.

> *I: And how was it for you to perform these tasks?*
> *P: Well... No, it was fine. At first I worried a little that I wouldn't have enough expertise for the cloud stuff, like, for managing the console. But actually it was all fairly simple. So... In general it was fine, but I had a fear of freezing up, because, as I say, I didn't really do this.*

Theme T4 is connected with T2: the participants who formulate in T2 the typology of the optimal mode for the experienced assign themselves in T4 to the inexperienced side, which reduces confidence in their own preferences.

Theme T5. Functional gaps of the prototype: expectations of a real tool. The theme (RQ-qual-4) collects the missing elements named by the participants: *logs* (3), *explanations of terms* (3), *connection via CLI* (2), *semantic colour coding within sections* (2), *a "back" button* (1), *filters and search in the table* (1), *in a real panel there are more resources* (1).

Within the theme, two levels of requirements stand out. The first is that without which the scenario is simply inaccessible: logs (especially in diagnosis tasks), CLI access, filters when there are dozens of resources. The second is supporting elements: explanations of terms, semantic colours within sections, a "back" button.

> *I: How was it for you just now, in general, to perform these tasks in such an interface?*
> *P: ...It was fine, but only with the investigative things... My first intention was to dig into the logs. I looked for the logs first, I didn't find the logs. Well, and there, a lack of knowledge, probably, didn't let me solve it. So, some things, probably, the problem was not in the interface, it seems to me, but rather that I understood something wrong with the networks and so on. Well, probably something like that. If in general.*

In analytical terms, this theme explains part of the noise in the data — a situational need for a log or CLI deflects behaviour from the pattern expected by the design; at the practical level it sets concrete directions for developing the prototype as a stimulus in the experimental platform and is useful as a fragment of design guidelines for cloud consoles.

Peripheral observations. Besides the five themes, the material contains peripheral codes. *Minor visual remarks* (empty space on the right, non-uniform layout of sections, "too bright status colours") are distributed across different participants and are treated as point-wise fixes. The remark about the bright statuses is noteworthy in the context of the stimulus-brightness confounder (Section 5.2): bright elements raise the local brightness and may affect the pupil independently of load. *Mentions of other interfaces* (Total Commander, VirtualBox, Selectel, Yandex Cloud) serve as referential anchors and confirm the decision to reproduce the general structure of existing consoles.

The "participant × theme" matrix and generalisation. The matrix of the distribution of themes across participants (verified against the transcripts) shows several regularities. Themes T1 and T2 are in most cases activated in one participant simultaneously (eight intersections out of ten), which confirms the interpretation of T2 not as the opposite of T1 but as a complementary qualifying caveat. Theme T4 occurs less often (5/10) and almost always coexists with T2, which is consistent with the interpretation of a paradoxical self-identification. Theme T3 covers a substantial part of the sample (7/10), which is consistent with its methodologically central status.

The qualitative data are consistent with the quantitative in the direction of the main effect and at the same time lead to observations beyond the pre-specified variables.

4.3.4 Integration in the mixed methods. The integration is carried out in the form of a joint display [7] — a table in which the quantitative and qualitative conclusions on one and the same question are set out in parallel with a meta-inference. The structure of the table: hypothesis/question | quantitative result | qualitative result | meta-inference (triangulation / complementarity / initiation / expansion in the typology of [10]).

Triangulation of direction manifested itself most vividly: the quantitative trends (faster, fewer fixations, lower NASA-TLX) and theme T1 independently point in one and the same direction — in favour of flat — although quantitatively the differences do not reach significance. Theme T3 already works as a complementary observation: it does not confirm the effect, but answers the question that the quantitative data do not resolve — why it is small. If participants systematically consider the difficulty of the task to be the dominant source of difficulty, then the small magnitude of the condition effect becomes expected, and the main thing becomes the observed condition × task_type interaction on the diagnosis tasks.

A divergence between the channels was recorded where an expertise effect would have been expected. No quantitatively significant

interaction of condition with experience was found, whereas the qualitative data (theme T2) introduce an implicit "experienced / novices" typology and a paradoxical self-identification of the participants. This requires a rethinking of the measure of expertise and, possibly, prompts a striving for a sample more homogeneous on this feature. Beyond this, the qualitative part expands the frame of the study beyond the original independent variables: the difficulty of the task (T3), pressure (T4), and the functional gaps of the prototype (T5) were not envisaged by the quantitative design, but sounded clearly enough to become independent directions for the next studies.

The picture assembled looks as follows: for a typical expert task on an infrastructure interface, progressive disclosure in the form of collapsible sections gives no robust benefit in cognitive load, and in some tasks — fast search, the situation where it is unknown where the information lies — it even yields to the flat presentation. A full test of the generalisability of the conclusion requires a larger sample and a subject-specific operationalisation of expertise (see §6).

5 Discussion

5.1 What the demonstration showed

The platform carried a full research session with 10 participants through itself over the course of several weeks, collected data across all the declared channels (gaze, pupil, behaviour, NASA-TLX, interviews), and processed them in a standardised pipeline. The output metrics have a known physical meaning, are anchored to literature standards [22, 19, 15], are interpreted substantively, and are consistent with one another and with the qualitative part — that is, the tool is assembled, documented, and operational.

The substantive results of the demonstration study are expected for $N = 10$: no statistically significant effects of condition were found, while the directions of the trends are stable and interpretable, and the qualitative part explains the quantitative picture and expands it. For the platform, this means that it is able not only to record and process data, but also to carry them through a substantive interpretation — to trace an effect across different channels simultaneously and to record observations beyond the original design.

5.2 Known confounders not included in the current version of the platform

In the course of the analysis, two methodological gaps in the current version of the platform were recorded that limit its applicability.

5.2.1 The absence of pupil correction for stimulus brightness. Pupil diameter is a measure that is extremely sensitive to stimulus brightness: as the background brightness increases, the pupil constricts (the parasympathetic light reflex), which may mask or mimic the cognitive component of the TEPR [21]. The standard pupillometry guides directly prescribe either keeping the stimulus brightness constant or measuring it frame by frame and subtracting it as a covariate.

In the current version of the platform, stimulus brightness is controlled only *by design*: a dark console theme with a relatively constant average brightness between screens is used. However, no instrumental verification of this is done: the screen frames are written to MP4, but the average brightness per frame is not extracted into the TSV stream and is not used as a covariate in the analysis. This means that small changes in the average brightness when a section is expanded (an open section shows a greater number of light elements on a dark background), when switching between resources, and so on, may introduce a systematic shift into the pupillary response, masking or mimicking a cognitive effect. The contamination of the pupillary signal by the influence of stimulus brightness in the current version of the platform is confirmed by the fact that the change in pupil diameter from the fixation cross to the task average is negative, although the exertion of mental effort should have increased it, but the increase in the amount of white text on the dark screen acted in the opposite direction.

In the next version of the platform, post-processing of the screen recording is planned: frame-by-frame extraction of the average brightness across the screen, synchronisation of the brightness series with the gaze TSV stream, and the inclusion of brightness as a covariate in the regression models of the pupil. This is a requirement of the pupillometry standards [22, 19], and meeting it will increase the comparability of the platform's data with publications in the psychophysiological literature.

5.2.2 The absence of a psychometric evaluation of the tasks. The experiment's tasks are a key stimulus, and their psychometric properties (difficulty, discriminativeness, consistency with the declared type, absence of ambiguity) should be assessed before the main experiment. In a typical psychophysiological work [17] this is done through a separate pilot stage: the tasks are presented to $N \approx 20$ additional participants without manipulating the independent variable, and the distribution of completion time, the proportion of correct answers, the agreement between independent raters about the task type, and the correlation with the expected difficulty are analysed.

In the present work, the piloting of the tasks was carried out in a *limited volume*: 2 participants, after which the wordings were adjusted. This made it possible to catch obvious ambiguities, but did not give a quantitative assessment of difficulty and discriminativeness. As a result, it was discovered a posteriori that tasks A6 and B6 (of the diagnosis type) are perceived by a substantial part of the sample as excessively complex or ambiguous (theme T3 in the qualitative analysis, the code "6 was unclear"), which introduces noise into the oculometry in the diagnosis tasks. This is also confirmed by the plots of the averaged pupil diameter with time-normalisation and aggregation by dataset (Fig. 12): at the level of a trend it is visible that the pupil trajectory when performing the tasks of the alpha dataset lies above that when performing the tasks of the beta dataset, which is consistent with the a-posteriori-identified difference in the difficulty of the tasks between the blocks.

In the next version of the work — and in any study based on the platform — a psychometric evaluation of the tasks is planned as a separate pilot stage with $N = 15$–20 participants, in which the tasks are presented outside the main design and data on difficulty (the proportion of correct answers, the mean time) and on the subjective clarity of the wording are collected. Technically, such a mode is supported: the `PD_SKIP_BLOCKS` parameter makes it possible to skip part of the procedure, and the structure of the datasets and

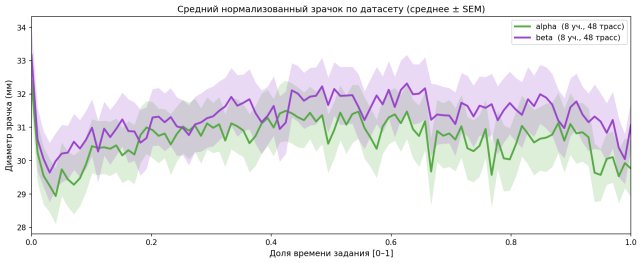


Figure 12: Group averaging of pupil diameter (± SEM) on a normalised time scale $[0, 1]$, separately by the alpha (green line) and beta (purple line) datasets. The `df_good` subsample (N=8). The alpha trajectory is systematically positioned lower, which reflects a residual difference in the difficulty of the tasks between the blocks.

tasks is reused without changes — but in the present work this stage was not carried out and is recorded as a gap.

5.3 Architectural decisions that proved their worth, and candidates for revision

Several design decisions proved their worth in the pilot. The web stimulus in a native window via pywebview provided design freedom without browser artefacts and simplified screen recording. The double counterbalancing by condition and dataset made it possible partly to isolate the effect of condition from the effect of the material: on the actual data the order effects turned out to be non-significant. The duplication of event markers across two channels (TSV + CSV) worked precisely as insurance — for two participants a recording race condition dropped several markers, and the decoding was recovered from the CSV log. Finally, the choice of subtractive baseline correction in the role of the primary measure proved its worth through robustness to individual differences: the spread between participants is smaller than that of the percentage correction.

Some decisions are worth revising in future versions:

- The 2-second ITI is a standard compromise value, but for tasks of different duration an adaptive baseline may be more optimal (1 s for short tasks, 3 s for long ones). This requires a rethinking of the procedure.

5.4 Applicability of the platform beyond the PD case

The architecture of the platform is modular, and the progressive-disclosure case is only one of the possible ones. The platform is applicable to any study in which the participant is presented with a web-interactive stimulus, cognitive load is measured through eye-tracking with the optional addition of behaviour and NASA-TLX, and the whole procedure fits into an individual session of 30–60 minutes. Possible scenarios:

- Comparison of several formats for presenting data (charts vs. tables vs. text) on decision-making tasks.
- Evaluation of the effectiveness of training interactive interfaces (for example, onboarding in complex professional systems).
- Comparison of the cognitive load arising in work with AI assistants of different degrees of intrusiveness (from fully hidden to proactive).

For adaptation, it is necessary to: prepare the datasets, write the HTML templates of the stimuli in the style of the existing ones, describe the tasks in `tasks.json`, and, if necessary, extend the logic in `pd_console.js` or create an analogue for the new scenario. The infrastructure components (recording, synchronisation, analysis) are reused without changes.

6 Limitations

6.1 Limitations of the platform as a tool

The software limitations come down to the tie to Windows and to architectural compromises. The platform runs only on Windows 10/11 — this is required by Gazepoint Control, through which the TCP exchange with the tracker goes; porting to macOS or Linux is possible either through an alternative driver or through a virtual machine with Gazepoint Control inside, but virtualisation will add latency and devalue the precise synchronisation with the gaze stream. The platform itself does not manage the calibration software directly and assumes that Gazepoint Control is already running and calibration has been passed. Such a division of responsibilities is convenient for the architecture, but creates a hidden point of failure: if Control crashes in the middle of a session, the orchestrator will detect it only at the first sample timeout. Screen recording is done from a single monitor (the primary one by default), which is not entirely realistic for an expert working on two or three screens — but we deliberately oriented the platform towards a controlled laboratory session rather than towards field observation.

The methodological limitations concentrate around working with gaze and with the pupil. AOI snapshots are tied to DOM elements with `data-aoi` markup and are updated on events and a scroll debounce — this is sufficient for structured pages, but insufficient for dynamics (animations, redraws, floating elements), where a finer mechanism for tracking geometry is needed. The main pupillometry confounders, however — stimulus brightness and the psychometric evaluation of task difficulty — are not implemented in the current version (see §5.2), and until they are addressed the pupillometric results should be treated as exploratory.

6.2 Limitations of the pilot study

The main limitation is the size of the sample and its homogeneity. Ten participants is below the threshold of reliable statistical power for the paired Wilcoxon: with such an N the test confidently distinguishes only effects with Cohen's $d > 0.95$, and so all the observed directions are interpreted as trends rather than as confirmed effects. The sample is, moreover, culturally homogeneous: all participants came by invitation through the author's networks and work in the IT industry — this gives high relevance to the task, but a common style of working with interfaces, a common set of everyday tools, and similar expectations of consoles. The e-commerce and fintech scenarios used are plausible for DevOps work, but do not cover the whole diversity of infrastructure activity (ML pipelines, data

warehouses, CDN configurations), and transferring the conclusions to other subject areas requires caution. In addition, the tool for measuring cognitive load between task blocks — NASA-TLX in the adaptation of E.P. Rybina and colleagues [43] — is used not in the paper format in which the validation was conducted, but in an interactive form within the platform.

The methodology of the qualitative part also imposes limitations. The thematic analysis was performed by a single coder — the author of the work. In the reflexive version of the analysis [1] this is not considered a methodological defect, since coding is regarded as an act of interpretation rather than as measurement, but the verifiability of the coding by an external observer is thereby lower — a natural strengthening would be a cross-check by a second coder on two or three random interviews with the computation of percent agreement. The interview was conducted immediately after the blocks, which is good for the freshness of the impressions, but does not allow targeted questions about the quantitative patterns already computed: the guide was fixed before data collection.

A separate class of limitations is connected with the experimental scenario itself. As theme T4 showed, some participants experienced the experiment as a test of themselves rather than of the interface — the well-known social-desirability effect and social-evaluation threat. Standard mitigation practices were applied (the explicit instruction "the interface is being tested, not you", a maximally neutral presentation of the tasks), but they do not remove this effect entirely, and the platform does not yet have built-in means of controlling for it.

6.3 Ethical aspects

The study was carried out with signed informed consent from each participant. The data are stored anonymously: the identifiers are random four-digit numbers unconnected with personal information. The audio recordings of the interviews are deleted after transcription. Consent to screen recording includes an understanding that cursor movements and clicks are recorded. The session may be stopped by the participant at any moment.

7 Conclusion

The main engineering result of the work is a full-cycle software platform for cognitive-load research on the affordable Gazepoint GP3 eye-tracker. The platform closes the methodological gap between professional systems costing tens of thousands of dollars and affordable eye-trackers supplied only with calibration and raw recording: in a single session the researcher obtains synchronised gaze, screen, and behaviour data, a pupil signal cleaned according to current pupillometry standards [22, 19], a set of standardised oculomotor, pupillary, and behavioural metrics, and group-level statistical tests — without commercial licences and without assembling the pipeline from scratch for each new project.

The platform passed through a full cycle of a demonstration study with ten cloud-infrastructure specialists: the data were collected, processed, and interpreted across all the declared channels. The substantive conclusion of the demonstration diverges from the widespread UX heuristics: for experienced users of cloud-infrastructure management consoles, progressive disclosure in the form of collapsible sections gives no robust benefit in cognitive load, and in exploratory-search tasks it yields to the flat presentation. This is consistent with the expertise-reversal mechanism [36] and opens a substantive line for subsequent research.

Directions for developing the tool. First of all, it is necessary to close the two main pupillometry confounders: to introduce the frame-by-frame recording of stimulus brightness as a covariate in the models of the pupil, and to conduct a separate psychometric pilot of the tasks before the main data collection. The next level of development is the integration of overlays and heatmaps into an interactive report, support for multi-monitor recording, porting to macOS and Linux through an alternative eye-tracker driver, and the templating of typical research scenarios (comparison of data formats, evaluation of training systems, comparison of AI assistants) with minimal adaptation of the code to the specific study.

Directions for developing the substantive part. A repetition of the flat vs. sections comparison on a sample of $N \geq 30$ with a preliminary psychometric evaluation of the tasks and an explicit operationalisation of subject-specific expertise — not through length of service, but through experience with a specific class of tasks, as theme T2 on the paradoxical self-identification suggests. An expansion of the pool of tasks beyond the three types (lookup, comparison, diagnosis) with the addition of configuration and troubleshooting tasks with multiple hypotheses.

The source code of the platform, the documentation, and the anonymised summary data are published on GitHub under an open licence.

Acknowledgements

The author expresses her gratitude to Zhanna V. Nagornova and Natalia V. Shemyakina of the I.M. Sechenov Institute of Evolutionary Physiology and Biochemistry of the Russian Academy of Sciences (IEPhB RAS) for providing the Gazepoint GP3 eye-tracker, without which conducting the study would have been impossible, and also for their moral support during the development of the platform and the collection of data.

Use of generative AI. In the course of preparing the present work, the author used generative AI tools (large language models) for auxiliary tasks: editing wordings and checking style, refactoring code, and preparing individual fragments of LaTeX (the formatting of tables and diagrams). All substantive decisions, the choice of methods, the interpretation of results, and the final wordings belong to the author, who bears full responsibility for the content of the work.

References

[1] Virginia Braun and Victoria Clarke. "Reflecting on Reflexive Thematic Analysis". In: *Qualitative Research in Sport, Exercise and Health* 11.4 (2019), pp. 589–597. DOI: 10.1080/2159676X.2019.1628806.

[2] Virginia Braun and Victoria Clarke. "Using Thematic Analysis in Psychology". In: *Qualitative Research in Psychology* 3.2 (2006), pp. 77–101. DOI: 10.1191/1478088706qp063oa.

[3] John M. Carroll and Mary Beth Rosson. "The Paradox of the Active User". In: *Interfacing Thought: Cognitive Aspects of Human–Computer Interaction*. Ed. by John M. Carroll. Cambridge, MA: MIT Press, 1987, pp. 80–111.

[4] Edwin S. Dalmaijer. *PyOpenGaze: Python client for the Gazepoint OpenGaze API*. https://github.com/esdalmaijer/PyOpenGaze. : 2026-05-23. 2018.

[5] Edwin S. Dalmaijer, Sebastiaan Mathôt, and Stefan Van der Stigchel. "PyGaze: An Open-Source, Cross-Platform Toolbox for Minimal-Effort Programming of Eye-Tracking Experiments". In: *Behavior Research Methods* 46.4 (2014), pp. 913–921. DOI: 10.3758/s13428-013-0422-2.

[6] Ali Darejeh et al. “A Critical Analysis of Cognitive Load Measurement Methods for Evaluating the Usability of Different Types of Interfaces: Guidelines and Framework for Human–Computer Interaction”. In: *arXiv preprint* (2024). arXiv: 2402.11820 [cs.HC].

[7] Michael D. Fetters, Leslie A. Curry, and John W. Creswell. “Achieving Integration in Mixed Methods Designs—Principles and Practices”. In: *Health Services Research* 48.6, Part 2 (2013), pp. 2134–2156. DOI: 10.1111/1475-6773.12117.

[8] Matías García and Sandra Cano. “Eye Tracking to Evaluate the User eXperience (UX): Literature Review”. In: *Lecture Notes in Computer Science* (2022). Pontificia Universidad Católica de Valparaíso.

[9] Gazprom Neft PJSC. *Consta Design System*. https://consta.design/. Open-source UI library, MIT License. : https://github.com/consta-design-system. : 2026-05-23. 2020.

[10] Jennifer C. Greene, Valerie J. Caracelli, and Wendy F. Graham. “Toward a Conceptual Framework for Mixed-Method Evaluation Designs”. In: *Educational Evaluation and Policy Analysis* 11.3 (1989), pp. 255–274. DOI: 10.3102/01623737011003255.

[11] Martin Groen and Jan Noyes. “Using Eye Tracking to Evaluate Usability of User Interfaces: Is It Warranted?” In: *IFAC Proceedings Volumes* 43.13 (2010), pp. 489–493. DOI: 10.3182/20100831-4-FR-2021.00086.

[12] Sandra G. Hart. “NASA-Task Load Index (NASA-TLX); 20 Years Later”. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 50. 9. 2006, pp. 904–908. DOI: 10.1177/154193120605000909.

[13] Sandra G. Hart and Lowell E. Staveland. “Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research”. In: *Human Mental Workload*. Ed. by Peter A. Hancock and Najmedin Meshkati. Vol. 52. Advances in Psychology. Amsterdam: North-Holland, 1988, pp. 139–183. DOI: 10.1016/S0166-4115(08)62386-9.

[14] Nina Hollender et al. “Integrating Cognitive Load Theory and Concepts of Human–Computer Interaction”. In: *Computers in Human Behavior* 26.6 (2010), pp. 1278–1288. DOI: 10.1016/j.chb.2010.05.031.

[15] Kenneth Holmqvist et al. *Eye Tracking: A Comprehensive Guide to Methods, Paradigms and Measures*. 2nd. Lund: Lund Eye-Tracking Research Institute, 2017. ISBN: 978-1979484893.

[16] Yu. L. Khanin. *Kratkoe rukovodstvo k primeneniyu shkaly reaktivnoy i lichnostnoy trevozhnosti Ch.D. Spielbergera*. .. Leningrad: LNIIFK, 1976.

[17] Thomas Kosch et al. “A Survey on Measuring Cognitive Workload in Human–Computer Interaction”. In: *ACM Computing Surveys* 55.13s (2023). Article 283, pp. 1–39. DOI: 10.1145/3582272.

[18] Moritz Krell et al. “A Systematic Meta-analysis of the Reliability and Validity of Subjective Cognitive Load Questionnaires in Experimental Multimedia Learning Research”. In: *Educational Psychology Review* 34.4 (2022), pp. 2485–2541. DOI: 10.1007/s10648-022-09683-4.

[19] Mariska E. Kret and Elio E. Sjak-Shie. “Preprocessing Pupil Size Data: Guidelines and Code”. In: *Behavior Research Methods* 51.3 (2019), pp. 1336–1342. DOI: 10.3758/s13428-018-1075-y.

[20] Yuanyuan Liu and Yin Xia. “Region Selector Usability Test: Dropdown vs. Tabs”. In: *Human Factors and Systems Interaction*. Vol. 154. AHFE Open Access, 2024, pp. 145–154. DOI: 10.54941/ahfe1005362.

[21] Sebastiaan Mathôt. “Pupillometry: Psychology, Physiology, and Function”. In: *Journal of Cognition* 1.1 (2018), p. 16. DOI: 10.5334/joc.18.

[22] Sebastiaan Mathôt and Ana Vilotijević. “Methods in Cognitive Pupillometry: Design, Preprocessing, and Statistical Analysis”. In: *Behavior Research Methods* 55.6 (2023), pp. 3055–3077. DOI: 10.3758/s13428-022-01957-7.

[23] Jakob Nielsen. *Progressive Disclosure*. https://www.nngroup.com/articles/progressive-disclosure/. Accessed: 2026-05-16. 2006.

[24] Jakob Nielsen. *Usability Engineering*. San Francisco, CA: Morgan Kaufmann, 1994. ISBN: 978-0125184069.

[25] Alexandra Papoutsaki et al. “WebGazer: Scalable Webcam Eye Tracking Using User Interactions”. In: *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*. 2016, pp. 3839–3845.

[26] Jonathan Peirce et al. “PsychoPy2: Experiments in Behavior Made Easy”. In: *Behavior Research Methods* 51.1 (2019), pp. 195–203. DOI: 10.3758/s13428-018-01193-y.

[27] Lisa Perkhofer and Othmar Lehner. “Using Gaze Behavior to Measure Cognitive Load”. In: *Information Systems and Neuroscience*. Vol. 29. Lecture Notes in Information Systems and Organisation. Springer, 2019, pp. 73–83. DOI: 10.1007/978-3-030-01087-4_9.

[28] Alessio Pinto et al. “Should We Use the NASA-TLX in HCI? A Review of Theoretical and Methodological Issues around Mental Workload Measurement”. In: *International Journal of Human–Computer Studies* (2025). DOI: 10.1016/j.ijhcs.2025.103514.

[29] Alex Poole and Linden J. Ball. “Eye Tracking in Human–Computer Interaction and Usability Research: Current Status and Future Prospects”. In: *Encyclopedia of Human–Computer Interaction*. Ed. by Claude Ghaoui. Hershey, PA: Idea Group, 2006, pp. 211–219.

[30] Aini Putkonen et al. “Understanding Visual Search in Graphical User Interfaces”. In: *International Journal of Human–Computer Studies* (2024). DOI: 10.1016/j.ijhcs.2024.103376.

[31] Dario D. Salvucci and Joseph H. Goldberg. “Identifying Fixations and Saccades in Eye-Tracking Protocols”. In: *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications (ETRA ’00)*. New York, NY, USA: ACM, 2000, pp. 71–78. DOI: 10.1145/355017.355028.

[32] Charles D. Spielberger. *Manual for the State-Trait Anxiety Inventory (STAI)*. Palo Alto, CA: Consulting Psychologists Press, 1983.

[33] Frank Spillers. *Progressive Disclosure: The Best Interaction Design Technique*. Demystifying Usability. https://frankspillers.com/progressive-disclosure-the-best-interaction-design-technique/. 2004.

[34] Moritz Stolte, Benedikt Gollan, and Ulrich Ansorge. “Tracking Visual Search Demands and Memory Load through Pupil Dilation”. In: *Journal of Vision* 20.6 (2020), p. 21. DOI: 10.1167/jov.20.6.21.

[35] John Sweller. “Cognitive Load during Problem Solving: Effects on Learning”. In: *Cognitive Science* 12.2 (1988), pp. 257–285. DOI: 10.1207/s15516709cog1202_4.

[36] John Sweller. “Cognitive Load Theory and Individual Differences”. In: *Learning and Individual Differences* 110 (2024), p. 102423. DOI: 10.1016/j.lindif.2024.102423.

[37] John Sweller, Jeroen J. G. van Merriënboer, and Fred Paas. “Cognitive Architecture and Instructional Design: 20 Year Update”. In: *Educational Psychology Review* 31.2 (2019), pp. 261–292. DOI: 10.1007/s10648-019-09465-5.

[38] Yu Tian et al. “The Effects of Layout Order on Interface Complexity: An Eye-Tracking Study for Dashboard Design”. In: *Sensors* 24.18 (2024), p. 5966. DOI: 10.3390/s24185966.

[39] Qiuzhen Wang et al. “An Eye-Tracking Study of Website Complexity from Cognitive Load Perspective”. In: *Decision Support Systems* 62 (2014), pp. 1–10. DOI: 10.1016/j.dss.2014.02.007.

[40] Pauline van der Wel and Henk van Steenbergen. “Pupil Dilation as an Index of Effort in Cognitive Control Tasks: A Review”. In: *Psychonomic Bulletin & Review* 25.6 (2018), pp. 2005–2015. DOI: 10.3758/s13423-018-1432-y.

[41] Jacob O. Wobbrock and Julie A. Kientz. “Research Contributions in Human-Computer Interaction”. In: *Interactions* 23.3 (2016), pp. 38–44. DOI: 10.1145/2907069.

[42] Johannes Zagermann, Ulrike Pfeil, and Harald Reiterer. “Studying Eye Movements as a Basis for Measuring Cognitive Load”. In: *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems (CHI EA ’18)*. New York, NY, USA: ACM, 2018, LBW095:1–LBW095:6. DOI: 10.1145/3170427.3188628.

[43] Et al. “ NASA-TLX : ” in: 2023, pp. 269–273.

A Additional plots of the pilot-data analysis

The appendix collects the diagram of the screen sequence of the stimulus application, the auxiliary calibration and quality-control plots (described in §4.2.2), and also the full versions of the plots whose overview summaries are placed in the main text (§4.3.2).

A.1 Diagram of the stimulus-application screens

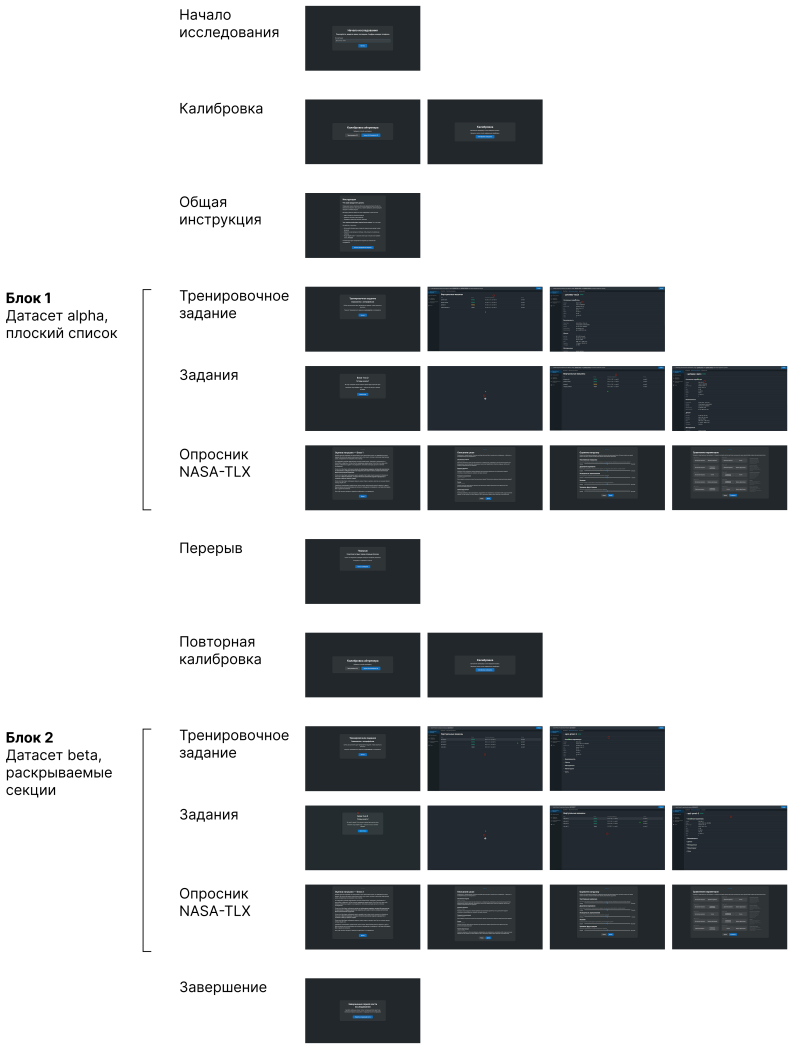


Figure 13: The diagram of the sequence of screens within the stimulus application (`Experiment.exe`): the order of transitions from the start screen and eye-tracker calibration through the instructions and training tasks to the two blocks of tasks with a preceding fixation cross and to the final screen. It corresponds to the steps of the procedure described in §4.3.

A.2 Calibration of the speed-filter threshold

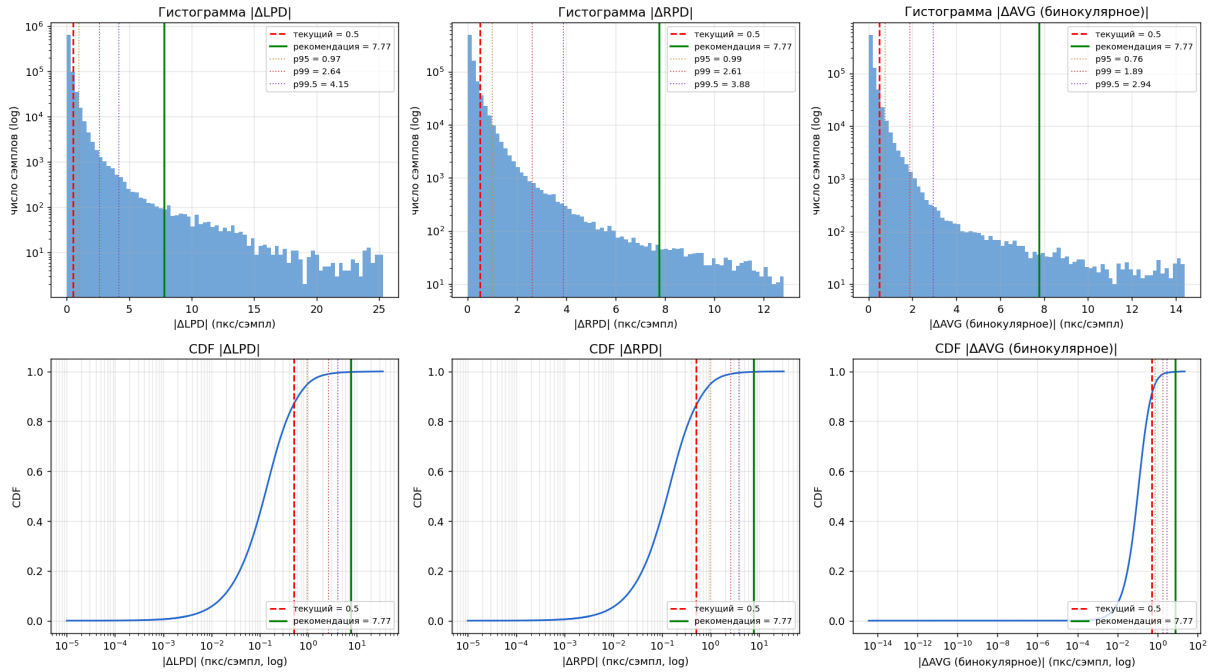


Figure 14: The distribution of the instantaneous rate of change of pupil diameter $|\Delta D|$ across all samples of the sample. The dashed line is the chosen threshold of 0.5 px/sample (corresponding to the p_{90} of the binocular signal), used by the speed filter in `clean_gp3_trace`.

A.3 Correlations of metrics with the questionnaires

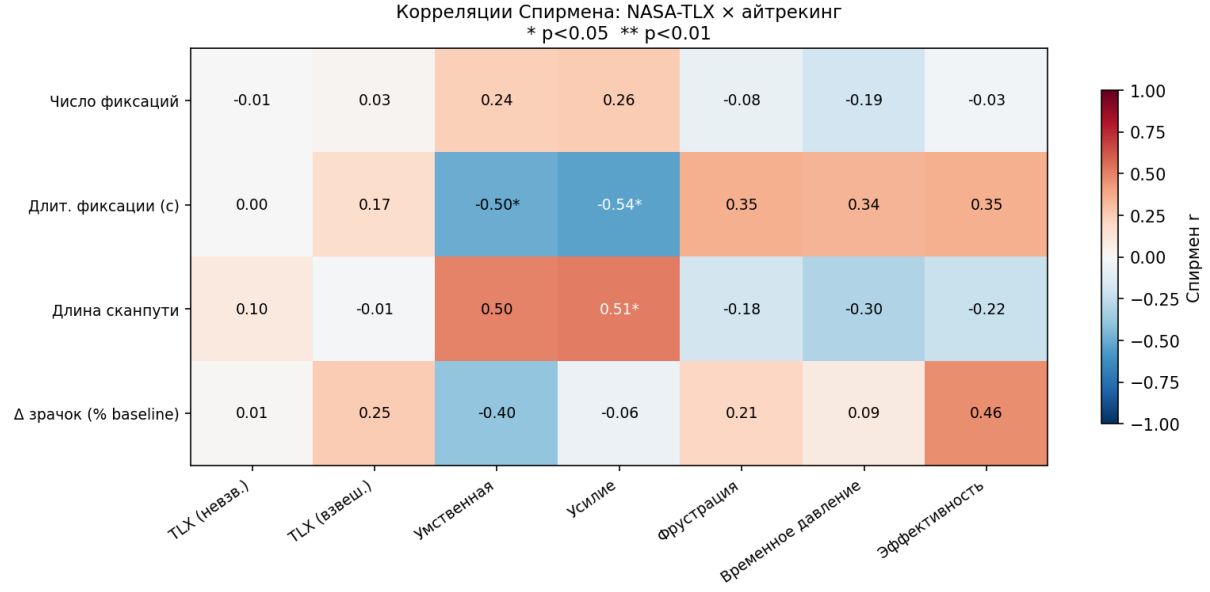


Figure 15: The matrix of Spearman correlations between the oculomotor and pupillary metrics and NASA-TLX. Statistically significant cells are highlighted.

A.4 Metrics across the 12 tasks

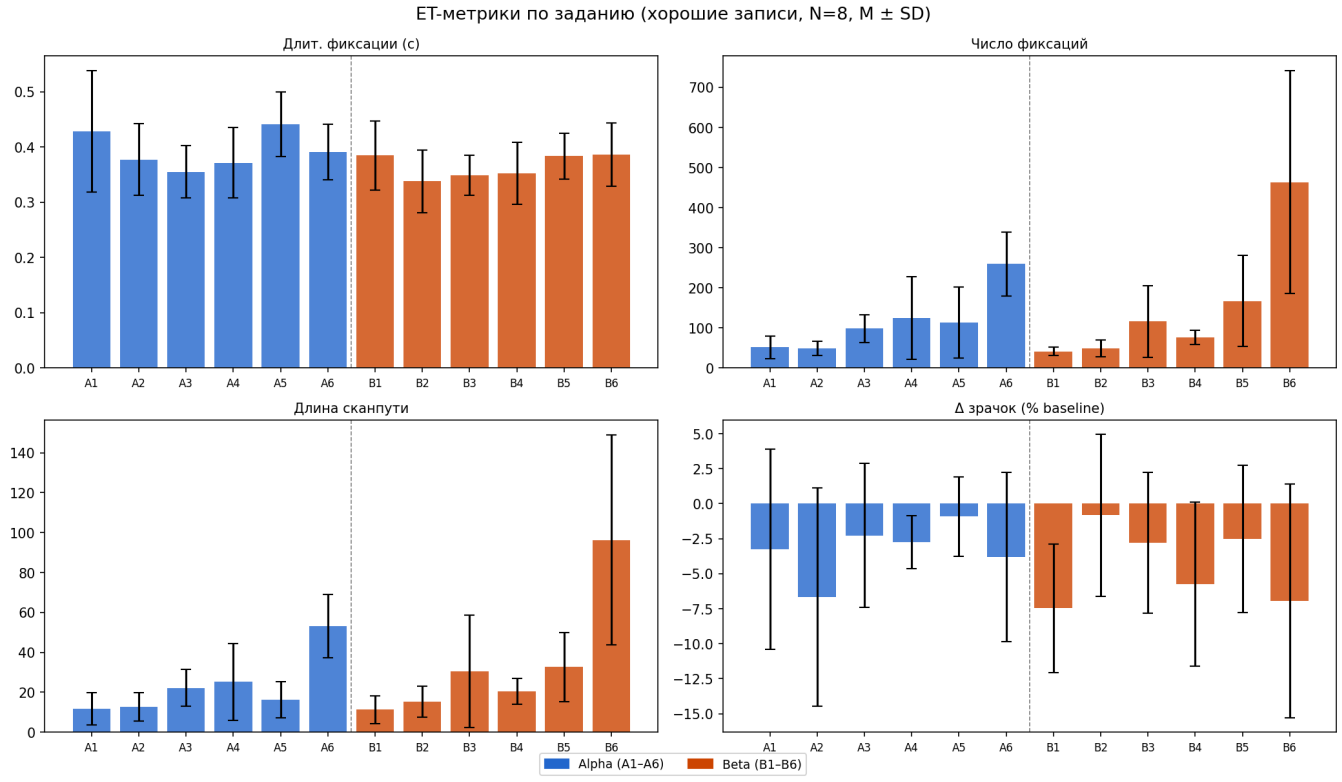


Figure 16: Eight key metrics (`pupil_pct_change_avg`, `pupil_z_avg`, `fixation_count`, `fixation_duration_mean`, `scanpath_length`, `pupil_speed_mean`, `blink_count`, `pupil_baseline_avg`) for each of the 12 tasks (A1–A6 in the alpha dataset, B1–B6 in the beta dataset), the mean across participants ± SEM.

A.5 Individual pupil trajectories

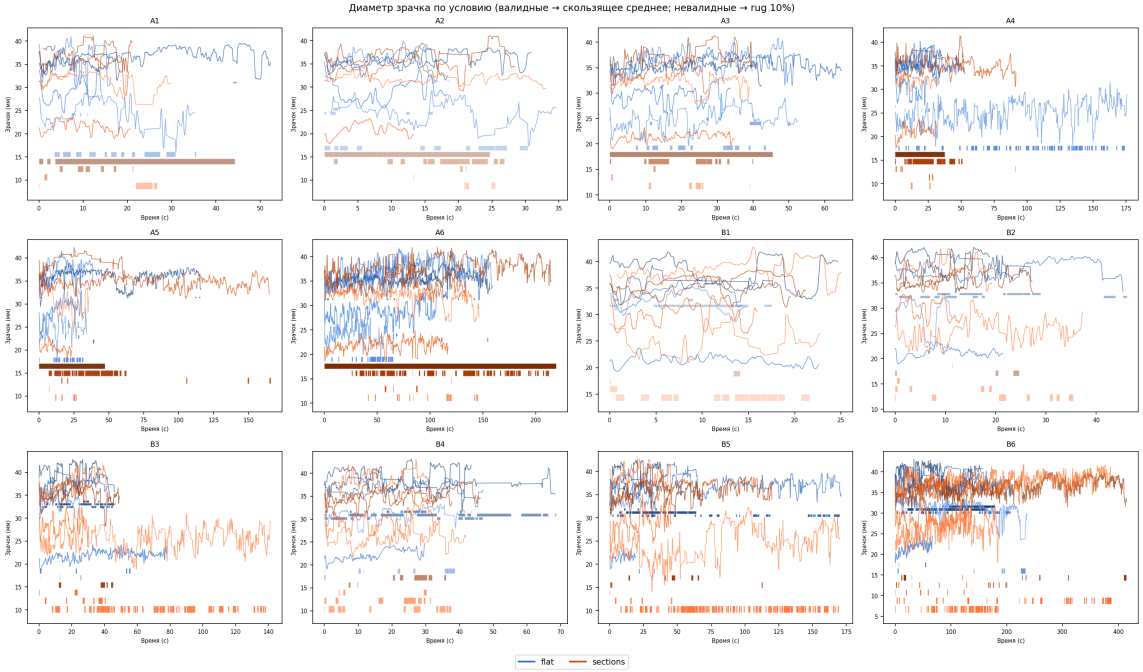


Figure 17: Individual trajectories of pupil diameter across all 12 tasks, all 10 participants (`df_all`), split by condition.

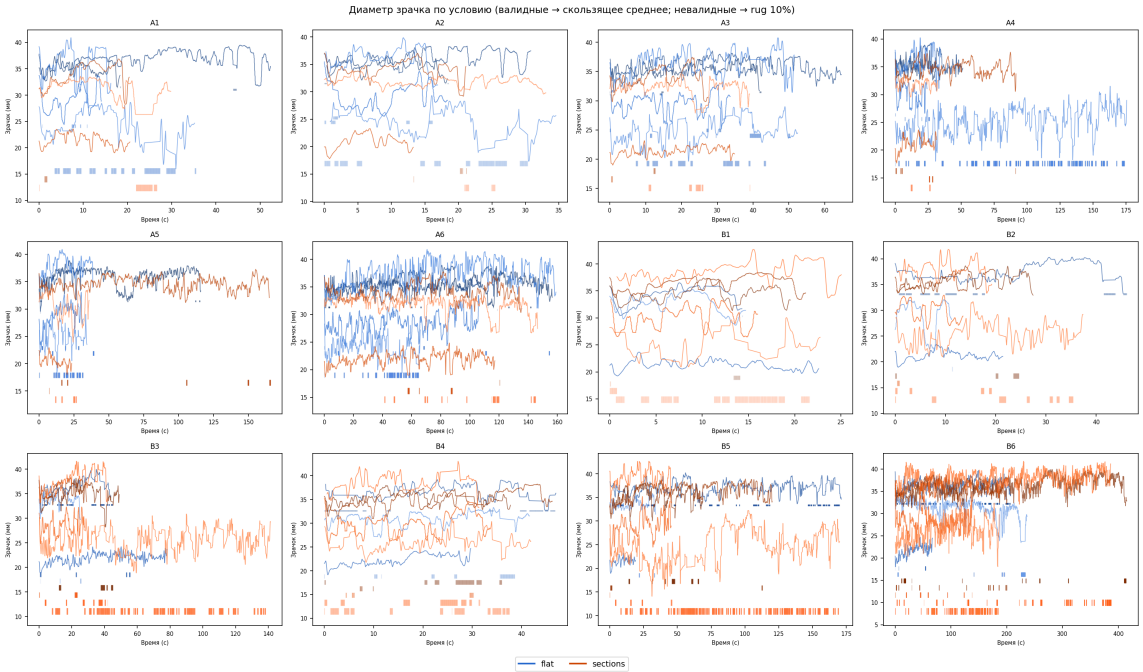


Figure 18: Individual trajectories of pupil diameter for the `df_good` subsample (N=8, proportion of invalid samples $\leq 30\%$), split by condition.

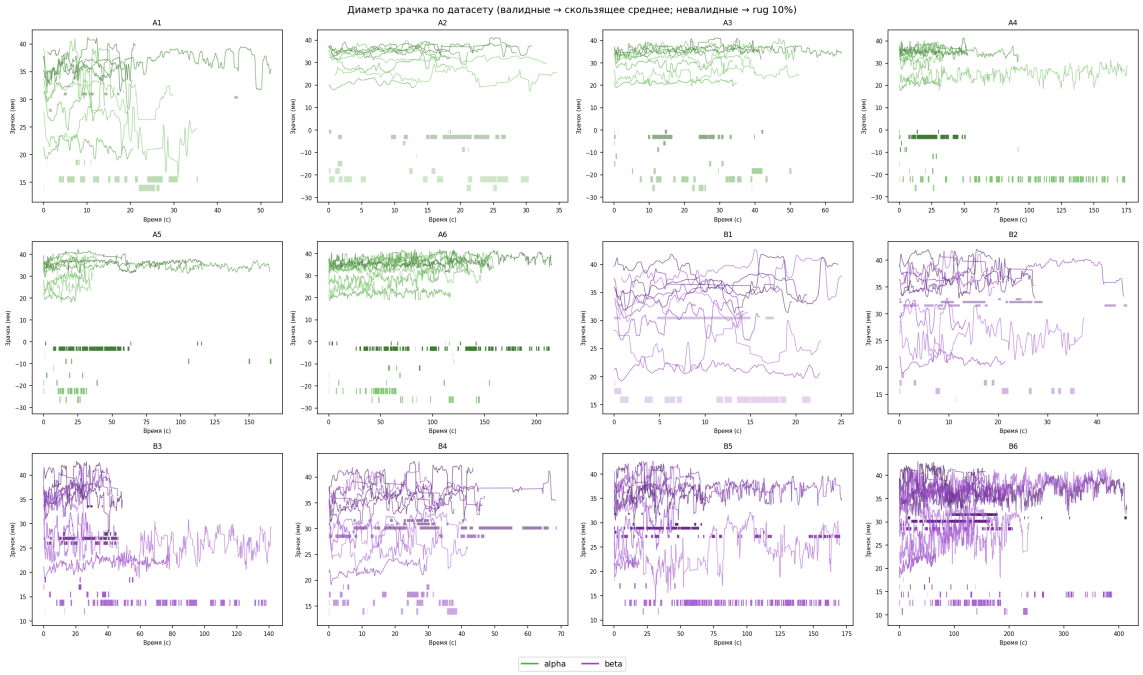


Figure 19: Individual trajectories of pupil diameter split by dataset (alpha / beta).