Investigating Feasibility of Webcam-Based Eye-Tracking as an Alternative Input Modality for Micro-Task Work

Davey Struijk
Supervisor(s): Garrett Allen
EEMCS, Delft University of Technology, The Netherlands
22-6-2022

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Abstract

People who perform work on micro-task crowdsourcing platforms, often do so using a mouse and/or keyboard for many hours at a time, while alternative modes of input could potentially provide a better experience. This research investigates the feasibility of using webcam-based eye-tracking in a micro-task work environment. We accomplish this by setting up a user study (n = 20), where participants are asked to perform a series of image classification tasks using either a mouse or just their eyes. Overall, results show that participants using a webcam are generally able to complete the tasks adequately. However, they perform somewhat slower and less accurate, and are less content with their overall experience. Based on our results, we suggest that there are still limitations to overcome when applying webcam-based eye-tracking to micro-tasks.

1 Introduction

Micro-task platforms, such as Amazon MTurk or Prolific, are services that provide a marketplace to crowdsource short tasks. On these platforms, workers receive a small monetary reward for performing simple tasks, such as finding information, validating content, performing interpretation, or filling out surveys. [1]

Most micro-tasks are repetitive in nature and are typically performed using a mouse or touch screen for many hours at a time, possibly leading to occupational health risks in micro-task workers [2]. In addition, the lack of task variety and significance using conventional input methods could lead to workers becoming bored or disengaged [3]. As a consequence, workers could benefit from using alternative forms of input for performing their work.

1.1 Webcam-Based Eye-Tracking Technology

The use of eye-tracking for human-computer interaction can be described as a more "direct" mode of input, where the input and display surface are unified [4]. It requires very little movement for the user, and doesn't have the occlusion problems touch-based input has. It is also an input method with one of the fastest reaction times possible – at the speed of gaze. A downside, however, is that there is no natural way of emulating button-presses, which is usually required when interacting with user interfaces.

While eye-tracking could previously only be performed using additional sensors or devices, advancements in webcam-based eye-tracking technology have made it possible to perform gaze tracking on commodity hardware such as mobile phones and laptops [5; 6]. Since most people already own a laptop or smartphone with a built-in webcam, micro-task workers can make use of this technology for no additional costs.

The accuracy of webcam-based eye-tracking algorithms is a well-researched topic, and may depend on factors like lighting, calibration, and head movement [7]. State-of-the-art javascript eye-tracking models generally produce an error of about 200px (17% screen size, 4.16°), which is accurate enough for completing proof-of-concept tasks involving fixation, pursuit and free-viewing. However, it is not clear whether the technology is feasible for use in common micro-task work environments.

1.2 Research Goal

We would like to assess whether webcam-based eye-tracking is a feasible input modality for micro-task work. To get a full picture of the viability of such an input, this study poses the following questions:

- How does this new input method fare in task accuracy & performance when compared to conventional input methods?
- What impact does using webcam-based eye-tracking have on the subjective experience of the worker?
- What effect does calibration time have on the performance of the worker?

1.3 Paper Structure

In Section 2, we talk about prior research related to the ideas in this paper. Section 3 describes the step-by-step process of the user study that was performed. Results of this study are then laid out in section 4, where we analyze the data and denote variables of interest. Section 5 discusses the results and reflects on the process. Finally, section 6 provides a conclusion and recommendations for further research.

2 Related Work

There has been quite some research on webcam-based eye-tracking in the past decade, and several novel libraries and datasets have been made available since [6]. Because most applications of this technology are real-time, it is important that the prediction can be updated many times per second, which limits the use of time-consuming models. Also, due to privacy concerns, webcam footage usually has to stay on the user's device, requiring the processing of video to be done on commodity hardware [9]. These limitations make webcam-based eye-tracking an interesting field, with various libraries out there using different methods to achieve similar results.

Project	Since	Platform	Algorithm	Open-Source?
OpenGazer	2010	Linux	Gaussian	✓
optimeyes	2014	Desktop	Geometric	✓
GazePointer [5]	2014	Windows	Geometric	
xLabs Gaze	2015	Browser	Unknown	
WebGazer.js [10]	2016	Browser	RR	✓
iTracker [6]	2016	iOS	CNN	✓

Table 1: Notable webcam-based eye-tracking libraries.

To give an overview of the current landscape, table 1 lists some notable webcam-based eye-tracking libraries from the past decade. Most implementations use either a purely geometric or a machine learning approach to emulate a cursor on

users' screens. Geometric methods rely on mathematically projecting the user's gaze vector to the screen, while a machine learning approach does this implicitly by using images of the user's eyes as input. Recent research suggests that a hybrid approach (geometric + ML) can provide better results [9]. However, a usable implementation of this is not publicly available.

As for current limitations, all of the projects mention the requirement of a well-lit face, and suggest that head movement should be limited to maintain accuracy. A correct distance to the camera is also often mentioned, as well as hardware that is not too slow. Reported accuracy can vary widely based on these factors, and may limit users from working in certain environments.

Of all the webcam-based eye-tracking implementations, WebGazer.js [10] was the most suitable for our experiment. Having participants install an application just for this experiment would be too time-consuming, and WebGazer.js is the only realistic option for embedding webcam-based eye-tracking in a browser environment. It is also well-maintained, and the most straightforward to implement out of the libraries previously mentioned. The library uses a ridge regression model, mapping pixels from the detected eyes to locations on the screen, but this tracking model can be substituted with other variants as well.

3 Methodology

We developed a web-application where workers complete a series of tasks representative of common micro-task work, using either webcam-based eye-tracking or conventional input. We then published this application as a survey on Prolific, a micro-task crowdsourcing platform commonly used for academic research. Participants were randomly (50/50) assigned to the "mouse" or "webcam" group, and performed the experiment using their assigned input modality. They were awarded £15.25/hr on average for completing the experiment, which is above market rate on this platform.

To be able to answer all of our research questions, we measured participants' task accuracy & speed, but also concluded the experiment with a survey to ask about their subjective experience.

3.1 Reproducibility

By using open-source technologies and documenting our specific implementation details, we try to make the study as reproducible as possible. Related files, including the user study implementation in its entirety, are available via GitHub (https://github.com/daveystruijk/cse3000).

3.2 Privacy & Consent Notice

Before starting the experiment, we first need to inform the participant that they are being involved in a TU Delft study, and that their data could be published in a paper in an anonymized form. Also, because this is a study that involves accessing the user's webcam, we need to make it very clear that any webcam footage or other personally identifiable information will not leave the user's computer. This is done by having the user explicitly agree with these statements before beginning the experiment.

3.3 Calibration (Webcam Only)

The eye-tracking library used in this experiment requires a calibration step, which is done by having the user click on parts of the web page while looking directly at the cursor. In our implementation, we enforced a minimum of 10 calibration datapoints, and then let the user decide when calibration is good enough to continue. These datapoints are recorded as well, to assess whether differences in calibration behavior have an influence on other variables.

3.4 Micro-Task Experiment

To mimic a common micro-task workflow, we implemented a user interface that could be used with either input modality. Using the participant's assigned input method, they perform 10 instances of such a task. Considering the exploratory nature of this experiment, we decided upon a set of binary image classification tasks. Participants were asked to view the image, and select whether it represents a cat or a bird.

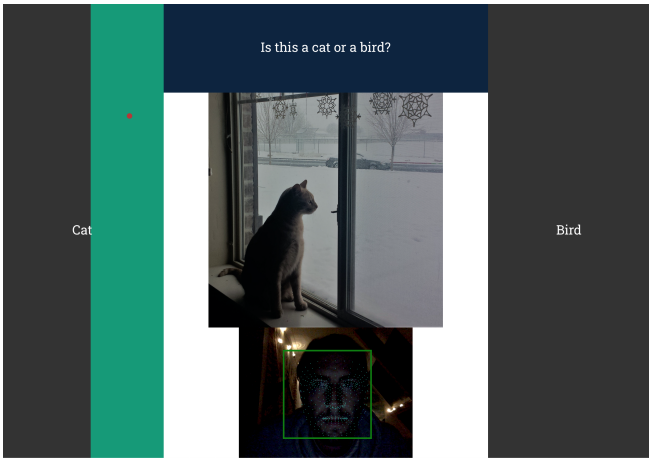

Figure 1: A screenshot of the image classification workflow, being controlled using eye-tracking.

We implemented large button areas that the participant could select by holding their eyes on it for 1 second. These buttons span 25% of the screen on both sides, to accommodate for a potentially large margin of error when using eye-tracking. When performing the tasks using a mouse, these button areas are simply clickable as usual.

To measure how well participants perform using our implementation of webcam-based eye-tracking, we track task accuracy and speed. Task accuracy, defined by the amount of errors in the image classification tasks, shows the reliability of data obtained using each input method. Task speed, the amount of time it takes to respond to each question, reveals the difference in timing between the two input methods. Both measurements assist in determining whether participants are able to complete the tasks adequately.

3.5 Survey

To get a better understanding of the participant's overall experience during the experiment, we present them with a survey of 15 questions after the tasks are performed. These questions are categorized in three groups:

- Demographics: Age (group), gender, and experience with micro-task platforms.
- Engagement: We include 6 questions from the Short Form User Engagement Scale [11], namely PU-S.1..3 and RW-S.1..3. Answers are given on a 5-point Likert scale.
- Workload: We assess the 6 types of perceived workload as described in the NASA Task Load Index [12]. Answers for these are given on a similar 5-point scale (low/high or poor/good).

4 Results

20 participants (12 male, 8 female), from a diverse age group, completed the tasks and subsequent survey. Most (70%) described their prior experience with micro-tasks as mid-level. The set of image classification tasks took 30 seconds on average to complete, and the experiment in its entirety lasted for about 2-3 minutes. Due to an error in the experiment setup, some of the measurements for the first answers were timed under 0.01ms, which is not realistically possible. For this reason, each participant's first answer was removed from the dataset, leading to a total of 180 micro-task observations (90 mouse, 90 webcam).

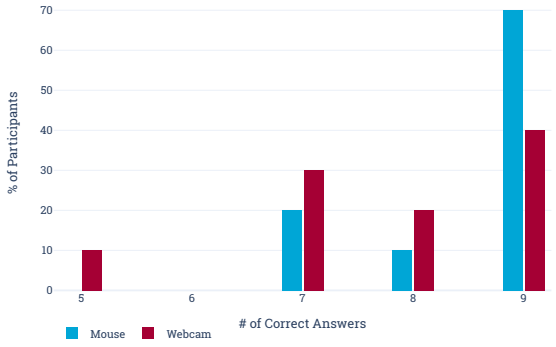

Figure 2: Amount of correct answers (out of 9) in image classification tasks, by input modality.

In figure 2, we compare the amount of mistakes made between the groups. It is clear that mouse users outperform webcam users in accuracy, with the webcam group making more than twice as many mistakes on average (mouse: 5.5%, webcam: 13.3%).

We also noticed that the amount of calibration data participants were willing to generate consisted of more calibration points than minimally required (+20%). This average excludes one outlier, who generated an extreme amount of calibration points (75). Interestingly, this participant also made the most amount of mistakes in the image classification tasks.

As for timing, figure 3 shows that webcam users were slower on average (mouse: 2.0s, webcam: 3.0s). Considering the added second of selection delay for the webcam

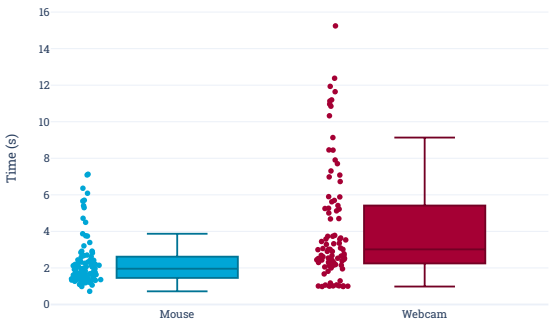

Figure 3: Time it takes to complete one image classification task.

group, these results are reasonable. However, 11 of the 90 measurements for the webcam group are extremely close to 1 second, meaning they may have been unable to move their cursor away from the button in time, leading to an accidental click. These accidental clicks may partially account for an increase in mistakes and frustration for the webcam group.

4.1 User Feedback

Results from the survey indicate that participants were generally less content with the experience when performing tasks with our proposed webcam-based eye-tracking solution.

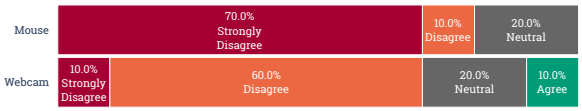

Figure 4: "I felt frustrated while using this application."

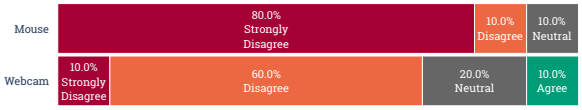

Figure 5: "I found this application confusing to use."

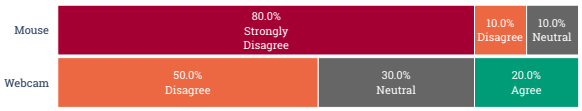

Figure 6: "Using this application was taxing."

In figure 4-6, we asked participants questions related to perceived ease of use. While the responses for the mouse group reveal no negative experience at all, there were a few webcam users who found the application difficult to use. Not all participants were worried about usability, with an average of 1.4 (strongly disagree) for mouse users, and 2.6 (neutral) for the webcam group. However, the difference between the

groups is noticeable, and shows that participants had a harder time working with the eye-tracking implementation.

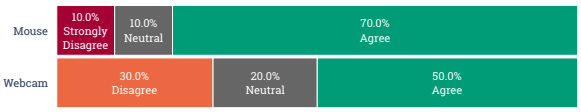

Figure 7: ”Using this application was worthwhile.”

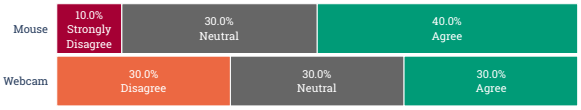

Figure 8: ”My experience was rewarding.”

Figure 9: ”I felt interested in this experience.”

When asked about how rewarding or interesting the experience was (figure 7-9), the mouse group responded with a 3.8 on average, as opposed to a 3.5 from the webcam group. These results are both above neutral on average, which is an interesting metric on its own.

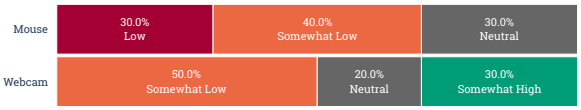

Figure 10: ”How much mental and perceptual activity was required (e.g. thinking, deciding, calculating, remembering, looking, searching, etc.)? Was the task easy or demanding, simple or complex, exacting or forgiving?”

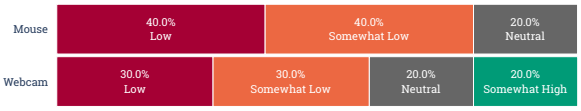

Figure 11: ”How much physical activity was required (e.g. pushing, pulling, turning, controlling, activating, etc.)? Was the task easy or demanding, slow or brisk, slack or strenous, restful or laborious?”

Figure 12: ”How much time pressure did you feel due to the rate at which the tasks occured? Was the pace slow and leisurly or rapid and frantic?”

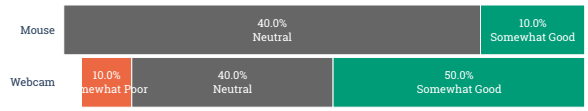

Figure 13: ”How successful do you think you were in accomplishing the goals of this task? How satisfied were you with your performance in accomplishing these goals?”

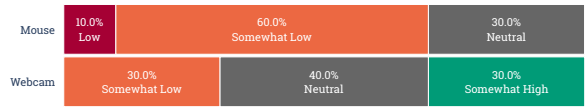

Figure 14: ”How hard did you have to work (mentally and physically) to accomplish your level of performance?”

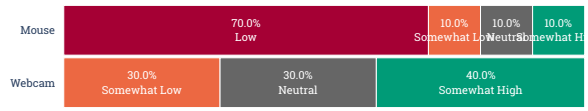

Figure 15: ”How insecure, discouraged, irritated, stressed and annoyed vs. secure, gratified, content, relaxed, and complacent did you feel during the task?”

Further questions about perceived workload (figure 10-15) confirm what we observed in some earlier answers, and data clearly shows that the webcam group generally perceived the experiment as more taxing. In most cases, the difference is not very pronounced. However, the most contrasting difference can be seen in participants' frustration level (figure 15), which also has a negative correlation (-0.42) with the amount of correct answers. This correlation might hint at participants being annoyed because they wanted to perform well in the experiment, but were unable to. Figure 13 shows another interesting effect where, even though the webcam group performed worse overall, they perceived themselves as slightly more successful (mouse: 3.4, webcam: 4.1).

5 Discussion

Overall, the data shows that webcam-based eye-tracking is generally usable in micro-task work, as most participants were able to complete the tasks adequately. However, they perform somewhat slower and less accurate, and are less content with their overall experience. This can be attributed to several factors, some related to the experimental setup, and some related to current technological limitations.

Because the first answer of the experiment had a bug for webcam users, there is a possible negative effect on the overall experience of the participant, and this may have resulted in less favorable user feedback. Furthermore, some participants seemed to have made a few misclicks because of the 1-second button delay being too fast. The second issue could have been prevented by requiring the user to re-focus on the center image before being able to select an option again.

Currently available libraries for webcam-based eye-tracking also carry some technological limitations which may have had an impact on the results. During experimentation with WebGazer.js, we noticed the model accuracy decreasing

as time passed. This issue was described in similar research as well, and is mainly attributed to head movement during the experiment. Mitigations include frequent recalibration, or a head rest [8]. However, both of these solutions would introduce additional hindrances for the user. Also, in our data, additional calibration points did not necessarily improve task performance.

Some other options exist for improving performance. For example, projects like TurkerGaze retrain the model after the experiment is finished for better accuracy [8]. Not all tasks are suitable for post-processing though, e.g. if the task requires the feedback to be real-time, or video footage must be kept on the user's device. Post-processing can also be used for another purpose: studies have shown that even with many errors in binary or categorical labeling tasks, it is possible to ultimately gain faster results by focusing on speed instead of accuracy [13].

All-in-all, webcam-based eye-tracking can currently prove useful in niche task types when set up correctly. However, the input modality might see more widespread usage if reliability was less of an issue.

6 Conclusion and Future Work

In this paper, we test the feasibility of webcam-based eye-tracking by performing a user study. We compare data between a group of mouse and webcam users, and measure differences in speed, accuracy, and reported experience. Even though the webcam group showed slightly worse results, users from both groups were able to complete most tasks adequately. We think our results show that webcam-based eye-tracking is a promising input modality for performing micro-task work, but the current implementations have some flaws which made the technology frustrating to use for participants.

Further research should focus on improving existing implementations, and making them more resilient to changes in head posture and other factors that may limit performance. Combined with advancements in machine learning research and increasing hardware capabilities, webcam-based eye-tracking may become reliable enough to develop into a widespread input modality in the future.

References

[1] Gadiraju, Ujwal & Kawase, Ricardo & Dietze, Stefan. (2014). A taxonomy of microtasks on the web. HT 2014 - Proceedings of the 25th ACM Conference on Hypertext and Social Media. 10.1145/2631775.2631819.

[2] Bajwa, U., Gastaldo, D., Di Ruggiero, E. et al. (2018) The health of workers in the global gig economy. Global Health 14, 124. 10.1186/s12992-018-0444-8

[3] Deng, Xuefei & Joshi, K. D.. (2016). Why Individuals Participate in Micro-task Crowdsourcing Work Environment: Revealing Crowdworkers' Perceptions. Journal of the Association for Information Systems. 17. 648-673. 10.17705/1jais.00441.

[4] Hinckley, Ken. (2002). Input Technologies and Techniques. 10.1201/b10368-12.

[5] Ghani, Muhammad Usman & Chaudhry, Sarah & Sohail, Maryam & Geelani, Muhammad. (2014). GazePointer: A Real Time Mouse Pointer Control Implementation Based on Eye Gaze Tracking. Journal of Multimedia Processing and Technologies. 5. 64-75.

[6] Krafka, K., Khosla, A., Kellnhofer, P., Kannan, H., Bhandarkar, S., Matusik, W., & Torralba, A. (2016). Eye tracking for everyone. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2176-2184).

[7] Semmelmann, K., & Weigelt, S. (2018). Online webcam-based eye tracking in cognitive science: A first look. Behavior Research Methods, 50(2), 451-465.

[8] Xu, P., Ehinger, K. A., Zhang, Y., Finkelstein, A., Kulkarni, S. R., & Xiao, J. (2015). TurkerGaze: Crowdsourcing Saliency with Webcam based Eye Tracking. arXiv. 10.48550/ARXIV.1504.06755

[9] Gudi, A., Li, X., van Gemert, J. (2020). Efficiency in Real-Time Webcam Gaze Tracking. In: Bartoli, A., Fusiello, A. (eds) Computer Vision – ECCV 2020 Workshops. ECCV 2020. Lecture Notes in Computer Science(), vol 12535. Springer, Cham.

[10] Papoutsaki, A., Sangkloy, P., Laskey, J., Daskalova, N., Huang, J., & Hays, J. (2016). WebGazer: Scalable Webcam Eye Tracking Using User Interactions. In Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI) (pp. 3839-3845)

[11] Heather L. O'Brien, Paul Cairns, Mark Hall. (2018) A practical approach to measuring user engagement with the refined user engagement scale (UES) and new UES short form. International Journal of Human-Computer Studies, Volume 112, (pp. 28-39)

[12] Hart, S. G. (1986). NASA task load index (TLX).

[13] Ranjay A. Krishna, Kenji Hata, Stephanie Chen, Joshua Kravitz, David A. Shamma, Li Fei-Fei, and Michael S. Bernstein (2016). Embracing Error to Enable Rapid Crowdsourcing. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16). Association for Computing Machinery, New York, NY, USA, 3167–3179