Usability of gaze-transfer in collaborative programming: How and when could it work, and some implications for research agenda

Roman Bednarik
AuthorCo, Inc.
123 Author Ave.
Authortown, PA 54321 USA
author1@anotherco.com

Andrey Shipilov
AuthorCo, Inc.
123 Author Ave.
Authortown, PA 54321 USA
author5@anotherco.com

DUET 2012, Feb 5–10, 2012, USA.
ACM xxx-x-xxxx-xxxx-x/xx/xx.

Abstract
Dual eye-tracking is becoming popular in CSCW community as a method to investigate and support collaboration and interaction. Based on our preliminary study and subjective feedback of the participants, we reflect on the usability and possibilities of synchronous gaze transfer during collaborative programming. We focus on a discussion of when gaze-cursor is beneficial and what forms it could take, and on some challenges and lessons learned when implementing gaze transfer. Our results inform about the research and design of future gaze-contingent collaborative interfaces.

Keywords
gaze transfer, programming, collaboration

ACM Classification Keywords
H.5.m [**Information interfaces and presentation (e.g., HCI)**]: Miscellaneous.

General Terms
Human Factors, Experimentation

Introduction
Eye-tracking has become a standard tool in many domains as a tool to understand the visual attention of users. Recently, eye-tracking research has witnessed a leap from

the single-user scenarios to multiple user, collaborative domains. In this research we are focusing on the role of gaze in collaborative programming, in particular on synchronous tasks such as pair programming, collaborative debugging, and code co-comprehension.

As in other problem-solving tasks, gaze has an important role in solo programming [1]. When software development happens in a collaborative way, however, the gaze and visual attention are also employed to maintain a good quality of communication between the peers. An effective collaboration requires efforts towards planning joint action and interpreting the peers' action, to name few.

In distributed programming teams, joint attention, by the means of gaze direction, and as a way of establishing and maintaining the common ground for joint action, has been till recently virtually impossible to achieve. Using a tool we developed to allow multiple-participant gaze-transfer between networked workstations [3], we conduct studies that investigate not only where the participants look at during collaboration, but we evaluate what are the effects of displaying the gaze of the programmers on the quality of collaboration [2].

Our results indicate that real-time gaze transfer[1] has some measurable effects on the behavior of the peer who receives it. These include the increased mean fixation duration and a more balanced distribution of the total fixation time between a code and a graphical display [2].

In other studies, real-time gaze-cursor, a replay of other's gaze, or a modification of the stimuli based on other's gaze have been found to improve learning [7, 4, 5].

[1]In this paper we interchangeably refer to gaze transfer as gaze-cursor or gaze-mark, due to the particular technique of gaze visualization employed.

Despite the growing evidence suggesting that gaze-display and related methods do have important effects on the collaboration processes, there are numerous questions that remain open. We suggest that the challenges we should ask next are then,

- When gaze-cursor (gaze-transfer, gaze-display) works
- How should gaze of a collaborator be displayed

After an overview of the subjective feedback we collected in a collaborative programming study, in this paper we discuss these questions in a more detail.

Gaze transfer in programming: subjective preferences of participants

We conducted a synchronous programming lecture study, in which gaze of an expert programmer was displayed on the screen of a novice programmer. The task of the expert was to explain functioning of an algorithm. The task of the novice was to try to understand the way the algorithm works and then apply it on an unseen data-structure. The two programmers worked in a shared desktop environment, see Figure 1; for more details about the study see [2]. The environment presented the problem using two adjacent representations: the source code of an algorithm and a graphical visualization of input data.

The study has been designed as a within-subject experiment, in which all participants experienced both with-gaze and without-gaze conditions. There was no transfer of the gaze from the novice to the expert, as the expert was proceeding according to a script. A preliminary analysis of the effects of the gaze transfer on performance showed no significant improvements [2].

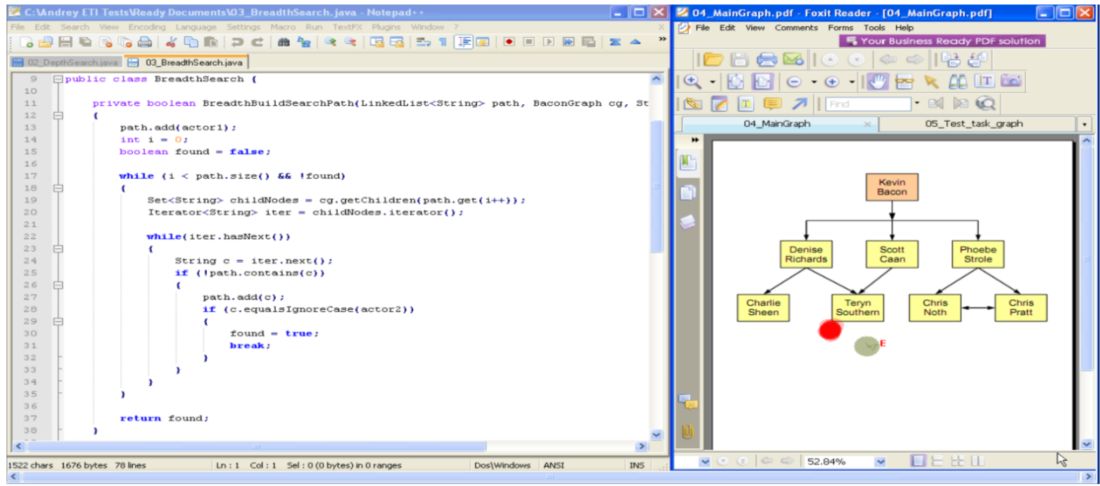

Figure 1: A screenshot from the experiment. Expert gaze in grey color (shown realtime) is denoted by "E", novice's gaze (not shown during experiment) is reconstructed from recorded data.

In the gaze transfer condition, the size of the gaze-mark was approximately 25 pixels in radius, resulting in about 0.65 degrees of visual angle at a viewing distance of 60cm. The rendering of the gaze-mark was based on fixation data. This means that every time the expert client has identified a fixation, it was transferred to the server side and immediately rendered on the novice's client screen.

Using a primarily close-ended questionnaire form we collected the following data:

- 1. Did gaze-mark help to understand explanation?
- 2. Was gaze-tracking mark distracting?
- 3. What was most distracting?
- 4. What do you think about the size of the gaze-mark?
- 5. What do you think about the filling of the mark?
- 6. What do you think about the shape of the mark?
- 7. What do you think about tracking the movements of the gaze-mark?
- 8. What would you change in the user interface, concerning presenting eye-tracking information to make it more usable for you?

Altogether, twelve novice participants completed the questionnaire, six were male and six female. While typically participants responded by one answer only, in some questions related to the appearance few of the participants selected more than one option.

Results
Regarding the first item, only two participants considered the gaze-cursor as not helping in understanding the explanation of the program code[2]. Five participants replied with 'yes', while three selected 'sometimes' and one with more specific answer that the gaze-cursor was helping when attending the graphical representation.

The second and third questions were related to possible distractions by the gaze-cursor. Seven (58%) participants replied with 'No', three with 'Sometimes', and remaining two with 'Yes'. Of the five that replied No or Sometimes, three participants responded that most distracting about the gaze-cursor was its trembling.

The fourth item, related to the size of the gaze-mark, prompted seven participants to reply with 'It should be smaller'. One of the seven replies contained a note 'It was good on the graph'. Another four subjects responded with 'It was good'. Finally, two participants suggested that the size of the gaze-cursor should be adjustable depending on the actual task.

In the fifth question, we were interested about the appearance of the gaze-cursor, specifically about the saliency, opacity and pattern of filling. The distribution of the responses contained seven responses with 'It was good', two replies with 'Depending on the background media'. One participant would prefer more transparency, and one less transparency. Finally, one response suggested changing the saliency of the gaze-cursor (by the means of opacity) depending on the distance to the current point of regard.

The last item related to the appearance of the gaze-cursor concerned the shape. Eight (67%) participants had no complains about the shape. Two participants suggested the gaze-mark should take a shape of a cursor and thus be more accurate over the text. Another two participants wished for an 'As simple as possible ' solution, specifying that for textual representations whole lines of source code should be highlighted, and that no animations or other special effects should be used. Although one option suggested an animated, more visible form of a bouncing ball, none of the participants selected that option.

With presenting the two last questions, we aimed at collecting more ideas for further development of the visualization of the gaze transfer. In total, nine (75%) of participants specifically suggested that no trail should be added to the gaze (eighth question), while on the other hand three saw some benefits such as in improving concentration. Three participant suggested smoothing of the movements of the gaze-mark or required that trembling should be reduced. Other individual suggestions included customizable color, bigger font on the annotation, and using red color for better visibility.

Discussion

When gaze-transfer works
In what kind of situations does gaze transfer work? The responses we collected give us a good evidence that gaze-transfer should be task-dependent and should be adaptive on the progress of the process. Not only our participants suggested that seeing other person's gaze is at times disturbing, they also indicated that certain stages of the collaborative code explanation call for varying

[2]One of the participants was not paying attention to the explanations coming from the expert. We are including the outlier here for completeness.

degree of intrusiveness.

Despite striving for technical perfection when processing, transferring and rendering the gaze data, the properties of human visual system make it hard to visualize the real time gaze deployment in a meaningful and non-distracting way. Trembling of the gaze-cursor and swift saccadic movements of the sender require extra efforts on the side of the receiver.

How should gaze of a collaborator be displayed
Some of the participants expressed that the gaze-cursor could be smaller. We believe this is due to two related reasons. First, it is understandable that if a bigger opaque shape stays in the current point of regard, it is difficult to extract information that is being covered. This is related to the second aspect, namely, the occlusion is more serious on small-size stimulus such as text. When a peer explains some details about a certain piece of source code, it is distracting for the receiver to see the part in question covered. The same was not true when talking about the graphical visualization, perhaps due to the larger geometrical shapes involved.

While a more subtle form of gaze-cursor could be used, we believe this would not contribute to easier discovery of the peer's gaze location. A smaller size or a more translucent form could simply be harder to spot. We suggest that a possible solution could be to adaptively adjust the opacity (and saliency of the enhanced presentation) based on the actual distance between the gaze of the sender and the gaze of the receiver.

Apart from the actual form of presentation of the gaze-mark, we suggest that some other, media-sensitive forms of cuing the receiver's visual attention based on the gaze transfer could be implemented. For instance, in the integrated development environments, a certain block of the source code of the program could have only been highlighted, depending on the position of the gaze cursor and task being carried out. Similarly, graphical representations could be enhanced by the knowledge about the peer's point of regard, instead of the currently used circular point overlaid over them.

Future research
One of the primary challenges of the research into effects of gaze-transfer is to delineate when precisely gaze-display works. To help answering this problem, we suggest future studies can make use of explicit feedback from the receivers. As an extension, a concept of 'gaze on demand' should be investigated. Similarly as in the studies of radiology and gaze replay for feedforward training [4], the effects of various levels of fidelity of the displayed gaze on performance should be investigated.

We envision future peer-gaze-contingent systems. Such kind of intelligent interfaces would be built on a detailed knowledge of the task at hand, type of media, current point of regard, and collaborator skills; the list of factors and inputs, on which a model would be built and maintained, is longer.

Other methodological considerations
In the following, we discuss the challenges we met when connecting two or more eye-trackers and used them for gaze-cursor display. While we are aware our experience is hard to generalize, we believe that some problems may be more common and thus the discussion can improve the state-of-art of the dual eye-tracking methodology.

Synchronization of data streams, processing lag, and precision

If in single-user situations the eye-tracking researcher's task was to align data from an eye-tracker with audio, video and other protocols, the situation becomes more complex in dual eye-tracking and in distant, networked collaborations. The lag of the network, transport of data, processing times including fixation identification on one hand, and the swiftness of the gaze on the other hand, all contribute to sometimes experienced instability and flicker of the gaze-cursor.

To decrease the effects we implemented a mechanism that limits the number of messages sent from the client to the server application. As a result, the server software collects a buffer of previously recorded gaze positions and outputs only a representation of it, such as a weighted average of previously seen fixations. This feature will also be useful when connecting low-cost eye-trackers, whose precision or sample rate are currently lower than those of commercial devices.

Connecting different eye trackers

Another common problem concerns connecting devices from different vendors, or connecting low-cost self-made eye-trackers. Our approach provides a platform independent solution, by employing a middleware layer for independent gaze-date collection and processing, the ETU driver [6]. The system supports numerous devices and models, see [6] for an up-to-date list.

Conclusions

We presented an evaluation of gaze-transfer in the domain of collaborative programming. Our representation of the gaze-cursor was a fixation-associated point overlapped over the user interface. Taking a usability perspective, we collected feedback from the participants using a questionnaire.

The analysis of the responses shows that gaze-cursor should be media dependent and that instead of using a pointer-style representation, some form of highlighting may be preferred. In addition, our participants disliked the trembling and flicker of the gaze-cursor, especially when the sender performed a series of larger saccade. Such event was then hard to follow and our subjects reported the feeling of being lost.

To achieve the proposed gaze-contingency, task, expertise and media understanding should be in the center of the future research.

References

[1] R. Bednarik. Expertise-dependent visual attention strategies develop over time during debugging with multiple code representations. *International Journal of Human-Computer Studies*, 70(2):143 – 155, 2012.

[2] R. Bednarik and A. Shipilov. Gaze cursor during distant collaborative programming: A preliminary analysis. In *Proceedings of the DUET 2011: Dual Eye Tracking in CSCW, 2011*, 2011.

[3] R. Bednarik, A. Shipilov, and S. Pietinen. Bidirectional gaze in remote computer mediated collaboration: setup and initial results from pair-programming. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, CSCW '11, pages 597–600, New York, NY, USA, 2011. ACM.

[4] D. Litchfield, L. Ball, T. Donovan, D. Manning, and T. Crawford. Viewing another person's eye movements improves identification of pulmonary nodules in chest x-ray inspection. *Journal of Experimental Psychology: Applied*, 16(3):251, 2010.

[5] S. Sadasivan, J. Greenstein, A. Gramopadhye, and A. Duchowski. Use of eye movements as feedforward training for a synthetic aircraft inspection task. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 141–149. ACM, 2005.

[6] O. Spakov. *Eye-Tracking Universal Driver (ETUDriver), COM interface description, Version 1.25*. University of Tampere, 2006.

[7] R. Stein and S. Brennan. Another person's eye gaze as a cue in solving programming problems. In *ICMI*, pages 9–15, 2004.