ObSys – a Tool for Visualizing Usability Evaluation Patterns with Mousemaps

Michael Gellner

University of Rostock
Department of Computer Science
Software Engineering Group
Albert-Einstein-Str. 21
18051 Rostock, Germany

Peter Forbrig

University of Rostock
Department of Computer Science
Software Engineering Group
Albert-Einstein-Str. 21
18051 Rostock, Germany

Abstract

This paper presents *mousemaps,* multidimensional visualizations of user interactions with software systems. The source for that visualizations are data that are recorded with the event recorder *ObSys*. On the one hand mousemaps are proper for enhancing the effectiveness of video analyses, on the other hand it is possible to empower the efficiency of video analyses by focusing automatically critical situations. Furthermore mousemaps can substitute video analyses because the visualization offers most of the interaction information on a two-dimensional view without the need to watch video material. The low level data recorded by *ObSys* allow the recognition of »mouse gestures« that are characteristically for certain usability problems. Automating the analyses of usability testing sessions by applying usability evaluation patterns without the need of modeling information as required for other automating tools becomes possible.

1 Introduction – our work and what was there before

Usability testing with testing persons is of prime importance to get information about how users really can work with a system. Since it is impossible to analyse sessions on the fly or in between testing sessions are recorded to be analysed afterwards. For recording such a session there are different technologies, which offer features but disadvantages otherwise.

Often applied is video recording, since it allows to save the actions of the user, the behaviour of the system and especially the mimic and gesture information. But video recording is not efficient: For every hour of material one has to calculate around six to eight hours watching for recognizing problems. Recording from different perspectives multiplies the time of the sessions with the time of the analysing. As a consequence often taped material stays unevaluated (Mayhew, 1999) (Preece, 1994). There are further factors that make video recording inefficient like high hardware requirements, the necessity to hide the intimidating technology and personal needs. Although video recordings contain effective information for an human evaluator it is nearly impossible to analyse such data by software automatically. A similar situation arises if the screen capturing method is used: one does not need to fulfil high requirements in hardware and human resources but the results after the sessions are videos that have to be analysed by humans again.

Mousemaps are based on event recording. Every user input with mice, keyboards, joysticks or other devices send signals to the operating systems, so called *messages*. The event recorder *ObSys* captures every message from input devices that MS Windows operating systems put in the message queues. So far this method seems to be similar to those from tools like Kaldi (Al-Qaimari & McRosstie, 1999), GUITESTER (Okada & Asahi 1999) and others. In contradiction to those systems *ObSys* offers synchronisation to video files from cameras, screen capturing tools and event logging (input of manually typed comments by predefined short cuts with timestamps). In difference to the other published tools *ObSys* can play back the captured events. With that function *ObSys* can substitute the complete screen capturing technology with the advantage of getting 150 times smaller files[1].

The availability of this variety of methods enables studies which compare the abilities of the different methods. Although there are some papers that offer method comparisons, they are predominantly literature reviews; mostly results are compared based on estimations and derivations from different studies with different subjects, conducted by different evaluators with different testing persons, see for example (Gray & Salzman, 1998), (Hilbert & Redmiles, 1999) or (Ivory & Hearst, 2001).

2 Mousemaps – what they are, how do they look

Although mousemaps are projected on two dimensional windows they give multidimensional views. Beneath lines for visualising the mouse movements colours, thickness of lines and dots are used for further visual coding: The direction of each mouse move is indicated, the mouse clicks are given and the speed of the mouse moving is shown. For demonstrating this, we take the following minimal scenario: A small window is created with a button which has to be pressed (for isolating this visually another program is running that creates only a white background). Figure 1 shows the demonstration window in two testing series. On the left hand side the mouse is moved extremely slow (constantly), whereas the right screenshot shows slower and faster movements alternating.

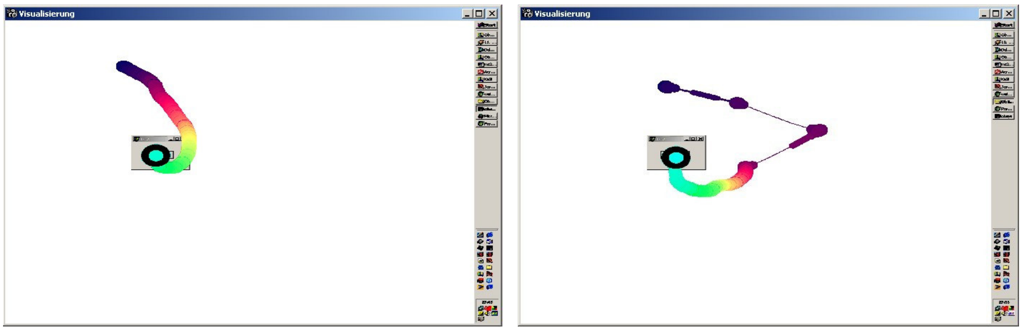

Figure 1: Visualisation of the speed of mouse moves (left slow, right mixed)

Figure 1 shows further that clicks are visualized as dots. As can be seen it can happen that after a certain number of clicks there is nothing more to recognize. For that reason *ObSys* allows to configure a range for the diameter of the clicks and for the width of the lines. The scenario in Figure 2 consists again of a minimal application: there are three buttons that have to be clicked with the left mouse button. Whereas on the left hand side there is hardly to see what lays under the click symbols the mousemap on the right gives a clear view what happened and on which positions that was (in case there are problems in finding details please consider the downscaling from originally 1280 × 1024 pixel).

[1] With one precondition that has to be considered by developing scenarios: All operations on windows (like sizing, moving, reconfiguring sub windows) and GUI elements must be reversible.

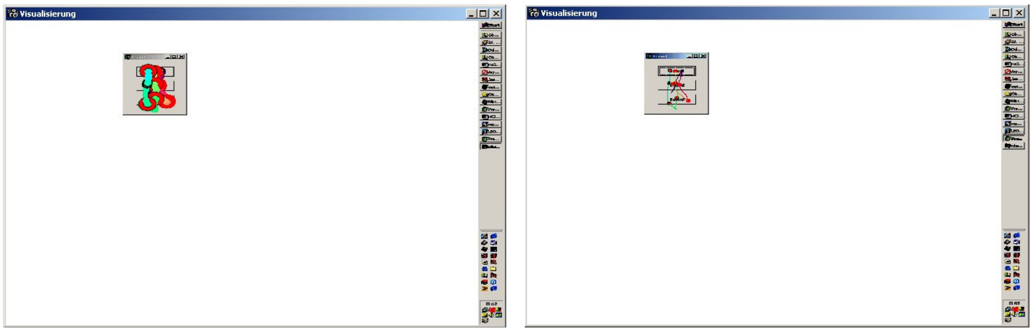

Figure 2: Visualising clicks – keeping track of the things that happened by customizing views

Another example demonstrates the visualisation of the direction to that the mouse was moved. In Figure 3 a paragraph was marked for applying any operation to that part. The first time the marking started from the left hand side and was moved to lower right, the second time the marking begun right and ended left above.

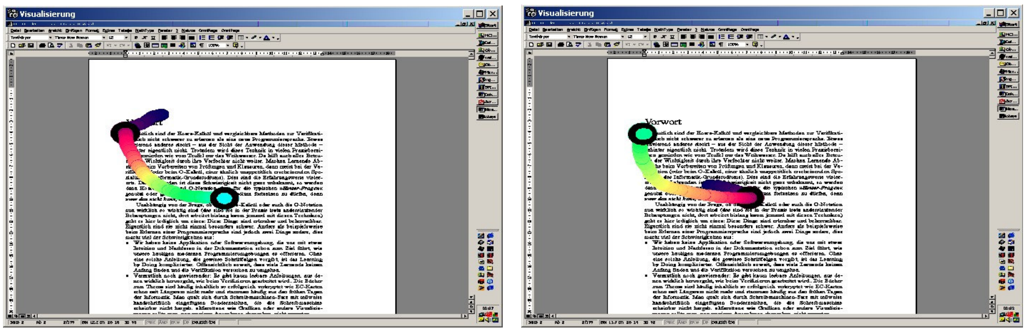

Figure 3: Marking text beginning from the left hand side (left screenshot) and beginning from the right hand side (right screenshot)

3 What do mousemaps tell us?

One of the most interesting aspects is that mousemaps give the tester a small insight into unconscious aspects of users behaviour. Quite a lot of actions seem to be accompanied by a mouse based pendant of body language. As can be seen in the figures above in general users do not take direct ways for their actions. Instead along linear paths the mouse is moved in curves similar to quarter segments of circles. Figure 4 shows the activities of an user by looking for information about publishing companies in Google (left). As the dots show only few manipulations took place.[2] Most of the movements supported the reading. *ObSys* allows to generate a view that considers only the activating actions (this excludes moves without activating further events). In this scenario moving causes no further actions like e.g. dragging. The figure on the right hand side shows the few lines that were necessary to activate the windows the user wanted. Errors of usage that would make those further acting necessary did not happen.

[2] some links were followed and some were opened in a separate window

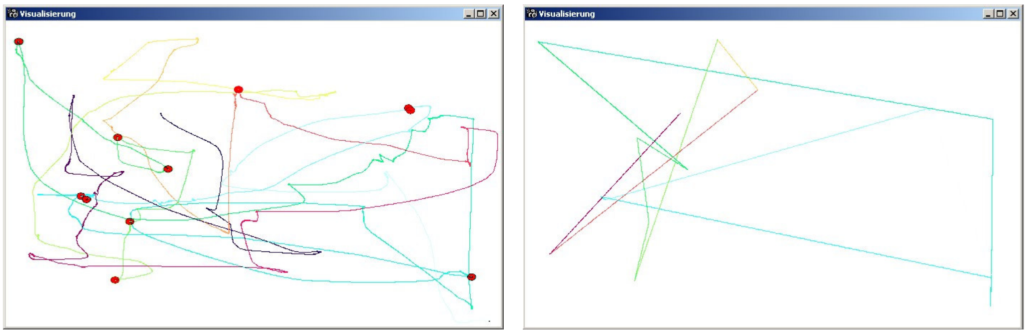

Figure 4: Mousemap with a view on all activities (left) and a click to click reduction

Contradictory explanations are possible: (I) The curvy behaviour is shown for physiological reasons. (II) As mentioned above: Similar to body language, mimic and gestures a mirror of our unconscious minds is given by using the mouse. To get more in detail at this point something that forces more precise placements of the mouse clicks has to be applied. The Windows game Minesweeper offers some features for such an investigation:

- it forces precise placements of mouse clicks
- since the user plays again time a little stress appears
- apart from the clicks the mouse can be moved free
- only clicks are relevant for that task, so the click to click view will not hide any important information

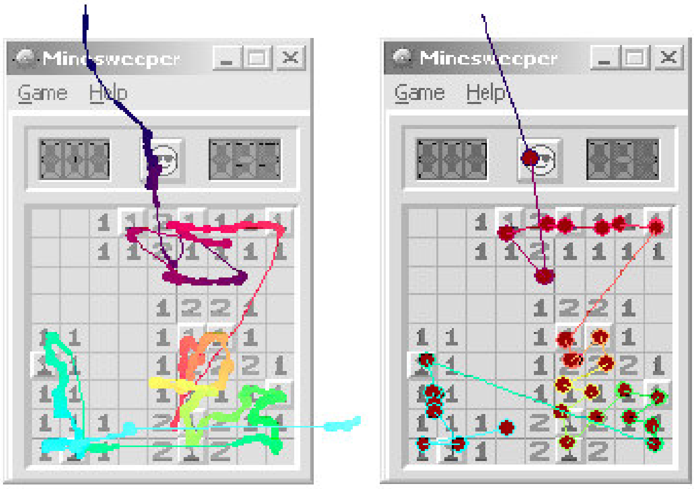

Figure 5: Mousemap for playing Minesweeper (left: all actions, right: click to click

Mousemaps for playing Minesweeper are shown in Figure 5. If the swinging movements would be necessary for physiological reasons to place the mouse on a matching position the left mousemap had to show a lot of curves. Instead of this there are nearly linear movements. The right mousemap shows the same testing series but reduced to the relevant click to click lines. In difference to Figure 4 the click to click shows mostly similarities to the normal view on the left hand side for that scenario. Based on that few data the physiological aspects seem to be not dominantly.

4 Conclusions and Perspectives

As (Smith et al., 2000) published based on eye tracking studies patterns of interaction between eye and mouse can be found. As outlined it is possible that acting with the mouse is one more way for communicating unconsciously. At the moment we tend to interpret the curvy mouse moving as a »normal« behaviour. *Normal* in that context means: without stress or forces from outside (like matching lines for assigning colours to them). Automating usability evaluations based on mousemaps were possible if one finds further indicators for the inner state of the user. This is our approach for further work.

This paper expresses our first experiences with our tool *ObSys* and the implemented multidimensional visualisation technology. On the one hand there are many aspects that we want to have implemented for further evaluations: The mouse event to key event ratio, the workflow jitter by using mouse and keyboard, an indicator for the time an input device is used and lots more. On the other hand a high number of evaluations with many scenarios and types of users are necessary to contain more facts with a high degree of clarity.

References

Al-Qaimari, G. & McRosstie (1999). *KALDI: A Computer-Aided Usability Engineering Tool for Supporting Testing and Analysis of Human-Computer Interaction*. In: Vanderdonckt, J.; Puerta A. (Ed.), Computer Aided Desgin of User Interfaces II (Proceedings of the Third International Conference on Computer-Aided Design of User Interfaces), Kluwer Academic Publishers, Lovain-la-Neuve, pp. 337 - 355.

Gray, W. D. & Salzman, M. C. (1998). *Damaged Merchandise? A Review of Experiments That Compare Usability Evaluation Methods*. In: Moran, T.D. (Ed.): *Human-Computer Interaction, A Journal of Theoretical, Empirical and Methodological Issues of User Science and of System Design*, Vol. 13, (3), pp. 203 - 261.

Hilbert, D.M. & Redmiles D.F. (1999). *Extracting Usability information from user interface events*. Technical Report UCI-ICS-99-40, Irvine: Department of Information and Computer Science, University of California.

Ivory, M. Y. & Hearst, M. A. (2001). *The State of the Art in Automating Usability Evaluation of User Interfaces*. In: ACM Computer Surveys, Vol. 33, No. 4, pp. 470 - 516.

Mayhew, D. J. (1999). *The Usability Engineering Lifecycle.* San Francisco: Morgan Kaufmann Publishers, Inc.

Preece, J. (1994). *Human-Computer-Interaction*. Harlow: Addison-Wesley.

Okada, H. & Asahi, T. (1999). *GUITESTER: A log-based usability testing tool for Graphical User Interfaces.* In: IEICE Transactions on information and systems. Volume E82, pp. 1030 - 1041.

Smith, B. A., Ho, J., Ark, W. & Zhai, S. (2000). *Hand eye coordination patterns in target selection*. In: Proceedings of the symposium on Eye tracking research & applications 2000, ACM Press, pp. 117 - 122.