

Measuring and Analyzing the Performance of the GUI Screens using Hotspot Identification

Ch. Satyananda Reddy¹ and Tadi Srinivas²

¹Andhra University, ²GITAM University
Visakhapatnam, INDIA.

¹satyanandau@yahoo.com, ²tadisrinivas_reddy@hotmail.com

Abstract

User Interfaces are the main windows which are used by the people to interact with the software program so they must be designed in better way which attracts the attention of the user. Software programs need more than expected time to learn and use, because of the poor design of User Interfaces. Therefore when designing software programs, it is necessary to find hotspots on the User Interfaces where the user gives more attention than others. In this paper, it is proposed a method of performance evaluation of the User Interfaces of the software programs by testing the software program with actual users of the software program and tracking their gaze point against the time. By using that information the software developers can find the hotspots of the User Interfaces and include the most important things to the user in those areas. So the software developers will be able to develop the User Interfaces of their programs attractive, efficient and effective manner which get more attention of the users of that software program.

Keywords: Human Computer Interaction; Eye Gaze tracking; Pupil extraction; Harr-feature classifiers; User Interfaces; Hotspot identification.

1. Introduction

Today, many software programs are used by most of the people and so that they have to interact with the computers and those software programs. When using these programs, the speed and the accuracy of using them are mostly depend on the person's attitude to the programs [1]. Therefore the success of a certain software program depends basically on the time taken to learn about the software program for the user and the performance and efficiency of that software program.

Even though many software programs are developed in these days, numerous problems can be seen when people are interacting with those programs. This is mainly because most of the User Interfaces (UIs) of these software programs are not developed according the UI design principles of Human Computer Interaction (HCI). This will mainly affect the human interaction with the software and because of that user will take more time to complete the tasks of the program than the expected. So it is necessary to evaluate the performance of the software program and to check whether users can easily learn about the program when designing UIs. Most of the time, this is done by the manual methods such as interviews or questionnaires but the feedbacks from those manual methods are not much acceptable.

This paper proposes and describes a method which can be used to evaluate the performance of UIs of software program by identifying hotspots of the UIs. System uses the gaze point of the actual users when they are using the software program and measuring gaze point coordinates against the number of occurrences and the hotspots in the UIs can be identified.

This paper is organized as follows. In section two, background information of software performance evaluation methods and eye gaze tracking techniques are discussed.

Section 3 describes the method of measuring the gaze point coordinates and identifying hotspots. Section 4 focuses on experiments and summarizes the results. A discussion on the results and its analysis is presented in Section 5. The final section concludes that the hotspots of user interfaces can be identified correctly and hence the accuracy and efficiency of the user interfaces can be improved.

2. Background and Related Work

The goal of evaluation of the software programs is to identify problems in the design of the software program [2] especially in the UIs where users interact with the software program. Mainly the evaluation is done through expert analysis and through user participation.

Evaluation through expert analysis is done by designer or the human factor expert so it mainly depends on that person. The main intention is to find any areas that cause difficulties as they violate cognitive principles or ignore empirical results [2]. As this kind of evaluations are performed by people who are not the actual users of the software program, results may vary when actual users are using the software program and also leads to reducing the performance of the program.

Evaluation through user participation will generate more acceptable results than the above methods as evaluation is done when the actual users are using the system or the software program. Though this kind of evaluations are better than expert analysis there is no proper way of getting correct data to generate results and reports.

2.1. Software Performance Evaluation

Recording user interactions with software programs through user participation and using those data for the evaluation of the software program is an evaluation method which is called Protocol Analysis (PA) [2]. In this method all the responses of the user including psychological response, view of user on the screen *etc.* will be recorded and later used for the evaluation purpose. As it is much difficult to analyze data which contains users' responses manually, Video Annotator Systems can be used as an automatic PA tool that can be used in the performance evaluation of the software programs.

Experimental Video Annotator (EVA) is a system that runs on a multimedia workstation with a direct link to a video recorder [3]. It records the user's responses when he or she works at the workstation and then evaluator can review them using the EVA System. Based on the research done on using PA [4] to evaluate usability of a commercial website, the usability of a website or a software program defines that how well and how easily a visitor without formal training can interact with the software program or the web site. In his research, commercial web sites were evaluated using PA method which asked the users to interact with the site and recorded actions and keystrokes of the users by using numerous PA tools [5]. These captures from the concurrent protocols included how users approach a task and why problems occurred when they interact with the system or the website [6]. It also stated that, this method can be used to understand how users from their cognitive model of the system work and investigate cause of errors, mistakes or misinterpretations. According to Tilson et al. [7] the usability problems in software programs or systems can be divided into three categories such as content, navigation and interactivity. They also proposed specific design recommendations to each of these problems by using the usability principles which were identified by the researchers.

2.2. Eye and Gaze Tracking

A number of eye and gaze tracking techniques are already available and used worldwide [8]. There are Electro-Oculography tracking techniques [9] which use

electrodes on the skin of eye and track the change of electrostatic field to track the eye and gaze but these are highly sophisticated and not suitable for many applications. Also some of people use video based systems which needs high processing power for eye and gaze tracking. These systems include Corneal Reflections [10] and Purkinje Image Tracking [11] *etc.*

Although there are many kinds of software performance evaluation methods and eye & gaze tracking techniques, there are no approaches which combine these two technologies to evaluate the performance of the UIs of software programs in accurate manner. So this paper presents a method which uses an algorithm to map eye gaze point of the user to the screen and evaluate the performance of the software program by identifying the hotspots in the UIs.

3. Gaze Point Measurement & Hotspot Identification

In this research, it is proposed a method to identify the hotspots of the UIs of software programs, by calculating the number of occurrences which user keeps his gaze on those points. Because of that there will be two main types of algorithms which can be identified as eye and gaze tracking algorithm and the hotspot identification mechanism. The accuracy of this proposed method mainly depends on the accuracy of the results of these two algorithms.

3.1. Eye and Gaze Tracking Algorithm

Eye and gaze point coordinates are used as the main input for identification of hotspots in UIs. In the proposed algorithm there are three sub modules as face detection, eye and pupil extraction and mapping the gaze point to screen targets. Figure 1 gives an overview of the Eye and Gaze tracking algorithm.

Algorithm is designed to work on grayscale images. In the first stage, the images from the web-cam are converted to grayscale. The histogram equalization is performed for the images to normalize the contrast of the image, so that the pixel intensities will be adjusted in accordance with the histogram of the image [12]. Then it is necessary to detect face from the acquired image. There is a rapid and robust object detection algorithm which can be used for face and eye detection known as Harr-feature based classifiers [13]. These classifiers are calculated to identify difference between the dark and light regions of the images by using the pixel values.

In the procedure of face detection using these Harr- features the acquired image is sampled and face is detected using classifier and those coordinates are mapped to the original image. Also this needs to be done in minimum amount of time as there is a need of real time video processing for the proposed approach.

Pupil detection from the eye image is very important and the measurement of eye gaze coordinates totally based on the relative displacement of the pupil from the center of eye. For that first the iris is detected from the eye and the pupil can be found from the center of iris. At the initial step the iris & pupil must be clearly detected from both eyes using Hough Circle Transformation (HCT) which can identify circles from the binary images. The next significant step is to map the eye gaze coordinates to the screen targets and this is done by using a mathematical method. For this first the eye corners needs to be identified which can be used to estimate the width, height and the center of the eye [8].

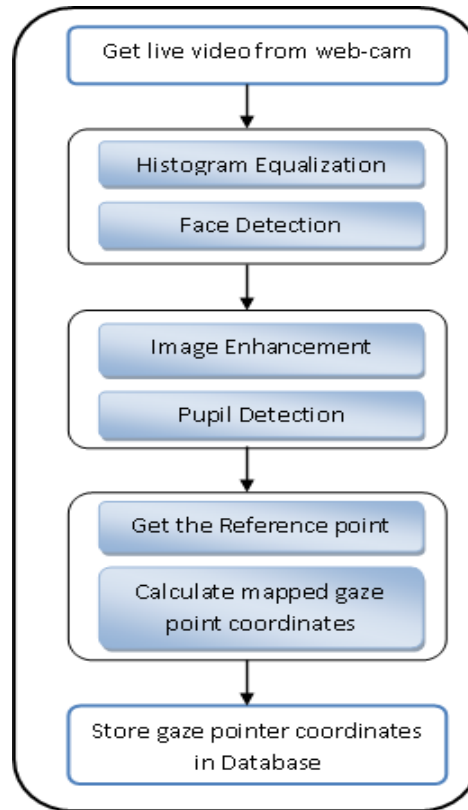


Figure 1. Eye and Gaze Tracking Algorithm

Then a reference point needs to be calculated and in this approach Center of Eye (CE) is taken as the reference point for all of the calculations. The Equation 1 and Equation 2 present the method of calculating the coordinates of the reference point where Top Left Corner (TLC), Top Right Corner (TRC), Bottom Left Corner (BLC) and Bottom Right Corner (BRC) are the margins of rectangle which represents the movable region of the eye.

$$CE_x = \frac{TLC_x + TRC_x}{2} \quad (1)$$

$$CE_y = \frac{TLC_y + TRC_y}{2} \quad (2)$$

CE_x and CE_y are the x,y coordinates of the center of eye and then height and width of movable region of the eye need to be calculated by using Equation 3 and Equation 4.

$$Width_{eye} = |TRC_x - TLC_x| \quad (3)$$

$$Height_{eye} = |BLC_y - TLC_y| \quad (4)$$

Then the scaling factor needs to be calculated which is required in mapping eye gaze coordinates to the screen targets. Scaling factor for x direction and y direction can be calculated as in Equation 5 and Equation 6 where $Width_{screen}$ & $Height_{screen}$ are the width & the height of the screen.

$$Scale_x = \frac{Width_{screen}}{Width_{eye}} \quad (5)$$

$$Scale_y = \frac{Height_{screen}}{Height_{eye}} \quad (6)$$

The gaze point coordinates are calculated by assuming the center point of the eye that reflects to the center point of the screen. Equation 7 and Equation 8 refers to the calculation of gaze point coordinates and Equation 9 and Equation 10 shows how to get displacement of pupil from eye reface point (CE) in both x, y directions.

$$GPoint_x = \frac{Width_{screen}}{2} + (Scale_x \times Displacement_x) \quad (7)$$

$$GPoint_y = \frac{Height_{screen}}{2} + (Scale_y \times Displacement_y) \quad (8)$$

$$Displacement_x = |PupilLocation_x - CE_x| \quad (9)$$

$$Displacement_y = |PupilLocation_y - CE_y| \quad (10)$$

Coordinates of the Gaze Point $GPoint_x$ and $GPoint_y$ are recorded in a database for further analysis and identification of the hotspots in UI screens. These coordinates represents the places on the screen where the user looks when using the software program. Figure 2 shows the gaze point of user on the screen and Figure 3 shows the reaction of pupil of the user's eye correspondent to gaze point in Figure 2.

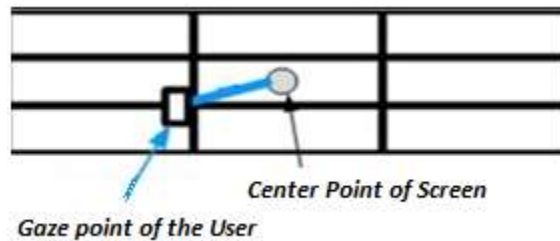


Figure 2. Gaze Point Location of the User on Screen

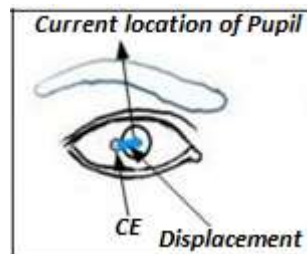


Figure 3. Pupil Displacement from Reference Point (CE)

3.2. Hotspot Identification Mechanism

Using the above mentioned algorithm, the gaze point of the user can be measured when he or she uses the software programs. The next task in the proposed method is to identify the hotspots on UIs by analyzing the gaze point coordinates and the timestamps which are recorded in the database.

First the records which contain the gaze point coordinates will be optimized and then new records will be plotted on a diagram for each UI which shows the hotspots in UI with colored circles. The radius of these colored circles are correspondent to number of occurrences that user kept looking that area when he or she using the software program.

In the proposed algorithm for hotspot identification, it compares the total records in the table, one by one with each other. For all records in the table, records will be grouped if the coordinate difference is less than predefined value for both x, y coordinates and the algorithm will calculate the mean x, y values for each group and plot

a diagram with circles whose radiuses are correspondent to the number of occurrences in that area of the screen.

4. Experimental Results

In this research, two main types of experiments were conducted for eye and pupil tracking with gaze point measurement and hotspot identification. Accuracy of these algorithms is measured as described below.

4.1. Experiments on Gaze Point Calculation

This experiment was divided into sub experiments as face and eye detection, pupil extraction and calculating the gaze point coordinates on the screen. The face and eye detection by the using Harr-feature classifiers is presented in Figure 4.



Figure 4. Detection of Eyes using Harr-feature Classifiers

Accuracy of the gaze point calculation algorithm was measured using an image which has numbered items in definded locations on the screen and asking the user to look at those numbers with a given time for each number. The records that were stored in the database were analyzed to check whether that the gaze point calculation was performed in an accurate manner. As the timestamps also recorded in the database efficiency of the face detection, pupil extraction and gaze point calculation also measured in this experiment. Table 1 & Table 2 presents the accuracy and efficiency results of face detection, pupil extraction and gaze point calculation algorithm.

4.2. Experiments on Identifying of Hotspots

Experiments were conducted to measure the efficiency of hotspot identification mechanism. In this algorithm, it checks the difference between two records in the database for each record and if the difference is less than the predefined value then records will be grouped. After that mean point of each group is plotted as circles for each UI as circles whose radius correspondent to number of records in the group. Figure 5 presents identified hotspots of an experiment which was conducted using this mechanism.



Figure 5. Hotspot Identification Experiment Results

The accuracy of the hotspot identification mainly depends on the accuracy level that can be gained by the gaze point calculation algorithm. Also the processing time of the algorithm used in hotspot identification mainly affected by the total number of records per UI and number of UIs and Table 3 presents the efficiency results of the algorithm used for hotspot identification.

Table 1. Accuracy of Gaze Point Measurement

Feature	Accuracy
Face & Eye Detection	100 %
Pupil Extraction	75% - 80%
Gaze Point Calculation	100 %

Table 2. Efficiency of Gaze Point Measurement

Feature	Processing Time (ms)
Face Detection	15 - 17
Eye Detection	10 - 15
Pupil Extraction	15 - 20
Gaze Point Calculation	02 - 05

Table 3. Efficiency of Hotspot Identification Algorithm

Step	Time taken (ms)
Grouping Records of one UI	25 - 30
Identification of hotspots	05 - 07
Plotting Results	03 - 05

5. Discussion

To implement the proposed method, mainly two algorithms were used. They are eye and gaze tracking algorithm, and the hotspot identification algorithm. When considering of the eye and gaze tracking algorithm, it assumes that the center of the screen reflects to the center of eye (CE). It is used as the reference point for the calculation of gaze point of the user on screen. The algorithm can be further divided into sub parts as face and eye detection, pupil extraction and the calculation of gaze point and for each of these component the accuracy and the efficiency was measured as presented in Table 1 and Table 2.

For face and eye detection Harr-feature classifiers were used and it can be seen that within 30 milliseconds face and eyes can be detected from a real time video frames with 100% accuracy. This accuracy can be gained as the classifiers that are used for this algorithm were trained with many of positive and negative sample images [13]. After eyes were detected pupil extraction needs be performed and when doing the experiments it can be seen that 75% to 80% accuracy in extracting pupil from the eye image. This is mainly because in the eyes of Asian people iris and pupil are not clearly distinguishable and

because of that sometimes the pupil extraction will not be performed clearly. Final part of this algorithm is to calculate the gaze point and that can be done within 5 milliseconds with an accuracy of 100%. In this gaze point calculations, as mentioned earlier, reference point is kept as a fixed point (center of the screen) and if the user moves his head to a side then the algorithm will give incorrect gaze point coordinates. Therefore the experiments were done by advising the user to keep his head in front of the center of the screen. For the further development of this algorithm it is needed to improve accuracy of the pupil extraction by using digital image processing techniques to make iris and pupil are clearly distinguishable and also to improve this algorithm to use a variable reference point which allows user to freely move his head while using the software programs.

By considering the efficiency results of the hotspot identification algorithm as presented in Table 3, it can be seen that grouping of records took a larger amount of time when compared with identification of hotspots and plotting them on screen. This is because for the grouping, the algorithm needs to check records that were stored in the database one by one. Algorithm can be further improved and time can be reduced if an optimizing technique can be used when inserting records to the database instead of grouping them after getting all the records. Other than that identification of coordinates of the hotspots in the UI and plotting them on UI screens can be done within about 15 milliseconds. The performance of each UI can be evaluated and summary report can be generated to each UI and for the complete software program.

6. Conclusions

Many software programs are Graphical User Interface (GUI) based and includes number of UIs which are used by the users of that program to interact with it. UI designing is not an easy task and user interfaces of the most software programs are confusing, complex and not easy to understand. From the perspective of the user of a software program, it needs time to learn about the software programs but when the UIs are complex and confusing it will take more than the expected time to learn about interacting with the software programs.

When considering a design of an UI, there are hotspot places which get attention most users easily. Users are mostly looking at those places and will not give much attention to other places and widgets which increases the time taken to perform tasks by using the software programs. Most of users accepted that there should be a mechanism to evaluate the performance of the UIs of the software programs before they are released. To that extent, in this paper, it is proposed a method to evaluate the performance of UIs using eye and gaze tracking of user.

In this research, the user's eye and gaze are tracked by using web-cam and gaze points are recorded in a database when the user interacts with the software program. Using those gaze point coordinates, hotspot coordinates are identified and they can be plotted in original UI screen images helping the developer of the software program to identify the hotspot areas in each UI. In this research, it has been used two main algorithms, one is to track the users gaze point and next is to identify hotspots of UIs. Based on the experiments conducted, it is concluded that the accuracy of eye gaze tracking algorithm is about 90% and the hotspot identification algorithm is about 95% percent. Accuracy of the overall method is about 92% percent and by using this method the hotspots of UIs can be identified correctly by checking them with actual user interactions. The proposed method will be helpful for software developers to evaluate UIs of their software programs and design UIs in effective and efficient manner which takes least amount of time for the users to perform tasks using them.

References

- [1] J. Chin, V. Diehl and K. Norman, 'Development of a Tool Measuring User Satisfaction of the Human-Computer Interface', University of Maryland, (1988).
- [2] Allan Dix, Human-computer interaction, Harlow, England: Pearson/Prentice-Hall, (2004).
- [3] W. Mackay, "EVA: an experimental video annotator for symbolic analysis of video data," SIGCHI Bull., vol. 21, no. 2, (1989), pp. 68-71.
- [4] R. Benbunan-Fich, "Using protocol analysis to evaluate the usability of a commercial web site," Information & Management, vol. 39, no. 2, (2001), pp. 151-163.
- [5] P. Todd and I. Benbasat, "Process Tracing Methods in Decision Support Systems Research: Exploring the Black Box," MIS Quarterly, vol. 11, no. 4, (1987), p. 493.
- [6] A. Scharl and C. Bauer, "Explorative analysis and evaluation of commercial web information systems," Proceedings of the Twentieth International Conference on Information Systems, (1999), pp. 534-539.
- [7] R. Tilson, J. Dong, S. Martin and E. Kieke, "Factors and principles affecting the usability of four E-commerce sites," Proceedings of the 4th Conference on Human Factors and the Wbe, (1998).
- [8] M. Ghani, S. Chaudhry, M. Sohail and M. Geelani, 'GazePoint: A real time mouse point control implementation based on eye gaze tracking', INMIC, (2013).
- [9] R. Barea, L. Boquete, M. Mazo and E. Lopez, 'System for assisted mobility using eye movements based on electrooculography', IEEE Trans. Neural Syst. Rehabil. Eng., vol. 10, no. 4, (2002), pp. 209-218.
- [10] D. H. Yoo, J. H. Kim, B. R. Lee and M. J. Chung, 'Non-contact eye gaze tracking system by mapping of corneal reflections', Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition, (2002).
- [11] K. Arai and M. Yamaura, "Computer Input with Human Eyes-Only Using Two Purkinje Images Which Works in a Real-Time Basis without Calibration," CSC Journals, vol. 1, no.3, (2010), pp.71-82.
- [12] R. Gonzalez and R. Woods, Digital Image Processing, 3rd ed.: Pearson Education, (2009).
- [13] P. Viola and M. Jones, 'Rapid object detection using a boosted cascade of simple features', Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR, vol. 1, (2001), pp. 511-518.

Authors



Ch. Satyananda Reddy is an Associate Professor in Computer Science and Systems Engineering Department, College of Engineering, Andhra University, Visakhapatnam. His research interests include Software Engineering, Software Project and Process Management, Software Estimation, Software Metrics, Software Project Scheduling, Software Quality Assurance, Human Computer Interaction, Software Requirements Engineering, Software Architecture, Software Testing, Data Engineering, Documents Summarization.



Tadi Srinivas is an Associate Professor in Computer Science and Engineering Department, GITAM Institute of Technology, GITAM University, Visakhapatnam. His research interests include Software Engineering, Software Architecture, Software Project Management, Software Estimation, Software Project Scheduling, Human Computer Interaction, Computer Networks, Cryptography and Network Security.

