Performance Evaluation of the User Interfaces in Software Programs using Eye & Gaze Tracking

Dumira Athukorala, Himali Ganhewa, Uvindhu Jayasinghe, Dinuka Karunarathne, Lochandaka Ranathunga
Department of Information Technology,University of Moratuwa
Katubedda, Sri Lanka
{dumira1990, himaliganhewa, uvindhu, akdmkarunarathne}@gmail.com, lochandaka@uom.lk

Abstract

There are many software programs developed by many software developers in these days. The people who are going to interact with them need some time to learn about how to do their tasks using those software programs. Some of these software programs need more than expected time to learn about using them because of poor design of the User Interfaces. User Interfaces are the main windows which are used by the people to interact with the software programs, because of that they must be designed in better way which attracts the attention of the user. Therefore in Human Computer Interaction principles, there some principles for User Interfaces design effective User Interfaces to attract attention of the user to the program. When designing software programs it is necessary to find hotspots on the User Interfaces where the user gives more attention than others. This paper proposes 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 gaze point of users with 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. The software developers will be able to develop the User Interfaces of their programs attractive, efficient & effective manner which get more attention of the users of that software program.

Key Terms:

Human Computer Interaction, Protocol Analysis, Eye tracking, Pupil extraction, Gaze point, Performance evaluation, Hotspot identification.

1. INTRODUCTION

Today there are many software programs used by most of the people and because of that they have to interact with those software programs. When using these programs, the speed and the accuracy of using them 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 & efficiency of that software program. Although many software programs are developed in these days, we can see problems 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 to 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 & to check whether users can easily learn about the program when designing UIs. Most of the time this is by the manual methods such as interviews or questionnaires but the feedbacks from those manual methods are not acceptable in some cases because the people may provide false information.

This paper proposes & 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 2, background information of software performance evaluation methods and eye tracking techniques are discussed. Section 3 describes the method of measuring the gaze point coordinates & identifying hotspots and Section 4 focuses on testing with the system and getting results. Section 5 consists of a discussion & Section 6 will conclude this paper with a conclusion and further work.

2. BACKGROUND & RELATED RESEARCHES

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 & through user participation.

Evaluation through expert analysis is done by designer or the human factor expert and 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 and using those data for the evaluation of the software program is an evaluation method which is used in evaluation through user participation and it 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 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 a research which was done on using PA to evaluate usability of a commercial website, it was said that 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 [4]. In that 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. 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 [5].

It is also mentioned that PA tools can be used to understand how users use their cognitive thinking and investigate cause of errors, mistakes or misinterpretations [4]. According to the results of that research usability problems in software programs or systems can be divided into three categories such as content, navigation and interactivity [6]. They have also proposed specific design recommendations to each of these problems by using the usability principles which were identified by the researchers [7].

2.2 Eye & Gaze Tracking

A number of eye & 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 & gaze but these are highly sophisticated and not suitable for many applications. Also some people use video based systems which needs high processing power for eye & 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 proposed method the hotspots of the UIs of software programs are identified 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 & Gaze Tracking Algorithm

Eye & gaze point coordinates are used as the main input for identification of hotspots in UIs. In the algorithm which is used in the method proposed in this paper there are three sub modules as face detection, eye & pupil extraction and mapping the gaze point to screen targets. An overview of the algorithm and these modules are presented in Fig.1.

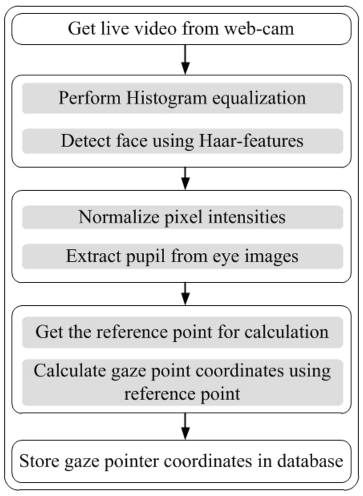
Get live video from web-cam
Perform Histogram equalization
Detect face using Haar-features
Normalize pixel intensities
Extract pupil from eye images
Get the reference point for calculation
Calculate gaze point coordinates using reference point
Store gaze pointer coordinates in database

Figure 1: Overview of Eye & Gaze Tracking Algorithm

Algorithm is designed to work on grayscale images so first the images from the web-cam are converted to grayscale. After that histogram equalization is performed for the images normalize the contrast of the image which adjusts the pixel intensities in accordance with the histogram of the image [12].

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 & eye detection known as Harr-feature based classifiers[13]. These classifiers are calculated to identify difference between the dark & 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 form 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 form both eyes using Hough Circle Transformation (HCT) which can identify circles from the binary images.

Next important 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 & the center of the eye [8].

Then a reference point needs to be calculated & in this approach Center of Eye (CE) is taken as the reference point for all of the calculations. The eq.1 1 and eq.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) & Bottom Right Corner (BRC) are the margins of rectangle which represents the movable region of the eye.

$$CE_x = \frac{TLC_x + TRC_x}{2} \tag{1}$$

$$CE_y = \frac{TRC_y + BRC_y}{2} \tag{2}$$

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

$$Width_{eye} = |TRC_x - TLC_x| \tag{3}$$

$$Hight_{eye} = |BLC_y - TLC_y| \tag{4}$$

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 eq.5 and eq.6 where $Width_{screen}$ & $Height_{screen}$ are the width & the height of the screen.

$$Scale_x = \frac{Width_{screen}}{Width_{eye}} \tag{5}$$

$$Scale_y = \frac{Hight_{screen}}{Height_{eye}} \tag{6}$$

After scale factors are calculated the gaze point coordinates can be calculated by assuming the center point of the eye reflects to the center point of the screen. Eq.7 and eq.8 refer to the calculation of gaze point coordinates and, eq.9 and eq.10 show how to get displacement of pupil from eye reface point (CE) in both x & y directions.

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

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

$$Displacement_x = |PupilLocation_x - CE_x| \tag{9}$$

$$Displacement_y = |PupilLocation_y - CE_y| \tag{10}$$

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

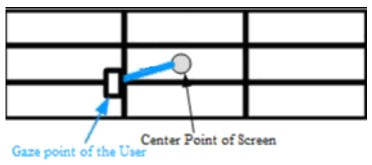

Figure 2: Gaze point location of the user on screen

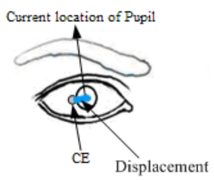

Figure 3: Pupil displacement from reference point (CE)

3.2 *Hotspot Identification Mechanism*

Using the above mentioned algorithm the gaze point of the user can be measured when he/she uses the software programs. The next task of the proposed method in this paper is to identify the hotspots on UIs by analyzing the gaze point coordinates and the timestamps which are recorded in the database. Places which get more attention of user can be categorized as the hotspots and this mechanism will identify hotspots in UIs by considering the gaze point coordinates stored in the database.

First the records which contain the gaze point coordinates will be optimized using an algorithm as shown below and then new records will be plotted on a diagram for each UI which shows the hotspots in UI with colored circles. The radiuses of colored circles are proportional to the time taken by user to keep his/her gaze point on those areas while using the software program.

Hotspot identification is done using details of gaze point coordinates stored in the database. This need to be done for each UI of the software program and because of that screen changes needs to be identified while storing the gaze point coordinates. After screen changes are identified with a timestamp the gaze point coordinates can be divided into UIs. The divided gaze point coordinates will be stored in different tables in database as one table for one UI.

The algorithm used for hotspot identification compares the total records in a table one by one with each other. Records will be grouped if the coordinate difference is less than predefined value for both x & y coordinates and after performing this for all records in the table algorithm will calculate the mean x & y values for each group and plot a diagram with circles whose radiuses are proportional to the number of occurrences in that area of the screen. This algorithm will be executed for each table in the database and plots the identified hotspots on each UI.

4. EXPERIMENTS & RESULTS

Two main types of experiments were conducted for the proposed method in the paper as testing done for eye & pupil tracking with gaze point measurement and hotspot identification algorithm. 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 & eye detection, pupil extraction and calculating the gaze point coordinates on the screen. The face & eye detection by the using Harr-feature classifiers is presented in fig.4. Accuracy

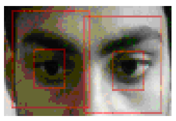

Figure 4: Detection of eyes using Harr-feature classifiers

of the gaze point calculation algorithm was measured using an image which has numbered items in defined locations on the screen & asking the user to look at those numbers with a given time for each number. After that records that were stored in the database were analyzed to check whether that the gaze point calculation was performed in an accurate manner. Accuracy of these algorithms was calculated by taking the percentage of number of correctly identified experiments & total number of experiments conducted.

As the timestamps also recorded in the database efficiency of the face detection, pupil extraction & gaze point calculation also measured in this experiment. Table.1 and Table.2 present the accuracy and efficiency results of face detection, pupil extraction & gaze point calculation algorithm.

Table 1: ACCURACY OF GAZE POINT MEASUREMENT

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

Table 2: ACCURACY OF GAZE POINT MEASUREMENT

Feature	**Processing Time (ms)**
Face	15–17
Eye Detection	10–15
Pupil Extraction	15–20
Gaze Point Calculation	2–5

4.2 Experiments on Identifying of Hotspots

There were experiments conducted to measure the efficiency of identification of hotspots. 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 radiuses are proportional to number of records in the group. Figure.5 presents identified hotspots of an experiment conducted using this mechanism. Above figure presents the identified hotspots

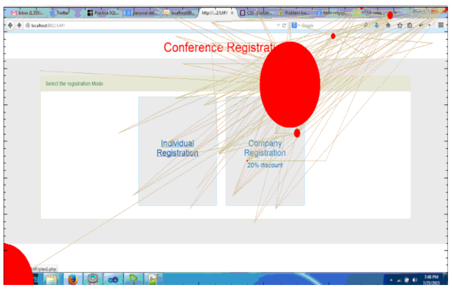

Figure 5: Hotspot identification experiment results

of a conducted experiment. Red blobs refer to the hotspot identified while someone is using the given software application. Red lines refer to the paths of the gaze point coordinates on the screen while using this application.

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.

5. DISCUSSION

To implement the method proposed mainly two algorithms were used. They are eye & gaze tracking algorithm & the hotspot identification algorithm.

Table 3: EFFICIENCY OF HOTSPOT IDENTIFICATION ALGORITHM

Step	**Time Taken (ms)**
Grouping records of one UI	25–30
Identification of hotspots	5–7
Plotting results	3–5

When considering of the eye & gaze tracking algorithm, it assumes that the center of the screen reflects to the center of eye (CE). That was 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 & eye detection, pupil extraction & 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 & 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.

When considering the efficiency results of the hotspot identification algorithm as presented in Table III, 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 which stored in the database one by one. Algorithm can be further improved & 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.

After getting outcome results for each UI as presented in Figure 5 performance of each UI can be evaluated and summary report can be given to each UI and for the complete software program.

When the hotspots are identified on UIs of software program the software developer will be able to find the types of widgets that took the attention of the user & types of objects that make user distracted from the expected path to be followed while using the software program. By using these details software developers can redesign the UIs of software program to increase the performance.

6. CONCLUSION

These days a lot of software programs are used by people all over the world. Almost all these 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 as a survey conducted by the authors of this paper results that most of users are complaining that UIs of most software programs are confusing, complex and not easy to understand. From the perspective of the user of a software program he may needs some time to learn about the software programs but when the UIs are complex & 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 of most users. 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.

The survey results also reveal that 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. Because of that the authors of this paper had proposed this method which uses eye gaze points of user to evaluate performance of software programs.

In this proposed method user's eye & gaze are tracked by using web-cam and gaze points are recorded in a database when the user interact with the software program. After that 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.

When considering of the method it uses two main algorithms, one is to track the users gaze point and next is to identify hotspots of UIs. After implementing this method some experiments were conducted and results shows 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 because of that by using this method the hotspots of UIs can be identified correctly by checking them with actual user interactions.

Authors of this paper thinks that 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.

ACKNOWLEDGMENT

Authors of this paper would like to acknowledge to the dean and other academic staff members of Faculty of Information Technology for the guidance given in conducting this research.

Authors are also indebted to Faculty of Information Technology, University of Moratuwa for giving us this opportunity to study on this topic and conduct a research on addressing the problem mentioned in this paper.

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, pp. 68-71, 1989.
[4] R. Benbunan-Fich, "Using protocol analysis to evaluate the usability of a commercial web site," Information & Management, vol. 39, no. 2, pp. 151-163, 2001.
[5] P. Todd and I. Benbasat, "Process Tracing Methods in Decision Support Systems Research: Exploring the Black Box," MIS Quarterly, vol. 11, no. 4, p. 493, 1987.
[6] A. Scharl and C. Bauer, "Explorative analysis and evaluation of commercial web information systems," Proceedings of the Twentieth International Conference on Information Systems, pp. 534-539, 1999.
[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 Web, 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, pp. 209-218, 2002.
[10] Dong Hyun Yoo, Jae Heon Kim, Bang Rae Lee and Myoung Jin 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, pp. 71-82, 2010.
[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 2001, vol. 1, pp. 511-518, 2001.