(SMT) REAL TIME MOUSE TRACKING REGISTRATION AND VISUALIZATION TOOL FOR USABILITY EVALUATION ON WEBSITES

Luis A. Leiva Torres
Technical University Of Valencia
Camino de Vera, s/n Valencia (Spain)
luileito@doctor.upv.es

Roberto Vivo Hernando
Technical University Of Valencia
Camino de Vera, s/n Valencia (Spain)
rvivo@dsic.upv.es

ABSTRACT

Mouse tracking in websites using standard web technologies is an interesting although not obvious user interface analysis tool, a great complement to classic server logs and usability tests. This paper is mainly focused on user interaction through mouse movements and clicks. A mouse tracking system is introduced to register, manage and visualize user movements on websites in real time, which is a good starting point for modelling user conducts and tendency profiles.

KEYWORDS

mouse tracking, real time visualization, Computer-Human Interation, web interaction, usability evaluation, user modeling.

1. INTRODUCTION

The Web is an attention economy where the main currency is the time that a user remains in a site [6]. Content is king. It is the reason why users come to Internet. Therefore, content structuring is fundamental, both internally and externally. The external part is the one that will have a more direct influence on the user when it comes to interacting with contents by means of a correct layout. This is the user interface. Nevertheless, in order to study the role of the human being in the interactive systems design, in addition to the computer science aspect, it is necessary to consider other disciplines. It is necessary to deal with the psychological aspects of the user, equipment ergonomics, social aspects, design subjects and many other that require more than good programming knowledge.

The standard user mainly operates in a website interface with the mouse and, to a lesser extent, the keyboard. The mouse is undoubtedly the most widely used device: it is under the user's hand and he decides everything. One of the main procedures to evaluate the usability of an interface, not only web, is the use of eye tracking, due to its high accuracy and reliability, providing a suitable and complete analysis on the user experience within this interface [4]. Some studies [3], [9] suggest that there is a strong correlation between eye and mouse movements while surfing the Web. For that reason, in the new information spaces it is possible to implement a mouse tracking at different levels of granularity, with the intention of evaluating the aforementioned activity of the user. Our tool is intended to work as a complement to these eye-tracking systems, not as a supplement, because users first focus their attention and then execute actions. The fact is that velocity and time are taken into account on our system, so these variables can be added to usability tests.

Given the vast number of webs which currently exist and the potential number of visitors, sometimes it is necessary to evaluate the user interaction in a faster way. This tool has been designed also to help administrators and webmasters to make fast interactivity tests and, above all, to analyse the gathered data.

We have called our system (smt) Simple Mouse Tracking, because it is a simple but powerful mouse tracking tool for web pages. It works at a high level of interaction with this device, registering and visualizing in an independent layer all the movements made by the user in real time. It is possible to observe how people interact with the site interface: where they click, what they highlight, how much time they spend reading some text, among other tasks.

There are many ways to register mouse movements but the most interesting one is the less obtrusive way, without either installing any additional software in the client side or interfering with the user's browsing habits [7]. This requirement, far from operating in a controlled environment it opens up a whole range of almost infinite possibilities, considering the large number of existing web browser versions and screen resolutions. Specifically, in new information spaces the most popular alternative are both the clickmaps and the heatmaps, which are digital maps that store information of clicks that users have made on a web page. This is the fastest and simplest way to implement, since it is an already solved problem, therefore obtaining similar results, although these are not as precise as those offered by eye tracking. Beyond usual statistics, these maps can find where users are clicking. This is quite useful to find areas in needing of change, layouts that do not work as intended or anchors that are not being understood as the web designer would like.

Yet, this information can be insufficient when it comes to facing a more complex analysis of the interface. For example, the path followed by the user until deciding to click a hyperlink or exiting the page cannot be determined. For that reason other alternatives have appeared, like [5]. Mouse movements were recorded through a web server and an occurrences map was then generated with these mouse coordinates. Nevertheless, in this system a direction vector is not considered in order to determine —again— where the pointer came from and where it went. Thus, in [1] with an application that works in the server side, mouse tracking is made and a map of positions, directions and clicks is created, emphasizing the entry and exit points of the web page. Consequently, with all this information they tried to infer the users' navigational habits in a more precise way. Almost in parallel, in [2] the methodology used is similar: by using a Javascript application in the client side the data are recorded, through a proxy server that process all HTTP client traffic the data are logged and structured, and they are processed through a server side application.

The great advantage of these previous methods is that the user does not perceive that he is being logged, and he does not need to install aditional plugins in his client application, he only must have a web browser. Nevertheless, these methods do not consider real time mouse tracking visualization.

The most ambitious attempts at automation of usability testing come from the Artificial Intelligence community. Many approaches are extremely compelling and interesting but doomed to fail due to that we really have not been able to generate truly autonomous user agents that act like humans [8]. Past and present attempts have been extremely costly, overly simplistic, and totally specific to the task and application. More modest approaches to automation attempt to record, analyze, and interpret interface events of real users performing either simulated tasks in the laboratory or real tasks in an environment.

Although in this article the system (smt) is introduced as a tool for real time mouse tracking visualization, its main goal goes beyond that: to analyse user behaviour and to infer a visitor profile by means of psychological, social and statistical relations.

2. SYSTEM FEATURES AND GOALS

Since (smt) is a tool for anyone interested in how visitors interact within their website, the main goal is to register mouse movements in real time and in any web page, a document in (X)HTML format. With this system the webmaster or site administrator can replay and visualize in real time the movements that a certain user has made in their pages. The most direct application is the user's interactivity evaluation from any computer across the world, which supposes a considerable amount of samplings to process. The number of potential visitors that can enter to a website is huge and more enriching in terms of real data: different users with different computers, different browsers, different computer knowledge. In short there is a wider range of possibilities than the one that can be obtained in a laboratory.

The main aim is that the (smt) system is quite simple to implement, regarding both set up and use. For that reason, neither the client nor the server need to install additional software. Neither is it necessary to use a database, although this could be a requirement with regard to each webmaster's needs.

Another significant aim is that the proposed system is able to work in any type of layout —in terms of content diagramation— and under any computer or client machine, apart from being independent of its screen resolution and the window size in which the web page is visualized. That is why the stored mouse path coordinates are normalized in real time while replaying them, according to a discrepancy ratio, which informs about the correlation that exists between the dimensions of the current window —where the mouse path is visualized— and the window of origin —where the mouse path was registered—. Therefore, a unitary value in both width and height indicates that both windows have the same size, which only happens when the two web browsers are also the same. When this ratio is not the unit, which is the most frequent situation, the normalized values vary noticeably from the previous ones. This is because of the decimal truncation that inevitably takes place, in order to lighten the computational load.

This tool works in any operating system with any modern web browser: IExplorer, Firefox, Opera, Safari or Gecko, among others. This system does not interfere with previous or later scripts in the page, which is, of course, a desirable and required feature. It is also possible to easily set the registration precision and the folders where each session reports will be stored, as well as the recording timeout[1].

A very interesting feature is that this tool begins to check the mouse activity when the Document Object Model (DOM[2]) is ready, a reason why it is not necessary to wait until all the binary content is fully loaded, like images or banner adds. Another remarkable feature is that each user and each visited page generate a single log in XML format. In this way, postprocessing tasks are notably facilitated, since most web programming languages have APIs to use documents in this format. Thus, it is possible to generate statistics from the stored data, which combined with any other analysis tool, like Urchin or Google analytics, make (smt) an efficient web interactivity evaluation system.

Last but not least, the main goal of this application consists in analysing user behaviour and inferring a visitor profile. This is a stage that will have to be reached when there will be enough representative samples. As an order of magnitude, we recommend between 200 and 300 processed logs, given the possible anomalies that might appear, or alternatively between 20 or 30 logs per user. Although the latter would be the best case, in order to personalize a page based on each user's personal characteristics (for example: element position and distribution, text size, etc.), it is in fact practically impossible to achieve if it is not made through a user authentication system or in laboratory tests, being the latter a very restricted setting and not accurately extrapolable to real situations.

3. SYSTEM ARCHITECTURE

This is an open source client-server architecture tool for webmasters, developers and even designers. The suggested application can be basically divided into two stages: the recording stage and the visualization stage. In the first one, an application in the client browser is executed (a single Javascript file) and in the second one a server application (a PHP file) processes the data. Both archives run in a transparent mode for

[1] It is advisable to establish a recording timeout, since the mouse is constantly monitored until the user exits the page. Thus, occasions in which data processing is unapproachable may occur: for example if the user enters the website and leaves the room.

[2] The DOM is a platform and a language independent standard object model for representing HTML or XML and related formats, proposed by the World Wide Web Consortium (W3C).

the user. The core of (smt) for real time recording and visualization lies in the fact that the storage of mouse coordinates is carried out at time-regular intervals, and these are measured in frames per second. Just like working with vector graphics, our visualization tool implies minimum download time, since the only data to be downloaded is the same web page that was visited by the user along with his mouse movements.

3.1 Preparation

To begin using (smt), neither the client nor the server need to introduce any changes. Only PHP —99% of traditional web servers fulfill this requirement— is needed. The webmaster must include one single line of JavaScript code in those (X)HTML pages that he wishes to monitor, after the <body> tag. Those pages are only modified by this line of code, and, what is more important, the included file is cached by the browser, so it does not have to be reloaded every time the user loads a new monitored page.

A Javascript program can only access to the page in which it is inserted, according to the SandBox security model. In fact, for that reason those pages that will be monitored must be modified manually. This can be automatized, and without using a proxy server, but we have decided to do it this way due to a preliminary analysis of (smt) and focusing on its development in a more effective way. An efficient solution is to write a script in the server that looks for all HTML, PHP or ASP documents of the site recursively, for instance, inserting the needed line of Javascript code. Yet, at this stage, this is not the intention of our work.

3.2 Registration

Some users connect to the web server and visit one of the pages that has the (smt) system included. They perform any task and after a while they exit the page, by clicking a hyperlink to another document or website, or even by closing the browser window. The moment the user exits the page, the data are sent via AJAX to the server, through an asynchronous connection that does not need to refresh the current window. Therefore the user experience is not altered at all and he does not have the sensation of having been recorded.

Later in the server, a PHP file, on the one hand, creates a copy of the visited page, including the mouse tracking layer, and, on the other hand, a log in XML format is generated with the most relevant data: URL, page title, client browser and operating system, screen resolution, the window size in which the user visited the web page, tracking time and, of course, the mouse coordinates and clicks made in every moment, which are stored both in individual arrays.

3.3 Visualization

The webmaster goes to the HTML log folder, in which exact copies of the original HTML pages can be found, including a registration layer with the data corresponding to each visit. When opening any of these HTML files with his web browser, the user's mouse path is exactly replayed until he exits the page. We have decided to generate a copy of the original HTML files with the mouse tracking layer so that they can be consulted both online and offline, for later studies. Yet, we do not rule out the idea of using a MySQL database in the future. The HTML logs are made in the server at the same time in which the user leaves the page, so movements revisions are available from that same moment.

3.4 Inference

Once the page-related data are collected, the corresponding statistics are generated, providing two types of data: concrete and abstract. The first ones are those that are stored in the XML logs. They show detailed information obtained from the data. For example, how many times and where the mouse was clicked, in which moment it was clicked, if a text was highlighted, the total visit time, etc. The second type includes data that inform about the user's skills, how he uses the web interface, if he is an impatient person, etc. With these data and enough representative samples it is possible to infer user behaviour in each page, creating therefore a customized user profile or one common profile for all of them.

4. RESULTS ANALYSIS

This system runs with no problems in any (X)HTML page. The fact is that nowadays a lot of webs exist with diverse embedded content: Java applets, QuickTime or RealAudio movies, flash animations, etc. Given the nature of these objects, to allow mouse tracking on them it is necessary to include the registration algorithm in the object itself, which is not possible in some cases, as it depends on the content player.

In Figure 1 three different levels of registration precision are shown, carried out simultaneously on the same page. It is remarked that the accuracy level is variable and totally configurable, measured in frames per second (fps).

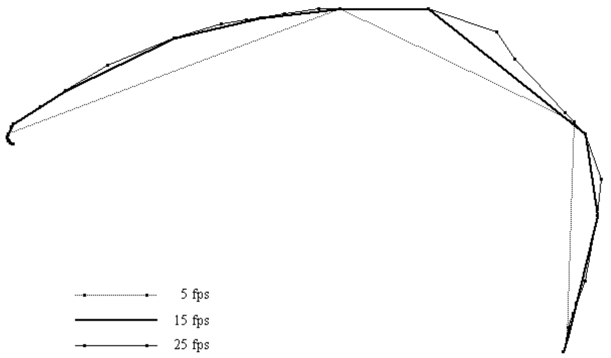

Figure 1. Different registration accuracy for the same mouse movement

Table 1. Number of mouse samples (coordinates) and overall size of XML logs according to Figure 1. That mouse movement was completed in 1.562 seconds

Frames per second	Number of samples	Size of XML log (bytes)
5	7	359
15	23	550
25	39	737
35	53	903

This is a sample XML log generated by (smt), as colophon:

```
<?xml version="1.0" encoding="UTF-8"?>
<log>
<url>http://localhost/smt/google_record.htm</url>
<title>Mozilla Firefox Start Page</title>
<client browser="Firefox" version="2" os="Windows"></client>
<screen width="1280" height="800"></screen>
<document width="1280" height="669"></document>
<trackingTime>18.922</trackingTime>
<fps>24</fps>
<coordsX><![CDATA[283,272,238,266,299,325,316,...,641,639,637,633,629,626]]></coordsX>
<coordsY><![CDATA[145,174,310,389,394,360,330,...,353,367,363,360,355,353]]></coordsY>
<clicksX><![CDATA[0,0,0,0,0,0,0,0,0,0,0,0,0,0,...,0,0,0,0,625,625]]></clicksX>
<clicksY><![CDATA[0,0,0,0,0,0,0,0,0,0,0,0,0,0,...,0,0,0,0,353,353]]></clicksY>
<clicksTime><![CDATA[1806]]></clicksTime>
</log>
```

A directly observable result is that experienced users (because they know the website, or because of their computer knowledge) show more uniform and rectilinear trajectories, with sparse registration points. The use of the scroll bars can be easily appreciated, since the mouse is located in the right margin of the window (in Latin writing, of course) and the trajectory is practically rectilinear. If the mouse wheel is used, the trajectory is a perfectly straight line. Another easily noticeable result is that of text highlighting: the button of the mouse is pressed and it is released when the text highlighting has finalized.

5. CONCLUSION

In this paper a system for real time mouse tracking analysis on websites has been presented, by applying it through a visual tool for later replaying.

This system is an added value to evaluate both website interactivity and user experience, without considering whether it is lawful to register user movements or not. It is up to the website administrator to inform his visitors that their mouse movements will be recorded, just as it is up to him to inform them about the later data postprocessing. This tool has been created as a help tool, not as a misusing instrument. For that reason the registration is made anonymously, totally transparent and imperceptible for the user and, of course, without interfering with other page scripts.

The main disadvantage of this system is that a Javascript code needs to be executed in the client browser, and there is a current minority of visitors who surf the Internet with a disabled Javascript console. However, they represent a too low percentage as to refuse the viability of this system, being most of these visitors the search spiders that use search engines to index web pages.

A smaller disadvantage is that, since the visualization stage is directly proportional to the recording speed, in long time visits the task of replaying all mouse movements can be exasperating, above all if these were made with no apparent intention (for example, playing around with the page). For that reason, a configurable registering timeout is set. It is possible to determine these stochastic situations by modelling behaviour profiles, which is one of the main later objectives.

And this is only the starting point. There are as many fields of use as those ones that imagination can create: self-adapting websites based on navigation habits, user gesture recognition, contextual help, real time technical support for learning communities, or even mouse tracking data as web services for data mining, to cite only some of them.

ACKNOWLEDGEMENTS

The visualization tool makes use of the High Performance JavaScript Vector Graphics Library developed by Walter Zorn in 2006.

There is a permanent URL where the (smt) tool can be downloaded and also be commented: http://smt.speedzinemedia.com.

REFERENCES

1. Arroyo, E. et al, 2006, Teaching User Interface Design using a Web-based Usability Tool. *Proceedings of the Conference on Human Factors in Computing Systems (CHI) 2006.* Montreal, Canada, pp. 2-4.

2. Atterer, R. et al, 2006, Knowing the User's Every Move -- User Activity Tracking for Website Usability Evaluation and Implicit Interaction. *Proceedings of The Fifteenth International World Wide Web Conference (WWW2006).* Edinburgh, Scotland, pp. 2-9.

3. Chen, M. et al, 2001, What Can a Mouse Cursor Tell Us More? Correlation of Eye/mouse Movements on Web Browsing. *Proceedings of Computer Human Interaction (CHI) 2001.* Seattle, USA, p. 280-283, 2001.

4. Goldberg, J. H. Et al, 2002, Eye Tracking in Web Search Tasks: Design Implications. *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA) 2002.* New Orleans, USA, pp. 51-58.

5. Mueller, F. and Lockerd, A., 2001, Cheese: Tracking Mouse Movement Activity on Websites, a Tool for User Modeling. *Proceedings of the Conference on Human Factors in Computing Systems (CHI) 2001, extended abstracts on Human Factors in Computing Systems.* Seattle, USA, pp. 1-2.

6. Nielsen, J., 1999. *Designing web usability: The Practice of Simplicity*. New Riders Publishing. Indianapolis, USA.

7. Nielsen, J., 2004. *Capturing Thoughts, Capturing Minds? - from Think Aloud to Participatory Analysis.* Working Paper No. 14-2004. Department of Informatics. Copenhagen, Denmark, pp. 2-5.

8. Norman, K. L. and Panizzi, E., 2006, Levels of Automation and User Participation in Usability Testing. *Interacting with Computers,* Vol. 18, No. 3, pp. 5.

9. Pan, B. et al., 2004, The Determinants of Web Page Viewing Behavior: An Eye Tracking Study. *Proceedings of Eye Tracking Research and Applications (ETRA).* San Antonio, USA, pp. 3-6.