

## Web Browsing Behavior Analysis and Interactive Hypervideo

LUIS A. LEIVA and ROBERTO VIVÓ, Universitat Politècnica de València

Processing data on any sort of user interaction is well known to be cumbersome and mostly time consuming. In order to assist researchers in easily inspecting fine-grained browsing data, current tools usually display user interactions as mouse cursor tracks, a video-like visualization scheme. However, to date, traditional online video inspection has not explored the full capabilities of hypermedia and interactive techniques. In response to this need, we have developed *smt2e*, a Web-based tracking system for analyzing browsing behavior using feature-rich hypervideo visualizations. We compare our system to related work in academia and the industry, showing that ours features unprecedented visualization capabilities. We also show that *smt2e* efficiently captures browsing data and is perceived by users to be both helpful and usable. A series of prediction experiments illustrate that raw cursor data are accessible and can be easily handled, providing evidence that the data can be used to construct and verify research hypotheses. Considering its limitations, it is our hope that *smt2e* will assist researchers, usability practitioners, and other professionals interested in understanding how users browse the Web.

Categories and Subject Descriptors: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Video; Evaluation/methodology; H.5.3 [Information Interfaces and Presentation]: Group and Organization Interfaces—Web-based interaction; H.5.4 [Information Interfaces and Presentation]: Hypertext/Hypermedia—Navigation

General Terms: Design, Performance, Human Factors

Additional Key Words and Phrases: User interaction, cursor behavior, hypervideo synthesis, InfoVis, interactive analysis, usability evaluation

### ACM Reference Format:

Leiva, L. A. and Vivó, R. 2013. Web browsing behavior analysis and interactive hypervideo. *ACM Trans. Web* 7, 4, Article 20 (October 2013), 28 pages.

DOI: <http://dx.doi.org/10.1145/2529995.2529996>

## 1. INTRODUCTION

Investigating how User Interfaces (UIs) are used has historically aroused a lot of interest in many research fields, such as product design and software engineering, for example, detecting areas of interest or misused layout spaces, time to complete a task, etc. To this end, video analysis has been considered a key component in multidisciplinary fields, like Human-Computer Interaction (HCI) and Human-Centered Multimedia (HCM). It is important for practitioners to record what was observed in addition to why such behavior occurred and to modify the application accordingly, if so desired.

This is a substantially extended version of a previously published conference paper [Leiva and Vivó 2012]. This work was partially supported by the MIPRCV Consolider Ingenio 2010 program (CSD2007-00018) and the TIN2009-14103-C03-03 project. It is also supported by the 7th Framework Program of the European Commission (FP7/2007-13) under grant agreement No. 287576 (CasMaCat).

Authors' address: L. A. Leiva (corresponding author) and R. Vivó, Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València; Camí de Vera, s/n – 46022 València, Spain; corresponding author's email: [llt@acm.org](mailto:llt@acm.org).

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or [permissions@acm.org](mailto:permissions@acm.org).

© 2013 ACM 1559-1131/2013/10-ART20 \$15.00

DOI: <http://dx.doi.org/10.1145/2529995.2529996>

Overall, observing the overt behavior of users has proved to be useful in investigating usability problems. For instance, based on live observations or analyses of video tapes, an evaluator can construct a problem list from the difficulties the users have accomplishing the tasks [Jacobsen et al. 1998]. Nonetheless, video data is time-consuming to process by human beings [Daniel and Chen 2003].

Analyzing video has traditionally involved a human-intensive procedure of recruiting users and observing their activity in a controlled lab environment. Therefore, several factors have led to the development of remote activity tracking for UI evaluation, for instance, (a) relevant data from real-world usage is required; (b) it is difficult to gather a representative user sample; (c) rapid prototyping requires sometimes preliminary studies, and time is a restriction; (d) recruiting users for a lab can be very costly, including equipment and resources; (e) a user's work context is difficult or impossible to reproduce in a laboratory setting; and (f) software applications usually have a life cycle extending well beyond the first release.

Processing user interaction data is therefore, at the very least, cumbersome. Moreover, if we move into the environment of the Web, their analysis is at best fairly limited and at worst virtually impossible. At present, the Web has grown massively in size, popularity, applications, devices, and number of users. It is so popular that usability in this environment has more impact on society than some other environments [Chen et al. 2001]. Unfortunately, to date, most theories on browsing behavior are based on studying patterns from server access logs. Such patterns consist of click paths or query logs, that is, sequences of browsed URLs. This may be enough for quantifying some observations, such as analyzing the so-called conversion rates and detecting whether the website has succeeded or failed in reaching their goals. However, the context of actions is of utmost importance in understanding browsing usage. Thus, when dealing with finer-grained exploration of human interaction on the Web, analytics based on server access logs alone are anything but accurate, making it necessary (in addition) to study the client side.

State-of-the-art user tracking systems use JavaScript logging tools, which fundamentally include mouse and (sometimes) keyboard tracking, since these input devices are ubiquitous. This way, specific hardware or special settings are not required to collect interaction data remotely. Furthermore, researchers have shown that the mouse cursor can inform about user intent and search interests, finding a strong correlation with how likely it is for a user to look at webpages [Chen et al. 2001; Mueller and Lockerd 2001; Huang et al. 2011]. Overall, remote activity tracking provides a series of interesting advantages when compared to classical usability tools. According to Arroyo et al. [2006], (1) it can be mass deployed, allowing for large datasets; (2) it is able to reach typical users and first-time visitors in their natural environment; (3) it can continuously test live sites, providing new insights as new website sections are deployed; and most importantly, (4) it is transparent to the users, so no experimenter bias or novelty effects are introduced, allowing users to browse websites as they would normally do.

Cursor tracking systems usually support replaying user interactions as *mouse tracks*, a video-like visualization scheme that allow researchers to easily inspect what is going on behind such interactions, for example, How many of the users considered clicking on the Buy button, and how many of them actually did click on it? In which order did the user fill in the form fields? Do users ever scroll down the web page? If so, how far exactly? However, traditional online video inspection has not benefited from the full capabilities of hypermedia and interactive techniques. We believe that these capabilities can better assist both novice and expert usability practitioners. On the one hand, hypermedia systems have long been shown to be engaging tools that make complex information understandable, handling images and motion as representational aids [Shiffer 1995].

On the other hand, interactive visualizations help understand human behavior in both qualitative and quantitative terms (e.g., characterizing changes in activity over time). Yet efficiently implementing this technology is a challenging task, as making video interactive and linking video and text has traditionally posed a series of general problems [Liestøl 1994].

It is our belief that hypervideo will gain notable importance in the field of Web tracking. Specifically, we believe practitioners and researchers—from here onwards we will refer to these figures as “the viewer”—can be presented with cursor data in a more engaging way. We also believe that, in addition, new ways of dealing with video-based cursor data demand new ways of interaction. With these ideas in mind, we have developed  $SMT2\epsilon$ , a general-purpose tracking tool for understanding Web browsing behavior.  $SMT2\epsilon$  is a significant continuation of our previous work,  $SMT2$  [Leiva and Vivó 2012], with a solid focus on better recording and hypervideo-based visualization capabilities.

### 1.1. Revisiting Hypervideo

*Hypervideo*, or hyperlinked video, is a video stream that contains user-clickable anchors, allowing navigation between video and other hypermedia elements [Smith and Stotts 2002]. As such, hypervideo offers the possibility of shifting from the linear order found in current cursor replay systems to a nonlinear and feature-rich visualization.

We redefine hypervideo as a composite video stream that contains embedded interactive elements, which is a more general definition that is compatible with the literature [Hirata et al. 1993; Francisco-Revilla 1998; Smith and Stotts 2002]. What differentiates  $SMT2\epsilon$  from other cursor replay systems, and from classical hypervideo systems, are the following manipulation capabilities.

- Different visualization choices can be made based on a series of informative layers that are rendered at runtime. The content of hypervideos is thus modified according to the viewer’s interests in real time.
- Multiple browsing sessions can be combined in a nonlinear structure, thus allowing the viewer to save time.
- Hypervideos can be linked to specific frames using timecode format (hh:mm:ss), enabling a specific time segment to be shared with friends or coworkers, for example.
- Hypervideos do support HTML-based annotations, which allows the viewer to add content with pointers to time segments, for example, to mark interesting parts for later review or even link to other hypervideos.
- Interaction profiles are generated as soon as new users access the website. Therefore, new data are available to analyze immediately upon installing the system.

In addition,  $SMT2\epsilon$  can be further augmented with JavaScript plugins and PHP extensions, so, as it is Web based, additional functionality can easily be implemented. Finally, it is worth pointing out that this tool is released as open-source software, which encourages collaboration, code fixes, bug reports, and other facilities that are not available in proprietary systems.

### 1.2. Organization

This article is organized as follows. First, in Section 2 we review previous work that relates most to our system and provide an overview of  $SMT2$  and how  $SMT2\epsilon$  differs from it. Next, in Section 3 we examine a series of questions about usage that can be answered by visualizing mouse cursor replays. The relevant parts of our system are described comprehensively in Section 4. Further on, empirical evaluations are carried out in Section 5, which mainly comprise a comparison of our tool against a commercial system under two studies (logging performance and usability evaluation). In Section 6

we illustrate how the captured cursor data can be used to construct and verify research hypotheses. We also provide a general discussion in Section 7 and comment upon the limitations of our system in Section 8. Finally, we end the article with a series of concluding remarks and opportunities for future work.

## 2. RELATED WORK

The utility of user tracking systems is evident throughout the research literature. For instance, mouse cursor data have recently been used to conduct studies on eye-mouse coordination [Rodden et al. 2008; Huang et al. 2011, 2012], Web search [Agichtein et al. 2006; Guo and Agichtein 2010a, 2012], reading behavior [Guo and Agichtein 2010b; Hauger and Velsen 2009; Hauger et al. 2011], and user modeling [Leiva and Vidal 2010; Leiva 2011; Buscher et al. 2012]. Yet, to date, the vast majority of research studies are conducted using ad-hoc tracking scripts. Therefore, a unified system for capturing and manipulating interaction data would seem to be of interest.

In this section, given the focus of the article, we relate to remote cursor tracking systems from the industry and academia. Note that earlier approaches involved installing software on the client side [Reeder et al. 2001; Tarasewich et al. 2005]. Here we comment instead on approaches that do not need to do so, since this is the way most systems operate at present.

### 2.1. Web Tracking Systems in Research

We found three works [Mueller and Lockerd 2001; Arroyo et al. 2006; Atterer et al. 2006] to be the contributors among the research community most relevant to our work. Their systems, though they currently no longer seem to be maintained, have set a precedent in Web-based user tracking applications, both in academia and the industry. Although not stated, Mueller and Lockerd [2001] presumably collected cursor movements in the same way as Arroyo et al. [2006] did, that is, storing in a database  $\{t, x, y\}$  tuples (timestamp and coordinates) whenever the mouse moves out of an  $R$  px circle radius. Arroyo et al. [2006] introduced the concept of collaborative filtering (i.e., working with aggregated users' data) and the idea of using a Web-based proxy to track external websites. Finally, Atterer et al. [2006] developed USAProxy, an advanced HTTP proxy that tracked the user's every move, including keystroke data, when a periodical scroll or a cursor event was captured. Interaction data were saved in a text format akin to the Apache access logs<sup>1</sup>, and the system was able to map cursor coordinates to Document Object Model (DOM) elements. While Mueller and Lockerd [2001] and Arroyo et al. [2006], overlaid an image on top of the webpages to represent the tracked interaction data, in USAProxy, it appeared feasible to replay a particular user's actions, though visualization was not the primary focus of the system. We argue that incorporating the temporal information may enhance understanding of human interaction, to replay exactly how users interact on a website. To this end, this is where video capabilities come into play, which, to some extent, have recently been implemented in industry systems.

### 2.2. Web Tracking Systems in Industry

We inspected the (minified) source code of most popular commercial Web tracking systems, which was possible by using their freemium versions—luckily, most of them offer a limited but functional service free of charge via email subscription. Among the available candidates, we found different approaches to registering and visualizing user interaction. Therefore, we begin highlighting the main differences while recording. Then, we compare each candidate with ours in terms of visualization.

<sup>1</sup><http://httpd.apache.org/docs/current/logs.html>.

Basically, commercial systems work as “hosted solutions”, that is, a software-as-a-service delivery model. These systems require the webmaster to insert a tracking script into the pages being targeted. Then, the tracking script submits the data back to the commercial server(s). Eventually, registered users can review the tracking logs at an administration area or admin site provided by the commercial system.

On the one hand, `CLICKTALE`<sup>2</sup> is being very actively developed and used at present and is heavily oriented to Web analytics. Among other setup instructions, it requires the viewer to establish a recording ratio to determine the percentage of visitors that will be selected. We use random sampling instead, that is, assigning equal probability of selection for every user. Nevertheless, we left this feature to the webmaster’s discretion, letting the latter establish any desired sampling strategy. To enable cross-domain communication, `CLICKTALE` uses the strategy depicted by Atterer et al. [2006], that is, requesting an image having a query string with the tracked data in the `src` attribute via Ajax. `USERFLY`<sup>3</sup> and `MOUSEFLOW`<sup>4</sup> are also popular cursor tracking tools, both built on top of jQuery (a general-purpose framework for JavaScript development). `USERFLY` encodes tracking data in a JSON string, while `MOUSEFLOW` uses base64url format. Then, data are submitted in the same way as with `CLICKTALE`. We set out next our approach for submitting tracking data and how it differs from these systems.

On the other hand, `MPATHY`<sup>5</sup> uses Flash socket connections and submits data whenever a browser event is registered. `CLIXPY`<sup>6</sup> creates a log file in a very similar as does way to Atterer et al. [2006], the difference being that the user IP is explicitly removed. To support cross-domain requests, this system proxies tracking data to a Flash object each three seconds. In both cases, relying on the Flash plugins on the client side poses a fundamental problem, since some users do not have such plugins installed, either because it is not supported on their operating systems (e.g., iOS) or simply because they do not want it, so it would not be possible to track these users at all. Therefore, depending on the target audience of the website, this could lead to us missing a huge share of the visitors who would otherwise have provided valuable insights about their browsing experience.

As for visualization, the conceptual idea used in our tool is similar to that of the majority of the previously mentioned commercial systems: rendering logged data events on top of an HTML page. However, besides providing limited support in terms of visualization capabilities, current systems only allow a single user session to be reproduced at a time. This can be extremely time consuming, depending on the number and duration of the visits. For this reason, we let the viewer simultaneously merge as many interaction logs as they like. Additionally, `SMT2ε` allows information to be displayed and modified at runtime, which is intended to make it easier to test different visualization strategies. Furthermore, `SMT2ε` provides the viewer with actual hypervideos, enabling annotations and links to specific parts of the visualization.

Finally, there exist other approaches for visualizing user activity, such as DOM-only based (e.g., the `<TAG> tracker`<sup>7</sup> system), or heatmap-based ones. The former lately has been gaining support in general-purpose Web analytics software. The latter, on the other hand, is implemented to some extent by most of the previously mentioned commercial tracking systems. Our tool supports both types of visualization, but the way we process the data is, again, more focused on infographics, as opposed to Web analytics.

<sup>2</sup><http://clicktale.com>.

<sup>3</sup>[http://userfly.com\(nowdefunct\)](http://userfly.com(nowdefunct)).

<sup>4</sup><http://mouseflow.com>.

<sup>5</sup><http://m-pathy.com>.

<sup>6</sup><http://clixpy.com>.

<sup>7</sup><http://otterplus.com/mps>.

For instance, our heatmap visualization strategy is dynamic, allowing heatmaps to be generated at runtime at the same time that the viewer is watching the hypervideos. Specifically, the resulting heatmap follows the ‘shadowmaps’ implementation detailed by Špakov and Miniotas [2007].

Besides the technical comparisons described next, analyzing interaction data like current tracking systems do can be rather limited, as the viewer is able to use at best play/pause as playback controls while replaying a user session. Now we describe SMT2, our previous work, and show how it differs from current systems. Then, we introduce SMT2 $\epsilon$  and describe how it differs specifically from SMT2.

### 2.3. SMT2 Overview

First, one important feature of our previous work as regards to state-of-the-art Web tracking systems is the ability to composite multiple interaction logs into a single hypervideo. This feature is not only useful for qualitatively assessing the usability of websites but also for discovering common usage patterns by simply inspecting the visualizations (see Section 3).

Second, another important feature of SMT2 is the generation of user and page models based on automatic analysis of collected logs. In this regard, we have not found any related tracking system that would perform implicit feature extraction from the users’ interaction data, that is, interaction features inherently encoded in cursor trajectories. We believe this is a promising line of research currently gaining attention from other authors (e.g., Guo and Agichtein [2010a]; Huang et al. [2011]).

Third, the recording approach used in SMT2 is different compared to the ones described in Section 2.2. Specifically, it performs a discretization of user interactions over time, following a simple event logging strategy and the *polling* technique, that is, taking a snapshot of the cursor status (mainly coordinates, clicks, and interacted elements) at a regular interval rate. This way, user actions are recorded exactly as they were performed. This also allows the speed at which movies can be replayed to be changed, which is helpful for normalizing trajectories that were acquired at different sampling rates when compositing a multitrack hypervideo.

### 2.4. Introducing SMT2 $\epsilon$

Regarding tracking capabilities, SMT2 $\epsilon$  behaves almost identically to its predecessor, with the notable exception that SMT2 $\epsilon$  features LZW compression to transmit logged data, thus saving bandwidth. This capability is evaluated in Section 5.2.1. The actual improvements made to SMT2 that eventually led to SMT2 $\epsilon$  concentrate on the server side.

First of all, our current work is geared towards interactive hypervideo synthesis from user browsing behavior. However, unlike conventional hypervideo, SMT2 $\epsilon$  is aimed at building full interactive movies from remotely-logged data. Furthermore, current hypervideo technology itself is limited to clickable anchors [Smith and Stotts 2002]. SMT2 augmented this technology with interactive *infographics*, that is, a series of information visualization layers that are rendered at runtime and which provide the viewer with additional information. For instance, hovering over a click mark displays a tooltip indicating the cursor coordinates, or hovering over a hesitation mark displays the amount of time the cursor was motionless. SMT2 $\epsilon$  extends this hypervideo-based technology with (1) *hyperfragments*: visualizations can be linked to specific start/end video parts, and (2) *hypernotes*: HTML-based annotations that point to specific video parts.

We believe these new improvements are convenient in a cursor tracking visualization system for a series of reasons. First, hyperfragments allow the viewer to select a slice

of the video that may be of particular interest. Hyperfragments can be specified with either a starting or an ending timecode. When doing so, the video length is trimmed to such a specified duration. This lets viewers quickly access the desired information without having to review the entire replay. Second, hypernotes allow the viewer to comment on the visualizations at a specific point in time, for example, to annotate some details about the video or to let coworkers know that the video has been reviewed. When a hypernote is created, the viewer can click on a note icon on the timeline that will seek the replay to the creation time of the hypernote. This provides the viewer with indexing capabilities that could be extended to content searching. Moreover, hypernotes are HTML-based, which enables rich-formatted text and insertion of images and links (for instance, to point to other hypervideos). This capability opens a new door to how visualizations can later be processed, for example, it would be feasible to summarize a user session in text format.

In addition, SMT2 $\epsilon$  features two installation modes: an all-in-one solution (when the website and the admin site are both located in the same server) and a hosted service (website and admin site are located in different servers). SMT2 was limited in this regard, because in order to allow cross-domain communication, every website would require at least PHP support to forward the requests to the storage server (at the admin site). With SMT2 $\epsilon$ , however, the only requirement for a website to be tracked is inserting a single line of JavaScript code, as other commercial systems do, so potentially any website could benefit from it.

Finally, SMT2 $\epsilon$  allows the viewer to classify pages according to user behavior (which is known as *behavioral clustering*) as new users access the website, by automatically mining cursor features from the database logs. This functionality was included because the viewer may find it useful to discover common interaction profiles as well as to easily identify outliers.

### 3. APPLICATIONS

To begin with, we provide a systematic examination of some questions about usage that can be answered by visualizing mouse cursor replays. As hinted later, currently, most of these questions can only be answered by using our tool. Yet, in Section 3.2, we list a series of scenarios where any user tracking system may be suitable.

#### 3.1. Decoding Browsing Behaviors

—*Where do users hesitate? How much?* We followed the notion of *dwell time* introduced by Müller-Tomfelde [2007], also named *idle time* [Buscher et al. 2012], which is essentially the time span that people remain nearly motionless while pointing at objects. Cursor dwell times are usually associated with ambiguous states of mind [Arroyo et al. 2006], possibly due to a thinking or cognitive learning process. In SMT2 $\epsilon$ , dwell times are displayed as circles with a radius proportional to the time for which the cursor does not move (Figure 1(a)). The system takes care not to display extremely large values of dwell times by limiting the circle radii to a quarter of the viewport size. Currently, dwell times are handled similarly in other cursor tracking tools, such as CLICKTALE.

—*Do users select or highlight content? How?* Selecting/highlighting is encoded at a lower level as a drag and drop operation to support a more general visualization. For instance, a Web application can support rearranging widgets to customize their layout or adding objects to a list to be processed (e.g., a shopping cart). SMT2 $\epsilon$  uses a specific format to encode cursor coordinates, borrowed from the handwriting recognition community, where the status of the click button is attached to each cursor coordinate. Therefore, SMT2 $\epsilon$  provides a specific visualization type for these cases (Figure 1(b)). To our knowledge, this capability is only provided by our tool.



Fig. 1. Combining visualization possibilities. (a) Displaying hesitations (circles) and clicks (small crosses). (b) Displaying entry/exit coordinates (cursor bitmaps), motion centroids (crosses), highlighting activity (shaded fog), and interacted DOM elements (numbered rectangles). (c) Analyzing a decision process; the user rearranged items in a list. Small circles represent cursor dwell times. Hovered DOM elements are labeled based on frequency (percentage of browsing time), including a blue color gradient (100% blue: most hovered items). The same scheme is used to analyze clicked items but using red color.

- What element is the user interacting with?* Thanks to the bubbling phase of JavaScript events, whenever a cursor event is dispatched (e.g., mouseover, touchmove) the tracking script traverses the DOM hierarchy to find if there is an element that relates to the event. Each tracking log stores a list of interacted DOM elements sorted by time frequency (Figure 1(c)), including hovered and clicked/tapped elements. Thus, such a list could be inspected either quantitatively (by looking at the numbers) or qualitatively (by looking at the colors). It is worth mentioning that most Web analytics applications only provide click information, since this information can be derived from server logs alone. Therefore, this visualization could be helpful to answer in-page browsing questions, such as if the users go straight to the content or whether the cursor hovered over a link without clicking.
- Which areas of the page do most of the interaction concentrate on?* To answer this question, a K-means clustering of the coordinates is performed each time a mouse track ends. So, focusing on the clustered areas allows us to visually notice where users are performing most of their actions. Each cluster is represented by a circle with a radius proportional to the cluster population. This visualization layer is notably appropriate when tracking data are rendered as a static image. To date, SMT2 $\epsilon$  is the only system that features this capability.
- Do different mouse tracks correlate?* The viewer can select the ‘time charts’ option from the control panel and visually compare multiple trails (Figure 2(a)). For instance, dwell times will be plotted as horizontal lines, and scrolling will be plotted as near-vertical lines. Each tick in the  $x$ -axis corresponds to the registration frequency used while tracking (e.g., for 24 Hz, each tick would be 1/24 seconds). Cursor trails are normalized in width and height, to avoid potential visualization biases that might be caused by aggregating multiple trails of different length. Optionally, SMT2 $\epsilon$  can display the average cursor trail for a given group of visitors. This feature may be of special interest when people behave similarly in order to summarize a group of nearly-similar trails; all in all, it provides a general gist of how users browsed a page. These visualizations are also exclusively offered by SMT2 $\epsilon$ .
- What is the persistence of the page?* It is commonly agreed that, to some extent, page relevance is correlated to the time spent on browsing it. In this case, a 3D visualization (Figure 2(b)) may be useful for unveiling these observations. The 3D chart plots the temporal evolution of the cursor coordinates along the  $z$ -axis and provides simple interactive controls to make further inspection easier. This way, it



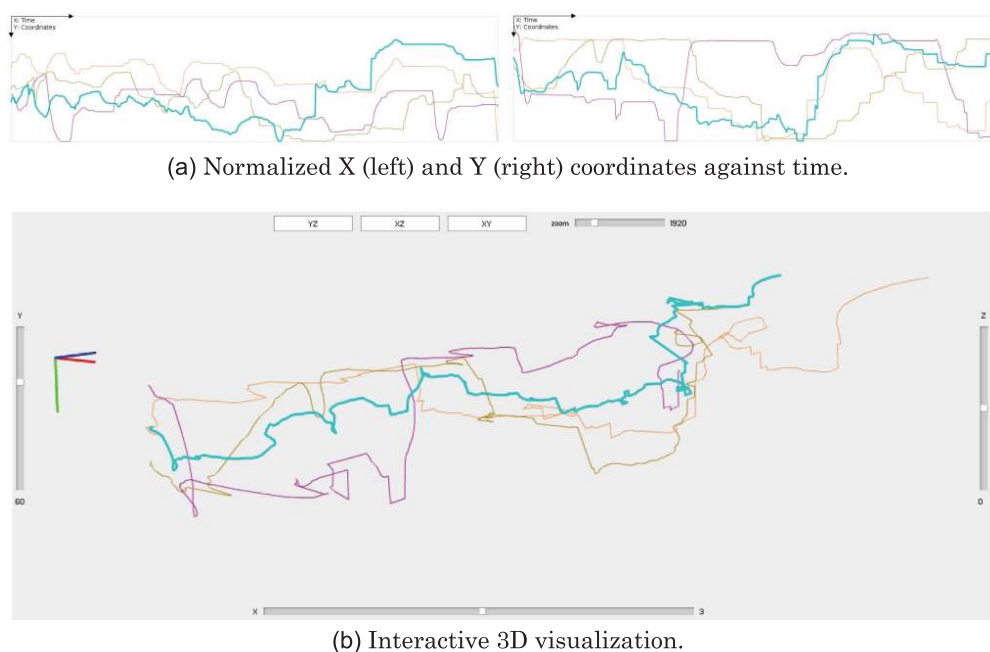


Fig. 2. Time charts visualization. The bold line is the averaged cursor trail, which is an optional visualization feature that takes into account the aggregated logs. In this example, there are three visits, and browsing times can be visually compared at a glance. The 3D view allows for rotating the axes with three sliders (one per axis), zooming, and projecting the lines in the YZ, XZ, and XY planes.

may be interesting to aggregate all visitors for a given page and observe at a glance the length of the cursor trails or whether some visual patterns can be found. Contrary to the time charts visualization, in this case, mouse tracks are not normalized in size. To date, only SMT2 $\epsilon$  provides this type of visualization.

### 3.2. Envisioned Usage Scenarios

We believe that beyond interactive visualizations, the scope of cursor replay systems in general—SMT2 $\epsilon$  in particular—could go further. In short, some real-world usage scenarios where these systems could suitably fit include the following.

- Interaction Research*. In general terms, understanding human movement is fundamental for improving input devices and interaction techniques.
- Data Mining*. Cursor data samples could be obtained on a large scale, so one could readily perform prospective studies.
- User Modeling*. The data could be used to build behavioral models with meaningful conceptual interpretation, according to user interactions.
- Usability Testing*. For a long time, activity tracking has been a reliable source of information about user interaction behavior.
- Gesture Recognition*. Gathered cursor positions could be mapped to specific gestures, thus allowing specific browsing commands to be triggered on a given page.
- Performance Evaluation*. Compare motor skills or pointing abilities of different users (or groups of users) within a particular task on a Web-based UI.
- Usage Elicitation*. If we want to avoid possible biases while analyzing human usage patterns, then we must deal only with interaction data coming from real users.

20:10

L. A. Leiva and R. Vivó

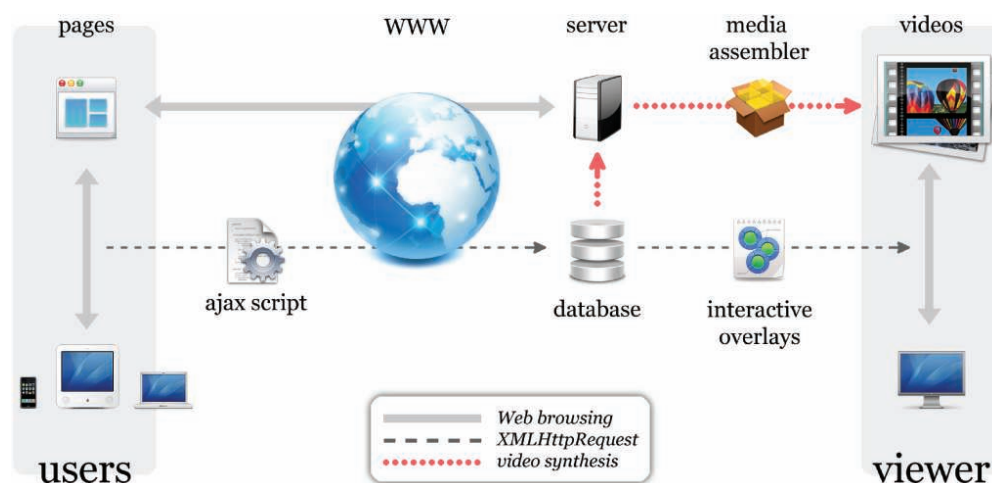


Fig. 3. System architecture for capturing user activity and synthesizing interactive hypervideos.

- Collaborative Filtering*. Unveil usage profiles and statistics among multiple viewpoints, data sources, and so on.
- Self-Adapting UIs*. Use interaction data to modify the appearance of the page elements based on each user's needs.

One may note that the applications depicted in the preceding list are generic enough to ensure a broad generalization scope.

#### 4. SYSTEM DESCRIPTION

SMT2 $\epsilon$  has been built using Web technologies and hence does not need to install additional software on the client side. The only requirement is a Web browser with JavaScript support, so any modern device capable of accessing the Internet (e.g., laptops, smartphones, tablets, etc.) could be a fair candidate to take part in a tracking campaign. As described next, SMT2 $\epsilon$  is composed of three fundamental parts: recording, management, and visualization. On the server side, any Web server supporting PHP and MySQL can run both the admin site and the visualization application.

##### 4.1. Architecture

This system uses the WWW infrastructure to log the user activity in a MySQL database (Figure 3). The technology for creating the hypervideos is a mixture of PHP (to interface with the database), HTML (to overlay the tracking data on top of it), JavaScript (to process the aforementioned tracking data), and ActionScript (to build the hypervideos and add interactivity).

##### 4.2. Logging User Interactions

While users are browsing as they would normally do, an Ajax script logs interaction events in the background and sends the data back to the server at fixed time intervals (Figure 4). Tracking is performed in a transparent way for the users, either silently or by asking their consent. The main captured events are summarized in Table I. Taking this small subset of browser events into account allows us to reduce the otherwise potentially huge amount of recorded interaction data.

To store the cursor coordinates, SMT2 $\epsilon$  makes use of a structured format inspired by UNIPEN [Guyon et al. 1994]—a popular scheme for handwriting data exchange

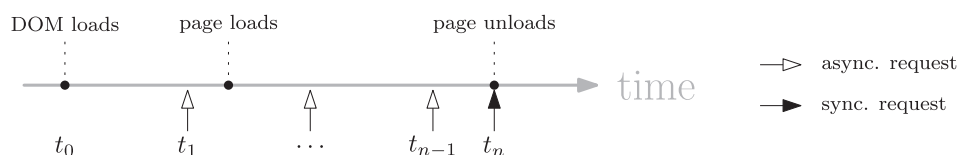


Fig. 4. Logging timeline. Tracking data are asynchronously submitted at regular time intervals, until the page is unloaded (where the submission is synchronous, for cross-browser support).

Table I. Captured DOM Events

document	window
mouse{down,up,move,over,out}	focus, blur
touch{start,end,move}	resize
key{down,up}	(dom)load, (before)unload

```

<script type="text/javascript">
smt2e.record({
  disabled: Math.round(Math.random()),
  warn: true,
  recTime: 180
});
</script>

```

Fig. 5. Tracking code example. We set random sampling for user selection and ask for consent of the chosen users for monitoring their browsing activity (they must agree to start recording), which will be tracked for three minutes at most.

and recognizer benchmarks. This way, it is possible to recompose the user activity in a fine-grained detail.

In order to reduce the data size to be transmitted, recording can be *continuous* (default behavior) or *intermittent* (i.e., tracking stops/resumes on blur/focus events), letting webmasters decide which operation mode is best suited to their needs. For instance, if an eye tracker is going to be used together with our system, then it is preferable to set continuous recording in order to keep mouse and gaze coordinate streams synchronized. On the other hand, if SMT2 $\epsilon$  is going to be used in standalone mode, then the webmaster may want to save storage space in the database by enabling intermittent recording. This will also result in less data being transmitted to the Web server. Additionally, as pointed out in Section 2.4, SMT2 $\epsilon$  features LZW compression on the logged data, which contributes to saving even more bandwidth.

On the other hand, it is possible to store interaction data from different domains into a single database, provided that each domain is under the webmaster's control. Otherwise, SMT2 $\epsilon$  can fetch external websites by using a PHP proxy, that is, users must start browsing from a dedicated proxy page, and the system will automatically insert the required tracking code (e.g., Figure 5).

### 4.3. Video Synthesis

The viewer indicates that the information that will be pulled from the database. For example, they might want to visualize a single browsing session (Figure 6(a)), in which case they would simply click on a play icon from the 'user logs' list (Figure 14 in the Online Appendix). Then, the system would retrieve the subsequent logs to replay all tracks sequentially. On the contrary, the viewer might want to filter out logs by operating system and browsed page, in which case she would use a 'mine results' form (Figure 14) that would merge the data into a single hypervideo (Figure 6(b)).

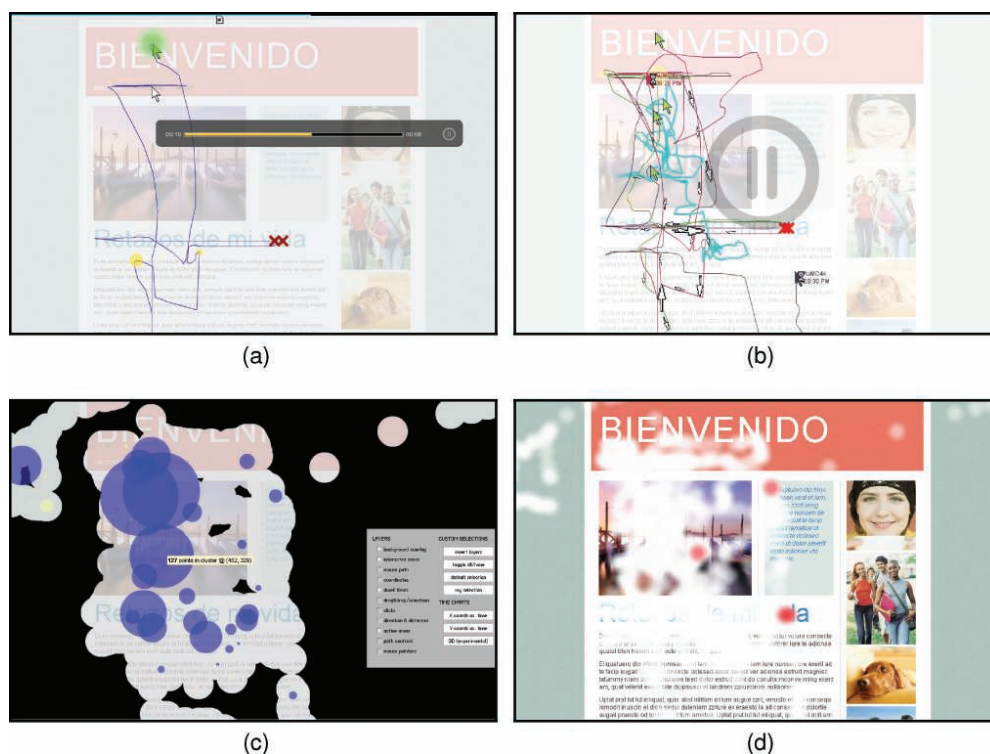


Fig. 6. Some stock examples of our hypervideo visualization tool. (a) Single session with embedded media player; hypernotes are attached to the timeline at the top. (b) Replaying cursor trails simultaneously, highlighting the average cursor trail (thick line) and overlaying direction arrows. (c) Clusters of mouse movements, displaying also masked areas of activity. (d) Dynamic heatmaps (shadowmaps) of mouse coordinates and clicks.

Retrieved data are encoded in JavaScript Object Notation (JSON), and they feed a precompiled video template. Cursor trajectories are normalized according to the original viewport of the user's browser and the current viewport of the viewer's browser, following a nonuniform affine mapping—either by scaling or translating the cursor coordinates—depending on the type of layout, namely, *left*, *center*, *right*, or *liquid*. A cached copy of the browsed page and the aforementioned interaction data are both bundled in a hypermedia player. This way, movies can be replayed on any Web browser.

#### 4.4. Accessing, Analyzing, and Interacting with the Data

The viewer can toggle different information layers interactively while they visualize the videos by means of a centralized control panel (Figure 7).

Furthermore, movies can be generated for individual users or by taking into account different kinds of segmentations, for example, time or date intervals, city locations, first-time visitors, and so on. For instance, the viewer can segment the tracking logs by user ID and determine which elements were interacted with the most or examine the percentage of scroll to infer interest, for example, if all users' browsed pages have minimum scroll reach, it may indicate that the user is searching unsuccessfully for a specific page. Figure 8 shows a screenshot of the analysis module.

Automatic analysis of interaction features is also feasible within the admin site, since collected data are readily available in the database. This way, besides explicit metadata that are assigned to content, implicit knowledge can help gain a better picture on the

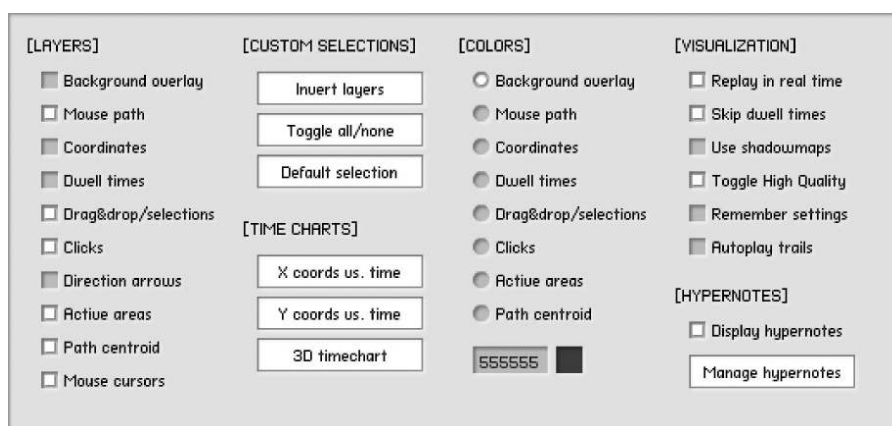


Fig. 7. A (draggable) control panel is the main link between the viewer and the synthesized hypervideos. One can manipulate different visualization possibilities which will be applied to the hypervideos at runtime.

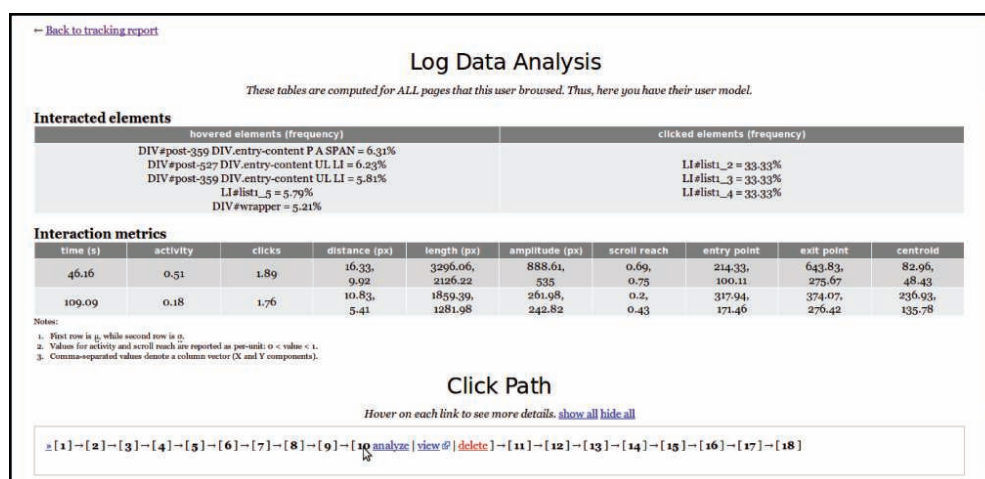


Fig. 8. A log analysis example, informing on all browsed pages from a particular user. The first table summarizes the most interacted DOM elements (hovered and clicked, respectively). The second table computes interaction features based on cursor activity (e.g., distances, entry/exit coordinates, etc.) In this table, the first row are mean values, and the second are standard deviations. Whether appropriate, values are indicated as a 2D vector, where dimensions are  $x$  and  $y$  components. The click path is the sequence of pages that the user has browsed. Each item in the click path allows the viewer to perform three actions over its corresponding log file: visualize, analyze, or delete.

nature of such content (see Section 5.3.1). Concretely, the features that  $\text{SMT}2\epsilon$  computes for a given tracking log are described as follows.

- Time*. Browsing time (in seconds) spent on the page.
- Activity*. Fraction of browsing time in which the cursor was moving, defined in  $[0, 1]$  (0: no movements at all, 1: always moving).
- Clicks*. Number of issued mouse clicks.
- Distance*. Averaged L2 distance (in px) between successive cursor positions.
- Length*. Accumulated sum (in px) of cursor distances.
- Range*. Difference (in px) between maximum and minimum coordinates.

20:14

L. A. Leiva and R. Vivó

—*Scroll Reach*. How far the cursor scrolled the page, defined in  $[0, 1]$  (0: no scroll at all, 1: scroll reached the bottom of the page).

—*Entry/Exit Points*. The first and last cursor coordinates, respectively.

—*Centroid*. Geometric center of all coordinates.

#### 4.5. Comments on an Early Prototype

We released a preliminary beta version of SMT2 three years ago at the GoogleCode repository, and overall, reaction to the system has been overwhelmingly positive. Since then, it has received successful feedback from researchers, developers, and practitioners worldwide.

Some users have suggested useful features, either related to the visualization module or to general usability guidelines. Other users have suggested enhancements to the admin site. Developers have confirmed that SMT2 $\epsilon$  could work with other programming environments, and have reported some modifications that met their needs. Finally, other users have asked for further applications of the tool (e.g., in marketing research or Web search tasks).

All in all, received feedback helped us fix existing bugs and refine the preliminary prototype. We have incorporated as many suggested enhancements as we could, and currently SMT2 $\epsilon$  is in a stable iteration. Some of the improvements included new visualization layers (e.g., dynamic heatmaps or the 3D visualization), a multi-user admin site with support for different roles, PHP proxy integration to track external websites, and automatic analysis of single/aggregated user and page models, among other revisited features. A detailed list of the source code changes is available at <http://code.google.com/p/smt2/source/list>.

## 5. EVALUATION

This section is an empirical assessment of what we believe are the key properties of SMT2 $\epsilon$ . In particular, we address the following questions.

- (1) *Logging Performance*. What is the network overhead incurred by tracking? What influence does frame rate have in this regard?
- (2) *System Usability*. Is the system easy to use? What is the perceived workload?

To show the value of these experiments, we focus on the benefits resulting from using our tool in contrast to commercial cursor tracking systems, among which we chose CLICKTALE because of their popularity and market share leadership. All experiments were performed on a Linux laptop with an i386 dual core processor @ 1 GHz and 2 GB of RAM.

### 5.1. Setup

We contacted one of the most renowned graphic design firms in Spain, Pepe Gimeno - Proyecto Gráfico. Their website is rather static; it only gets updated once a year. Designers organize the site in what they call “vintages”, or yearly collections containing their most notable works. Each vintage comprises a single page and displays a series of screenshots (Figure 9). When the site requires updating, webmasters from an outsourced Web design studio carry out the necessary changes.

Two of the outsourced webmasters were told to instrument the website with both the SMT2 $\epsilon$  and CLICKTALE systems at the same time, using the default settings. For the former, they were sent the SMT2 $\epsilon$  files in a ZIP archive, since our system was not publicly available at that time. For the latter, they subscribed a free plan at the CLICKTALE website (<http://clicktale.com>).

Users were tracked for approximately two months. Eventually, 2,084 visits (1,738 after outlier removal) were collected with SMT2 $\epsilon$  and 393 with CLICKTALE. (The maximum



Fig. 9. Homepage of the website tracked in this evaluation.

recording amount allowed by CLICKTALE's free plan is 200 visits per month.) A log was considered an outlier when browsing time exceeded 1.5 the value of the interquartile range (IQR). This technique [Tukey 1977] is very common in data mining and exploratory analysis, that is, removing samples from analysis that do not lie between  $Q1 - 1.5 IQR$  and  $Q3 + 1.5 IQR$ . In this way, we were able to deal with more consistent observations from the logs gathered. A quick review (using static visualizations) of some of the outliers did not reveal any particularly interesting behavior, such as whether users were confused or looking hard for what they need or looking at lots of detail on the page.

## 5.2. Performance Evaluation

*5.2.1. Performance Study 1.* The first experiment aimed to compare both systems in terms of logging performance. We reproduced the exact amount of bytes that were transmitted by SMT2 $\epsilon$  by retrieving the stored data from the tracking database and converting them to the 'application/x-www-form-urlencoded' format—which is the format that both SMT2 and SMT2 $\epsilon$  use to transmit the data over the network. We should remark that the only difference between SMT2 and SMT2 $\epsilon$  in terms of logging capabilities is that SMT2 $\epsilon$  features LZW compression.

At present, CLICKTALE transmits data in chunks of 1 kB via Ajax GET requests, perhaps to keep request URIs short and thus avoid 414 HTTP error codes.<sup>8</sup> Thus, logged data are buffered on the client side until reaching 1 kB, when the buffer is flushed and data are submitted to the server. Therefore, we compared SMT2 $\epsilon$  against this baseline of 1 kB, given our inability to manually modify CLICKTALE's logging scripts. The results of this study are shown in Table II.

In order to facilitate comparisons, the number of HTTP requests submitted by CLICKTALE was estimated according to the raw data size captured by SMT2, since CLICKTALE does not provide such detailed information. As observed in the table, the average browsing time reported by CLICKTALE and SMT2 $\epsilon$  is similar, which gave us a

<sup>8</sup><http://www.faqs.org/rfcs/rfc2616.html>.

Table II. Recording Performance of the Compared Systems Per Pageview

System	# Logs	Browsing Time (s)	Data Size* (kB)	# HTTP requests
SMT2 $\epsilon$	1,738	76.84 (103.1)	2.64 (2.7)	2.56 (5.9)
CLICKTALE	393	74	1	10.58 (23.1)

\*Data size transmitted in a single HTTP request.

Note: SDs, where available, are denoted in parentheses.

Table III. Network Overhead Incurred by Logging at Different Frame Rates

Hz	Browsing Time (s)	Uncompressed Log Size (kB)	Compressed Log Size (kB)
10	19.13 (10.7)	1.96 (1.0)	0.94 (0.4)
20	13.94 (10.6)	2.72 (1.9)	1.07 (0.5)
30	15.06 (11.4)	4.33 (3.1)	1.35 (0.7)
40	14.63 (14.3)	5.38 (4.1)	1.67 (1.0)
50	16.45 (10.6)	7.71 (4.9)	2.40 (1.5)

Note: SDs are denoted in parentheses.

hint that our estimates may be adequate. We conclude that the number of HTTP requests that CLICKTALE performs per pageview is significantly higher than that of SMT2 $\epsilon$  [ $t(2129) = -13.51, p < .0001$ ]. Furthermore, the effect size suggested moderate practical significance (Cohen's  $d = 0.71$ , Pearson's  $r = 0.33$ ). On the contrary, the data size that CLICKTALE transmits is less than half the size of that transmitted by SMT2 $\epsilon$ . Differences were also found to be statistically significant, with a moderate effect size [ $t(2129) = 11.70, p < .0001, d = 0.67, r = 0.31$ ]. A discussion about these observations is provided in Section 7. We must say that SMT2 $\epsilon$  features continuous recording by default, which was the setting used in this evaluation, so that tracking was enabled even when the browser window was not in focus (e.g., minimized in the background or occluded by a foreground application). Hence, if intermittent recording were set instead, perhaps much less data would have been transmitted.

**5.2.2. Performance Study 2.** To provide more insights about SMT2 $\epsilon$ 's logging performance, the following experiment aimed to compare it against its predecessor, SMT2. We instrumented the homepage at <http://smt.speedzinemedia.com>, and visitors were randomly tracked at different frame rates, namely 10, 20, . . . , 50 Hz. Eventually, 1,200 visits were collected, or 240 logs per each sampled frequency. In order to make this study comparable to the previous one, we used continuous recording mode.

We followed the same procedure to reproduce SMT2 $\epsilon$ 's transmitted data size, as described in the previous study. This time, users were tracked at most for 30 seconds, matching SMT2 $\epsilon$ 's default `postInterval` property. Thus, in all cases, the logging required exactly one HTTP request. Cursor data were stored on our server.

In Table III, the baseline case (uncompressed data) corresponds to SMT2, while compressed data corresponds to SMT2 $\epsilon$ . Together with Figure 10, it can be seen that SMT2 $\epsilon$  is more efficient than its predecessor for transmitting cursor data, achieving a compression ratio of around 1:2 for low frame rates (below 20 fps) and 1:5 for high frame rates (above 20 fps). In all cases, differences are statistically significant ( $p < .0001$ , two-tailed  $t$ -tests) with high effects sizes ( $d > 1, r > 0.5$ ).

Since only one HTTP request was required to log cursor data, if compared to CLICKTALE's logging approach, CLICKTALE would have been required to perform approximately 3.52 requests on average (SD = 1.1) to log the same data size. This estimation is made by interpolating in Table III the uncompressed data sizes corresponding to 20 and 30 Hz, since the default recording frame rate of SMT2 $\epsilon$  is 24 Hz. The reader can find a series of comments about logging performance in Section 7.



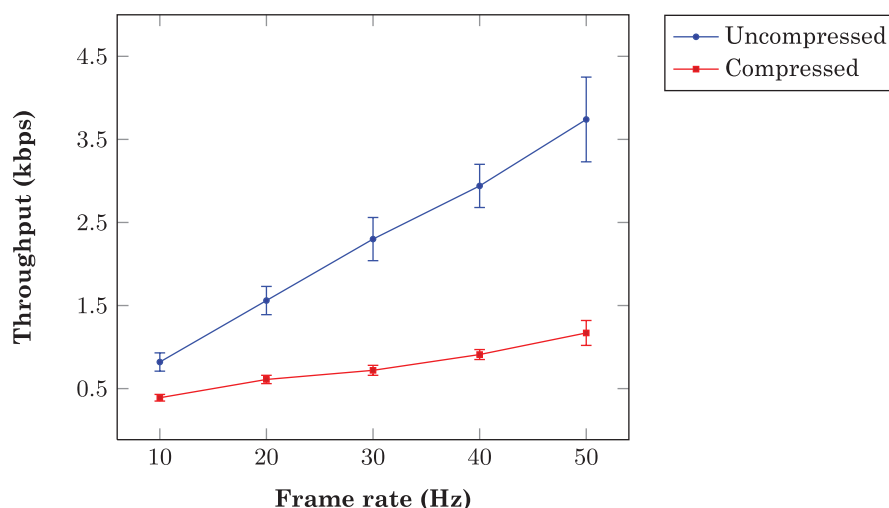


Fig. 10. Bandwidth required by SMT2 (uncompressed) and SMT2 $\epsilon$  (compressed) to transmit cursor data over the network. Error bars denote 95% confidence intervals.

### 5.3. Usability Evaluation

In this study, we return to comparing SMT2 $\epsilon$  and CLICKTALE, now in terms of usability. Ten professionals—five graphic designers and five webmasters—volunteered to participate. None was a usability expert or had previous experience in using Web tracking tools (they were only familiar with Google Analytics). After the first month of data gathering, participants used both tracking systems almost on a daily basis for approximately one month. Meanwhile, the website was still gathering data logs. Participants were interviewed before and after using both tracking systems.

**5.3.1. Case Study Overview.** Overall, participants found both systems very helpful. The main advantage suggested was in being able to exactly reproduce what users did on a webpage. Anecdotally, participants seemed to be more technically inclined to use SMT2 $\epsilon$ , mainly because of the customizations it could support and its openness. However, they complained about the lack of more detailed documentation in SMT2 $\epsilon$ . In this regard, they appreciated CLICKTALE's help desk and online support.

Regarding visualization possibilities, SMT2 $\epsilon$  was perceived as being more complete but also more difficult to start with. Some participants reported that this was in part because of the aforementioned lack of detailed documentation. Other participants suggested that this could be a consequence of having integrated all the visualization possibilities in a single section at the admin site (see the Online Appendix). As a result, they reported that CLICKTALE has clear organization and hence the learning curve may be smoother. Nevertheless, a common complaint regarding Clicktale was that visualizations are displayed inside an `iframe` and scroll bars appear quite often. This was perceived to be especially irritating by the interviewees.

Everybody liked the option of being able to switch SMT2 $\epsilon$ 's hypervideos to a static representation by pressing a single key stroke, specially when working with a large number of aggregated logs. Moreover, watching replays of simultaneous logs was perceived as a particularly time-saving feature. In fact, they did expect that CLICKTALE's replays would have provided some kind of aggregation facilities and were surprised at not being able to interact with the logged data in this way.

Participants also found the possibility of adding annotations to hypervideos very helpful, although in the end, we observed an anecdotal use of this feature. An

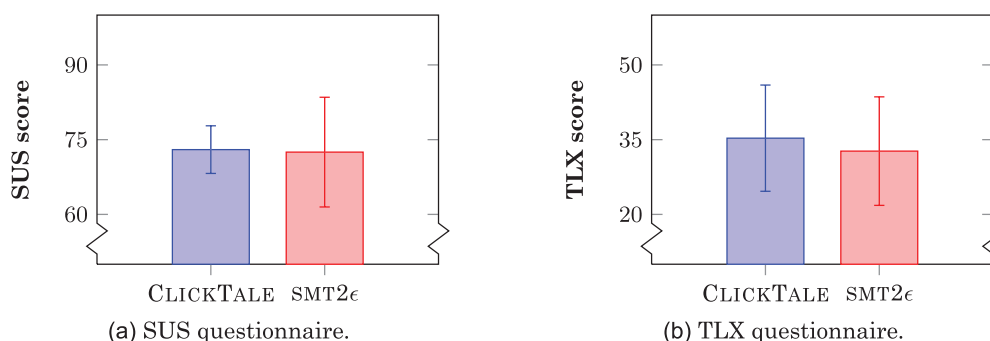


Fig. 11. Overall results of the subjectivity study. Error bars denote 95% confidence intervals.

interesting comment by one participant was that he created empty hypernotes to mark those videos he found worth revisiting. Everyone saw the value of hyperfragments, although they reported that hyperfragments would be more apt for large design teams to make the most of their potential. For instance, one participant mentioned that it would be helpful to share a link of a video that isolates a particular browsing behavior, like reading or scrolling.

Participants observed that people typically do not click while browsing the website beyond those clicks required to navigate to other pages. In this regard, besides not having any experience in usability or Web interaction, designers felt that this happened because the links are not very prominent and hence they might go unnoticed most of the time. Since pages were allocated a couple of minutes on average (see Table II), they hypothesized that the lack of clicking had no association with poor content structuring.

The observations thrown up by both tracking systems provided designers with more detailed thoughts on the website. They set out targets for the next year that would not have been thought of if they had not tested both systems. For instance, they decided to add more interactivity to vintages, and perhaps make images smaller to keep pages short in terms of vertical scrolling.

Finally, we did not get the chance of testing SMT2 $\epsilon$ 's automatic profile generation capabilities (the behavioral clustering module) with our participants. As this feature is exclusive to our system, there is no reasonable comparison to other tools for obtaining the same information at present.

**5.3.2. Measuring User Subjectivity and System Workload.** Beyond these previous qualitative observations, we also measured more tangible factors of both tracking systems. When the two-month tracking campaign finished, participants filled in a couple of online questionnaires. One focused on measuring system usability and other focused on measuring participants workload. The results are shown next.

Regarding user subjectivity, we used the System Usability Scale (SUS) questionnaire [Brooke 1996]. SUS comprises a ten-item scale giving a global view of subjective assessments of usability. SUS scores range from 0 to 100 (the higher the score, the better). As observed, both systems can be considered to be similarly perceived in terms of SUS: 73 (SD = 7.7) for CLICKTALE and 72.5 (SD = 17.7) for SMT2 $\epsilon$ . Differences are not statistically significant, with a small effect size [paired  $t$ -test, two tails,  $t(9) = 0.10$ ,  $p = .918$ ,  $d = 0.03$ ,  $r = 0.01$ ] (Figure 11).

As for the system workload, we used the NASA Task Load Index (TLX) [Hart 2006]. TLX uses six dimensions to assess perceived workload: mental demand, physical demand, temporal demand, performance, effort, and frustration. Workload scores range from 0 to 100 (the lower the scores, the better). Overall, both systems can be considered

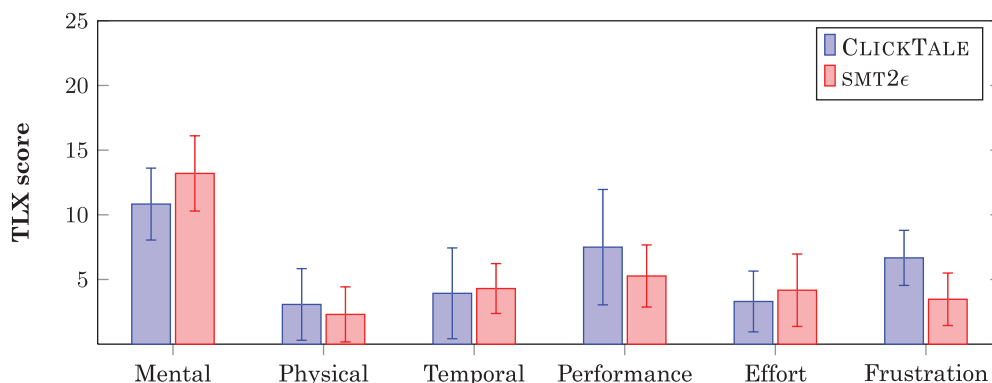


Fig. 12. Overall results of the workload study. Error bars denote 95% confidence intervals.

Table IV. Detailed Results of the Workload Study

	Mental	Physical	Temporal	Performance	Effort	Frustration
CLICKTALE	10.8 (4.4)	3.0 (4.4)	3.9 (5.6)	7.5 (7.2)	3.3 (3.7)	6.6 (3.4)
SMT2ε	13.2 (4.6)	2.3 (3.4)	4.3 (3.1)	5.2 (3.8)	4.1 (4.5)	3.4 (3.2)
$t(9)$	-1.15	0.83	-0.22	1.51	-1.45	1.87
$p$ -value	.278	.423	.826	.164	.180	.094
Cohen's $d$	-0.54	0.20	-0.08	0.40	-0.21	1.00
Pearson's $r$	-0.26	0.10	-0.04	0.19	-0.10	0.44

Note: The first two rows account for the mean (and SD) of the dimension scores, while the last four rows account for the results of the two-tailed paired  $t$ -tests.

to be similarly perceived in terms of TLX: 35.3 (SD = 17.2) for CLICKTALE and 32.7 (SD = 17.6) for SMT2ε. Differences are not statistically significant, with a small effect size [paired  $t$ -test, two tails,  $t(9) = 0.75$ ,  $p = .467$ ,  $d = 0.15$ ,  $r = 0.07$ ] (Figure 12).

A more detailed examination of the data revealed that each system scored slightly better than the other in half of the dimensions analyzed. Using Bonferroni correction at the  $p < 0.05/6 = .008$  level, differences were not found to be statistically significant for all of the six dimensions analyzed, with small to moderate effect sizes (Table IV).

In light of these results, our findings suggest that both systems perform approximately the same way. However, besides the lack of statistical significance, three TLX dimensions are worth commenting. First, SMT2ε seems to be more mentally demanding, with a moderate effect size ( $d = -0.54$ ,  $r = -0.26$ ). This finding was supported by the comments users made about SMT2ε requiring more cognitive effort than CLICKTALE to begin with. Second, CLICKTALE seems to be more frustrating than SMT2ε, with a high effect size ( $d = 1$ ,  $r = 0.44$ ). This finding was also supported by looking at user's comments, who specifically pointed out that SMT2ε provides particularly useful visualization possibilities. Finally, an important observation is that SMT2ε seems to perform better than CLICKTALE, with a moderate effect size ( $d = 0.4$ ,  $r = 0.19$ ). In this regard, participants often commented that they missed the possibility of aggregating multiple logs to visualize in a single replay with CLICKTALE.

To end this section, we should mention a series of limitations in the user study. First, we acknowledge that the sample size of the experiment is not large enough to draw definitive conclusions. While the results seem promising, in order to achieve statistically significant differences in TLX and SUS scores, we estimate that the sample size should be one order of magnitude higher, that is, about a hundred of users. Second, while the participants were actual professionals working on Web design and

development, they had no previous experience in using mouse tracking tools. Therefore, their use of these systems or the criteria on which they based their judgments may be very different from those of expert users of such systems. Third, the current evaluation is based on high-level subjective measurements, which are useful but perhaps do not provide the full picture. Other measures of effectiveness, such as task success rate and completion time could have helped to gain more insights into system usage. However, the instructions given to the participants were open-ended, that is, people were asked to evaluate both systems when there is no single correct answer. All in all, we feel that the feedback provided by the participants was very valuable and that these findings can be used to guide further improvement in both tracking systems.

## 6. PREDICTION EXPERIMENTS

SMT2 $\epsilon$  automatically extracts cursor features that can be used for user and page modeling (see Figure 8). Therefore, we felt that a study to show the value of these features was necessary. Eventually, this section illustrates that it is possible to do more research with our system.

We ran a series of experiments to predict time spent on page, cursor activity (amount of mouse movements), number of clicks, and vertical scroll reach. The decision to carry out these experiments over these cursor features was three-fold. First, research has shown that these features are mostly correlated with page relevance and user interest [Hijikata 2004; Guo and Agichtein 2012] and may have diverse applications in real-world contexts, for example, for website personalization, recommender systems, or Web search results reranking. Second, these features can be computed by mining the gathered logs, so they provide ground truth data to measure prediction accuracy, without having to manually label each log. Third, these experiments illustrate something that cannot be done with commercial systems, since commercial systems prevent researchers from accessing the data in raw form. We believe this latter point is of the utmost importance, since it should be possible for everyone to analyze cursor data with their preferred software, no matter how the data were captured.

Recapitulating from Section 4.4, SMT2 $\epsilon$  automatically computes 15 cursor features by mining the raw cursor data on the server side: browsing time (1D), cursor activity (1D), number of clicks (1D), cursor distance (1D), trail length (1D), cursor range (2D), scroll reach (2D), entry point (2D), exit point (2D), and cursor centroid (2D). We denote in parentheses the number of dimensions of each feature, that is, 1D means a single value, while 2D means that the feature has an horizontal and a vertical component.

### 6.1. Method

For each experiment, the feature that would be predicted was modeled as a linear combination of the remaining 14 features. Models were fitted using ordinary least squares for multiple regression. In order to decide the best features for the final fitting, we used the Information Criterion (IC) approach, which minimizes the information loss of the fit.

$$IC(k) = -2 \log(\mathcal{L}) + g, \quad (1)$$

where  $\mathcal{L}$  is the maximum likelihood for the model. When  $g = 2k$ , 1 is the Akaike's information criterion (AIC), while  $g = k \log(n)$  is the Bayesian information criterion (BIC),  $k$  being the number of parameters in the model and  $n$  the number of observations. AIC and BIC are both based on the idea that minimizing the relative entropy between the 'true' distribution and the tentative model yields the optimal model.

### 6.2. Procedure

We chose BIC for our experiments since, according to the literature, AIC frequently prefers a more complex model over a simpler model [Burnham and Anderson 2004],

Table V. Overview of Prediction Results Using the Models for Classification

Predicted Feature	% Accuracy	Adjusted $R^2$
browsing time	84.95 (1.9)	0.49 (0.02)
no. clicks	91.79 (1.3)	0.21 (0.01)
cursor activity	74.06 (2.1)	0.24 (0.01)
scrollY	94.21 (1.3)	0.24 (0.02)

Note: SDs are denoted in parentheses.

Table VI. Regressor Coefficients for Predicting Time Spent on Page

Feature	Coefficient	Std. Error
(intercept)	10.7	3.3
cursor activity	-1.1	$1.4 \cdot 10^{-1}$
no. clicks	1.2	$2.1 \cdot 10^{-1}$
trail distance	$-1.3 \cdot 10^{-4}$	$2.1 \cdot 10^{-5}$
trail length	$4.2 \cdot 10^{-8}$	$1.6 \cdot 10^{-8}$
entryX	$4.1 \cdot 10^{-2}$	$8.8 \cdot 10^{-3}$
exitY	$3.7 \cdot 10^{-3}$	$7.4 \cdot 10^{-4}$
rangeX	$6.3 \cdot 10^{-2}$	$8.2 \cdot 10^{-3}$
rangeY	$8.7 \cdot 10^{-3}$	$7.2 \cdot 10^{-4}$
centroidX	$-5.5 \cdot 10^{-2}$	$1.2 \cdot 10^{-2}$

Note: According to the model that provided the best classification accuracy (89.61%). All features were found to be statistically significant ( $p < .0001$ ) at the .05 level.

and it may fail to converge in probability to the true model, whereas BIC tends to choose a simpler model and does converge as  $n$  increases [Raftery 1995].

After the fitting, we used a variant of K-fold cross validation to evaluate the accuracy of the models. Up to 50 times, we allocated two data partitions: one for training and other for testing. Observations were randomly assigned to a partition, resulting in 70% of the samples (1,216 logs) for training and 30% (522 logs) for testing on each iteration. Therefore, a model derived from the training partition would be used as a binary classifier on the testing partition. A classification error was considered when the difference between the predicted feature and its true value was higher than (or equal to) the residual standard error provided by the model. Experiments were carried out with the R stats package.

### 6.3. Results and Discussion

Table V provides an overview of the experimental results. The best fitting models are given in Tables VI to IX.

Regarding time spent on page, prediction accuracy is 84.95% (SD = 1.9). All regressors in Table VI add statistical significance to the model ( $p < .0001$ ); however, it can be observed that trail distance and trail length could be considered weak predictors with extremely small coefficients. The adjusted  $R^2$  is 0.49, which suggests that the model is fairly adequate for predicting browsing time for unseen data. (Adjusted  $R^2$  decreases as predictors are added to the model, and this model uses 9 out of 14 predictors.)

In addition, the model reports a negative coefficient for cursor activity, which is consistent with what theory predicts, that is, a small amount of mouse movement means that the mouse cursor was motionless most of the time, indicating that browsing time is likely to be high. This observation is also consistent with the model shown in

Table VII. Regressor Coefficients for Predicting Number of Clicks

Feature	Coefficient	Std. Error
(intercept)	$-5.5 \cdot 10^{-1}$	$5.2 \cdot 10^{-1}$
browsing time	$5.0 \cdot 10^{-2}$	$3.8 \cdot 10^{-3}$
rangeX	$-4.0 \cdot 10^{-3}$	$9.8 \cdot 10^{-5}$
scrollX	$9.0 \cdot 10^{-2}$	$1.4 \cdot 10^{-2}$

*Note:* According to the model that provided the best classification accuracy (95.19%). All features were found to be statistically significant ( $p < .0001$ ) at the .05 level.

Table VIII. Regressor Coefficients for Predicting Cursor Activity

Feature	Coefficient	Std. Error
(intercept)	8.28	$8.3 \cdot 10^{-1}$
browsing time	$-7.8 \cdot 10^{-2}$	$6.8 \cdot 10^{-3}$
entryX	$1.2 \cdot 10^{-2}$	$1.5 \cdot 10^{-3}$
exitY	$-4.1 \cdot 10^{-4}$	$1.3 \cdot 10^{-4}$
rangeX	$1.5 \cdot 10^{-2}$	$1.5 \cdot 10^{-3}$
rangeY	$4.0 \cdot 10^{-4}$	$1.2 \cdot 10^{-4}$
centroidX	$-6.3 \cdot 10^{-1}$	$2.2 \cdot 10^{-3}$

*Note:* According to the model that provided the best classification accuracy (78.07%). All features were found to be statistically significant ( $p < .0001$ ) at the .05 level.

Table VIII. Similarly, the number of clicks is typically expected to be proportional to the time spent on page, which is also reported by the model shown in Table VII, with a positive coefficient in both cases. These observations suggest that prediction of browsing time could be derived from cursor features with good prospects of success.

For the other three models, prediction accuracy was found to be equally satisfactory, especially regarding clicks ( $M = 91.79\%$ ,  $SD = 1.3$ ) and vertical scroll reach ( $M = 94.21\%$ ,  $SD = 1.3$ ). Similarly, with previous observations, model coefficients were found to be consistent with the expected results. For instance, it was expected that both entry coordinate and vertical cursor range would have an influence on predicting scroll reach, which is reported by the model shown in Table IX. Nevertheless, adjusted  $R^2$  scores for the remaining models were around 0.2, which suggests that these models may be affected by other factors beyond the interaction features provided by SMT2 $\epsilon$ . As such, more research on building better models seem to be a promising avenue for future work.

Finally, a wrap-up discussion on the value of these experiments is given in the next section. We should mention that these experiments provide researchers with a quantitative assessment of the correlation between regressors, which can be used in prospective studies to decide which features are worth considering in a user model that leverages mouse cursor data. For instance, Guo and Agichtein [2012] performed a series of experiments in a similar vein, where document relevance was predicted from low-level cursor-based behavioral signals. Concretely, trail distance and range were found to be the most predictive signals of document relevance.

## 7. GENERAL DISCUSSION

Clearly, there is a compromise solution between tracking frequency and visualization roughness, that is, the higher the frame rate, the better the replay quality, but also

Table IX. Regressor Coefficients for Predicting Vertical Scroll Reach

Feature	Coefficient	Std. Error
(intercept)	11.49	13.1
entryY	$6.0 \cdot 10^{-2}$	$1.3 \cdot 10^{-2}$
rangeY	$3.8 \cdot 10^{-2}$	$2.0 \cdot 10^{-3}$
scrollX	$6.8 \cdot 10^{-1}$	$2.4 \cdot 10^{-1}$
centroidX	$-6.5 \cdot 10^{-2}$	$2.2 \cdot 10^{-2}$
centroidY	$-6.1 \cdot 10^{-2}$	$4.2 \cdot 10^{-3}$

*Note:* According to the model that provided the best classification accuracy (96.53%). All features were found to be statistically significant ( $p < .0001$ ) at the .05 level.

the higher the amount of data to be captured, and therefore the higher the overhead incurred by logging. Ensuring a low overhead is especially important in mobile contexts, where battery life is limited and where networks can be slow and very expensive. Therefore, using tracking frame rates around 20 Hz seem to be a well-balanced recommendation. Even better, another possibility would be to record user interaction only when a browser event is fired. This would require introducing some modifications to SMT2 $\epsilon$  in order to synchronize different cursor streams, but it would nevertheless contribute to reducing (even more) the size of the logged data. As regards user-visible performance, no participant reported any degradation incurred by logging during the evaluation. This was unsurprising, since the data collecting procedures of SMT2 $\epsilon$  are not computationally demanding and the tracking frame rate was set to 24 Hz, resulting in Ajax requests of around 2 kB on average each 30 seconds. Moreover, as indicated in Figure 4, SMT2 $\epsilon$  makes use of asynchronous Ajax requests, which are nonblocking and therefore allow for regular interaction with the page.

On the other hand, we have shown that SMT2 $\epsilon$  has a very small footprint in terms of number of HTTP requests performed. This also helps reduce overheads incurred by logging, as HTTP requests introduce an important load increase on the server. Not only is there a header block for both the request and the response, there is also a three-way handshake due to the TCP/IP layer, which underlines the transmission protocol. So, in the absence of Web server configuration tweaks, it is commonly agreed that transmitting more data in less HTTP requests performs better than the opposite case. This is another opportunity for industry systems to improve logging performance. Additionally, if the server supports persistent connections via the Keep-Alive header, then the tracking interval of SMT2 $\epsilon$  can be set to the default connection timeout of the server. This is expected to reduce latency and network traffic, as well as saving CPU and memory usage on the server side.

As for cursor data compression, we must mention that SMT2 $\epsilon$  features LZW encoding for practical reasons. One might think that Huffman coding would be particularly appropriate instead. However, using Huffman codes to compress cursor data is suboptimal. The space saved while compressing the data is then penalized with the size of the Huffman tree, which must be transmitted to the server in order to reconstruct (decode) the compressed data. If the tree were always known and fixed, then Huffman coding might outperform LZW. Nevertheless, each cursor trail is unique by definition, having thus a unique decoding tree, and therefore the data must be transmitted together with the tree. So, the size of a Huffman-compressed cursor trajectory plus its corresponding decoding tree is typically higher than the size of the data encoded with LZW. Perhaps some hybrid strategy would be worth trying in a future, for example, using Huffman to compress cursor data and LZW to compress Huffman trees.

Concerning usability evaluation and acknowledging the limitations of these experiments, we are very satisfied with the results. With around 20 employees<sup>9</sup>, CLICKTALE is a leader in the cursor tracking industry, while SMT2 $\epsilon$  is a single person's work. Overall, we did not find statistically significant differences in terms of SUS or TLX scores. From the results, it was observed that SMT2 $\epsilon$  was perceived to be more mentally demanding to operate but CLICKTALE caused more frustration to the users. These findings could be used to guide further improvement of both systems, especially for SMT2 $\epsilon$  in terms of identifying ways to decrease the user's workload.

On another level, we have shown a series of cursor features that can be predicted with high accuracy on average: 78% in the worst case, when predicting cursor activity; and 96.53% in the best case, when predicting vertical scroll reach. This could serve thus as a design guide for deciding which features are worth considering in a user model. Nevertheless, we capitalize on the fact that the purpose of the prediction experiments is merely to illustrate that SMT2 $\epsilon$  provides access to raw cursor data and that they are both accessible and easily manipulable. Therefore, these can be used to test theories on user behavior using real observations. Other features could also provide more insights about page-level interactions. For instance, being able to predict the centroid coordinate would allow the webmaster to consider placing interesting content near this coordinate, or a link to some promoted page. In addition, whether a model is truly a good fit or not depends on the context of the study. Using a more advanced technique for multiple regression may be desirable to increase precision and fit quality, although it is definitely outside the scope of this article.

To conclude, we remark that most of these studies would have never been possible without a system like SMT2 $\epsilon$ , since current commercial tools do not allow users to handle tracking data in such fine-grained detail.

## 8. SYSTEM LIMITATIONS

Web-based activity tracking systems have inherent limitations, and of course SMT2 $\epsilon$  is no exception. Although measuring page-level interactions is cheaper and enables remote data collecting on a large scale, the main drawback we have found is that assessing the allocation of visual attention based on interaction data alone is a non-trivial task. For instance, while it is commonly agreed that "a mouse cursor can tell us more" [Chen et al. 2001], other researchers have demonstrated that browsing time and user behavior have notable repercussions on gaze and mouse cursor alignment [Huang et al. 2012]. Also, it has been shown that users do not necessarily pay attention to what they are looking at, and they do not necessarily look at what they are paying attention to [Toet 2006]. Therefore, the usability practitioner should be aware of these facts before considering using a Web tracking system, depending on the task that would be assessed or the context of the study.

In addition, SMT2 $\epsilon$  allows the webmaster to control whether users should be informed that their cursor activity is being (or going to be) tracked. Although a privacy-friendly remainder like this may benefit the user, it might introduce evaluation artifacts, as users could change their usual browsing behavior. In fact, the Hawthorne effect or "observation bias" is a well-documented phenomenon that affects many research experiments [Adair 1984]. Therefore, the trade-off between tracking transparency and user privacy should be taken into account as well in a remote usability evaluation.

On the other hand, our tool was designed to handle a limited number of simultaneous user sessions in the same hypervideo. One may note that if the system were used to show data from, say, 10,000 concurrent users, then we believe the video visualization would not be very meaningful. Suffice to say it could be done, but at the cost of increasing

<sup>9</sup><http://www.aboutanalytics.com/clicktale>.



the cognitive load (since visually inspecting too many users at the same time can be stressful), and only limited by the processing power of the computer. In this situation, aggregated data would work much better if rendered as a single image—discarding thus the temporal information but retaining interactivity for the viewer. This way, it is still possible to visually infer time-based properties, such as mouse velocities (e.g., by looking at the ‘directions & distances’ layer, Figure 7).

Furthermore, besides the fact that our tool normalizes the mouse coordinates to avoid possible visual biases while replaying the hypervideos (even when the browser window is resized), we have noticed that sometimes visualization is not perfectly accurate, partly due to JavaScript rounding errors, partly due to discrepancies between how browsers render CSS. As such, in these cases, a few pixels could make a difference: image, for instance, a Firefox user clicking on the very top-right corner of an image link; if the video were rendered on the same screen dimensions but on a different browser, then the viewer could see that user clicking on an empty space (depending of course on the CSS rules that were established). These browser discrepancies could be greatly minimized by using a reset stylesheet on the webpage. On the other hand, higher discrepancies are expected when the user accesses from a mobile device and the viewer uses a desktop computer. We are currently investigating different methods that would tackle this problem, which is common to all Web-based tracking systems, and for which there is no direct solution. For instance, the system could use the mobile user agent to fetch the page the user had visited, but perhaps the page has been updated, or it no longer even exists. The same argument may apply to the stylesheets of that page. Therefore, a technically more advanced approach should be taken into consideration, such as caching all the assets for each single user session, at the cost of increasing the storage space on the tracking database.

Related to the previous issue, we should mention that, depending on page complexity, it could be the case that the screen shown in the playback is not exactly the one the users saw. For instance, zooming, JavaScript prompts, pulldown menus, or Ajax calls that validate form data or modify the DOM at runtime would not be shown during the replay. This can be solved by triggering events that relate to the cursor trajectory (e.g., hovering, clicking, etc.) or other registered browser events (e.g., keypresses). However, this approach could have undesired side effects, especially when replaying multiple logs simultaneously. For instance, event triggering may cause navigating away from the current page, inserting redundant information into a website database, or resubmitting a shopping cart form. As a result, we decided not to implement event triggering for purely practical reasons.

Finally, it is worth pointing out that some users may utilize other I/O devices to assist Web browsing, such as screen readers or speech recognizers. We acknowledge that in this case, no cursor tracking system would be appropriate for studying the behavior of those users, and in fact, we encourage the expert practitioner to consider other input signals if available when running usability tests. For instance, if an eye-tracker were used, it would be possible to easily synchronize both cursor and gaze streams, as indicated, for example, by Huang et al. [2011, 2012].

## 9. CONCLUSION

To better understand Web-based interaction, current tracking systems should rely on the browsing capabilities of the users instead of traditional server access logs. We believe delivering interaction data as a hypervideo for assessing the usability of websites is a promising, helpful idea.

This article has described the design and implementation of *SMT2 $\epsilon$* , a tool for automatically gathering, mining, and visualizing browsing data in an interactive hypermedia presentation. *SMT2 $\epsilon$*  collects fine-grained information about user behavior and allows

viewers to control what they watch, when, and how, by selecting diverse types of infographics. Our proposal is part of a larger move to bring more interactivity into browsing data exploration and analysis.

We have reported the main differences between SMT2 $\epsilon$  and both past and current Web tracking systems, including its predecessor (SMT2). We have also shown the value of enhancing video visualizations with interactive techniques to present the viewer with complex information quickly and clearly. A series of experimental evaluations have illustrated that ours is an efficient implementation and that it sometimes performs better than a leading commercial system.

Tracking page-level browsing activity with SMT2 $\epsilon$  requires no real effort from the user, other than standard usage. It also requires no training and provides context for actions. Armed with this awareness, one may conduct qualitative or quantitative studies to complement existing methodologies on Web browsing behavior. We believe that SMT2 $\epsilon$  is ready to extend its scope to a broader, interdisciplinary audience.

## 10. FUTURE WORK

Since 2009, we and others have used our system for a number of research studies on different websites. At present, it has been downloaded more than 100K times, and some of its components are being used in other projects and even in commercial products. This encourages us to continue to make improvements to the system.

Currently we are investigating new techniques for cursor data compression so that the logging footprint can be further minimized. As previously mentioned in Section 8, one of our priorities for future work is implementing a more advanced caching method for minimizing rendering discrepancies, for instance, when the same URL is accessed from mobile devices and desktop computers. We also plan to work on scalability and performance limits, especially concerning high-recording speeds. In this regard, we plan to test the HTML5 WebSocket API for transmitting data. Using this technology would also enable real-time activity sharing, were users and viewers interact concurrently (e.g., an online customer support service).

Another line of research could lean towards enriching the system with other types of behavior analysis, such as working with eye-tracking data, since user interaction is inherently multimodal. Therefore, considering more independent data sources could only make evaluation studies stronger. Finally, the reader is encouraged to try SMT2 $\epsilon$ , which is open-source software that can be inspected at no cost and downloaded from <http://smt2.googlecode.com>.

## ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

## APPENDIX

Detailed monitoring of user interactions at the page level could be very useful to help shape a more usable website or make it more appropriate to the skills and abilities of their users. However, as in other Web tracking applications, this work raises privacy concerns. We are interested in understanding Web browsing behavior, but we also want the user to be respected, so we designed the SMT2 $\epsilon$  system with this notion in mind.

First, we believe that logging keystrokes could be employed for unfair purposes, depending on the uses that one could derive from this tool. For this reason, we rejected logging raw keystroke data and decided to track only keyboard events instead, without registering the associated character codes. Second, we believe that users should not be monitored without their consent. This is a webmaster's responsibility, but not doing so could be considered unethical in some countries. Therefore we recommend always asking the user before tracking takes place. Furthermore, once a user has agreed to

tracking, we advocate asking their consent again after a prudent amount of time (e.g., a few hours, until the end of the browsing session, or when a tracking campaign ends). Third, we believe that logged data should be stored in a server the webmaster owns and not in one they cannot control. At least, it should be possible to let users access and download their (raw) data. We encourage commercial tracking systems to do so, since the possibility exists and current Web technologies can support this. Finally, unlike most analytics packages that track other sites users have visited or the searches they have made, we do not collect other information than basic browser events derived from normal interaction with the site. This way, we try to avoid illegitimate abuse of our system (e.g., not notifying at all that users are being tracked or hijacking submitted form data). Above all, the ethical use of computers should be above any functionality or feature.

### ACKNOWLEDGMENTS

The authors wish to thank Ryen White and the anonymous reviewers for their helpful comments and suggestions for improving this manuscript. We also thank the staff members at `pepegimeno.com` and `nectarestudio.com` who volunteered to take part in our evaluation studies.

### REFERENCES

- ADAIR, J. G. 1984. The Hawthorne effect: A reconsideration of the methodological artifact. *J. Appl. Psychol.* 69, 2, 334–345.
- AGICHTEIN, E., BRILL, E., AND DUMAIS, S. 2006. Improving Web search ranking by incorporating user behavior information. In *Proceedings of the 29th conference on Research and Development in Information Retrieval (SIGIR)*. 19–26.
- ARROYO, E., SELKER, T., AND WEI, W. 2006. Usability tool for analysis of Web designs using mouse tracks. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*. 484–489.
- ATTERER, R., WNUK, M., AND SCHMIDT, A. 2006. Knowing the user's every move – User activity tracking for website usability evaluation and implicit interaction. In *Proceedings of the 15th International Conference on World Wide Web (WWW)*. 203–212.
- BROOKE, J. 1996. SUS: A “quick and dirty” usability scale. In *Usability Evaluation in Industry*, P. W. Jordan, B. Thomas, B. A. Weerdmeester, and A. L. McClelland (Eds.). Taylor and Francis, London.
- BURNHAM, K. P. AND ANDERSON, D. R. 2004. Multimodel inference: Understanding AIC and BIC in model selection. *Socio. Methods Res.* 33, 2, 261–304.
- BUSCHER, G., WHITE, R. W., DUMAIS, S., AND HUANG, J. 2012. Large-scale analysis of individual and task differences in search result page examination strategies. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM)*. 373–382.
- CHEN, M. C., ANDERSON, J. R., AND SOHN, M. H. 2001. What can a mouse cursor tell us more? Correlation of eye/mouse movements on Web browsing. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*. 281–282.
- DANIEL, G. AND CHEN, M. 2003. Video visualization. In *Proceedings of the IEEE Conference on Visualization (VIS)*. 409–416.
- FRANCISCO-REVILLA, L. 1998. *A Picture of Hypervideo Today*. Center for the Study of Digital Libraries Technical Report. Texas A&M University, College Station, TX.
- GUO, Q. AND AGICHTEIN, E. 2010a. Ready to buy or just browsing? Detecting Web searcher goals from interaction data. In *Proceedings of the 33rd International ACM Conference on Research and Development in Information Retrieval (SIGIR)*. 130–137.
- GUO, Q. AND AGICHTEIN, E. 2010b. Towards predicting Web searcher gaze position from mouse movements. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*. 3601–3606.
- GUO, Q. AND AGICHTEIN, E. 2012. Beyond dwell time: Estimating document relevance from cursor movements and other post-click searcher behavior. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*. 569–578.
- GUYON, I., SCHOMAKER, L., PLAMONDON, R., LIBERMAN, M., AND JANET, S. 1994. UNIPEN project of on-line data exchange and recognizer benchmarks. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*. 29–33.
- HART, S. G. 2006. NASA-Task Load Index (NASA-TLX); 20 years later. In *Proceedings of the 50th Annual Meeting on Human Factors and Ergonomics Society (HFES)*. 904–908.

- HAUGER, D., PARAMYTHIS, A., AND WEIBELZAHN, S. 2011. Using browser interaction data to determine page reading behavior. In *Proceedings of the 19th International Conference on User Modeling, Adaptation, and Personalization (UMAP)*. 147–158.
- HAUGER, D. AND VAN VELSEN, L. 2009. Analyzing client-side interactions to determine reading behavior. In *Proceedings of the 17th Workshop on Adaptivity and User Modeling in Interactive Systems (ABIS)*. 11–16.
- HIJIKATA, Y. 2004. Implicit user profiling for on demand relevance feedback. In *Proceedings of the 9th International Conference on Intelligent User Interfaces (IUI)*. 198–205.
- HIRATA, K., HARA, Y., SHIBATA, N., AND HIRABAYASHI, F. 1993. Media-based navigation for hypermedia systems. In *Proceedings of the 5th ACM Conference on Hypertext (HT)*. 159–173.
- HUANG, J., WHITE, R. W., AND BUSCHER, G. 2012. User see, user point: Gaze and cursor alignment in Web search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. 1341–1350.
- HUANG, J., WHITE, R. W., AND DUMAIS, S. 2011. No clicks, no problem: Using cursor movements to understand and improve search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. 1225–1234.
- JACOBSEN, N. E., HERTZUM, M., AND JOHN, B. E. 1998. The evaluator effect in usability tests. In *Proceedings of the CHI Conference Summary on Human Factors in Computing Systems*. 255–256.
- LEIVA, L. A. 2011. Mining the browsing context: Discovering interaction profiles via behavioral clustering. In *Adjunct Proceedings of the 19th Conference on User Modeling, Adaptation, and Personalization (UMAP)*. 31–33.
- LEIVA, L. A. AND VIDAL, E. 2010. Assessing user's interactions for clustering Web documents: A pragmatic approach. In *Proceedings of the 21st ACM Conference on Hypertext and Hypermedia (HT)*. 277–278.
- LEIVA, L. A. AND VIVÓ, R. 2012. Interactive hypervideo visualization for browsing behavior analysis. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*. 381–384.
- LIESTOL, G. 1994. Aesthetic and rhetorical aspects of linking video in hypermedia. In *Proceedings of the ACM European Conference on Hypermedia Technology (ECHT)*. 217–223.
- MUELLER, F. AND LOCKERD, A. 2001. Cheese: Tracking mouse movement activity on websites, a tool for user modeling. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*. 279–280.
- MÜLLER-TOMFELDE, C. 2007. Dwell-based pointing in applications of human computer interaction. In *Proceedings of the 11th IFIP TC 13th International Conference on Human-Computer Interaction—INTERACT*. Lecture Notes in Computer Science, Vol. 4662, Springer, Berlin Heidelberg, 560–573.
- RAFTERY, A. E. 1995. Bayesian model selection in social research. *Socio. Methodol.* 25, 1, 111–163.
- REEDER, R. W., PIROLI, P., AND CARD, S. K. 2001. WebEyeMapper and WebLogger: Tools for analyzing eye tracking data collected in Web-use studies. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*. 19–20.
- RODDEN, K., FU, X., AULA, A., AND SPIRO, I. 2008. Eye-mouse coordination patterns on Web search results pages. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*. 2997–3002.
- SHIFFER, M. J. 1995. Interactive multimedia planning support: Moving from stand-alone systems to the World Wide Web. *Environ. Plan. B: Planning and Design* 22, 6, 649–664.
- SMITH, J. AND STOTTS, D. 2002. *An Extensible Object Tracking Architecture for Hyperlinking in Real-time and Stored Video Streams*. Technical Report 02-017. University of North Carolina and Chapel Hill.
- ŠPAKOV, O. AND MINIOTAS, D. 2007. Visualization of eye gaze data using heat maps. *Electron. Electr. Eng.* 74, 2, 55–58.
- TARASEWICH, P., POMPLUN, M., FILLION, S., AND BROBERG, D. 2005. The enhanced restricted focus viewer. *Int. J. Human Comput. Interact.* 19, 1, 35–54.
- TOET, A. 2006. Gaze directed displays as an enabling technology for attention aware systems. *Comput. in Human Behav.* 22, 4, 615–647.
- TUKEY, J. W. 1977. *Exploratory Data Analysis*. Addison-Wesley Publishing Co, Boston, MA.

Received March 2012; revised October 2012, March, June 2013; accepted June 2013