Technical Overview of Eye-Tracking Study

Study published as:
Whalen, T., Inkpen, K. Gathering Evidence: Use of Visual Security Cues in Web Browsing. Graphics Interface 2005.

Here are a few short notes about our experimental setup, as this may be useful for those running a simulated web study. First, a quick overview of our study: we wanted to gather data about people's attention to browser elements related to security, while they engaged in both security-related and non-security related tasks. The goal was to see whether these elements were noticeable; we gathered data during the web tasks by using an eyetracker. The eyetracker hardware recorded eye movement data that was processed with accompanying software.

In an ideal world, we would place an eyetracker in a participant's usual browsing location, capture all the eyetracking data and screenshots, and perform the analysis. This was, of course, impossible. First of all, the eyetracker is a specialized piece of equipment that requires precise calibration: it is not suited to use outside of a lab setting. Secondly, and more importantly, we cannot collect private user data: recording their actual banking transactions, passwords, email, etc., is obviously a violation of ethical standards. It may be technically possible to exclude these items from the collected data, but we were unable to find a way to do so when we ran this study. Our eyetracking software collected full screen data, including mouse movements and all entered data, so that we could replay and analyse the entire set of tasks. (See screenshot in Figure 1, below). Therefore, we had to use specialized accounts and simulated websites, rather than actual user data and accounts.

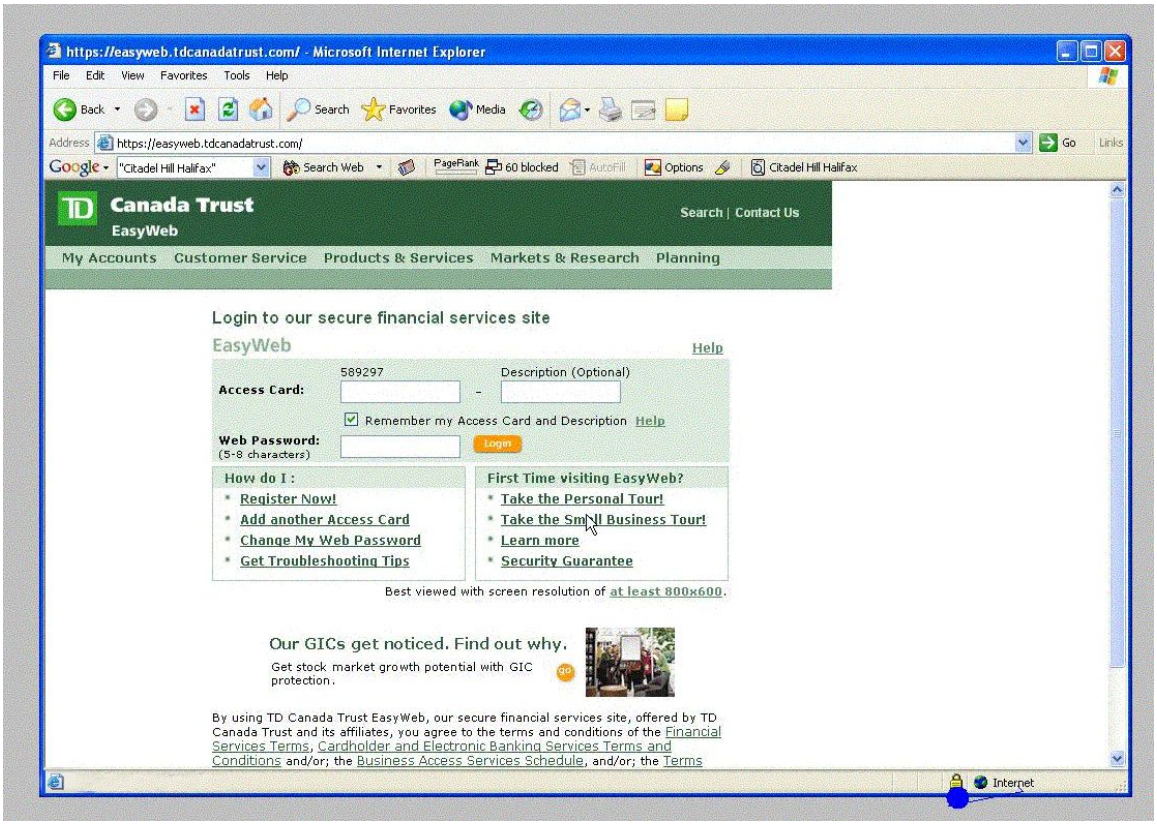

Figure 1: example of data collected by eyetracker (login data has not yet been entered)

We created special email accounts on hotmail and Gmail; since anyone can create an account here, it was very easy to do so for our experimental purposes. However, this could not be easily done for bank accounts: conceivably, one might enter into a partnership with a bank to create a dummy bank account, but this might not be possible (especially with several banks). Similarly, we did not want to make a "real" purchase on a merchant site with "fake" credit card data, as this would likely constitute fraud. We also could not have participants make a real purchase of an item, due to budget constraints. Therefore, for credit card and banking transactions, we used a proxy server to serve up fake, local web pages for the secure part of the site. Figure Two gives a visual overview of our setup.

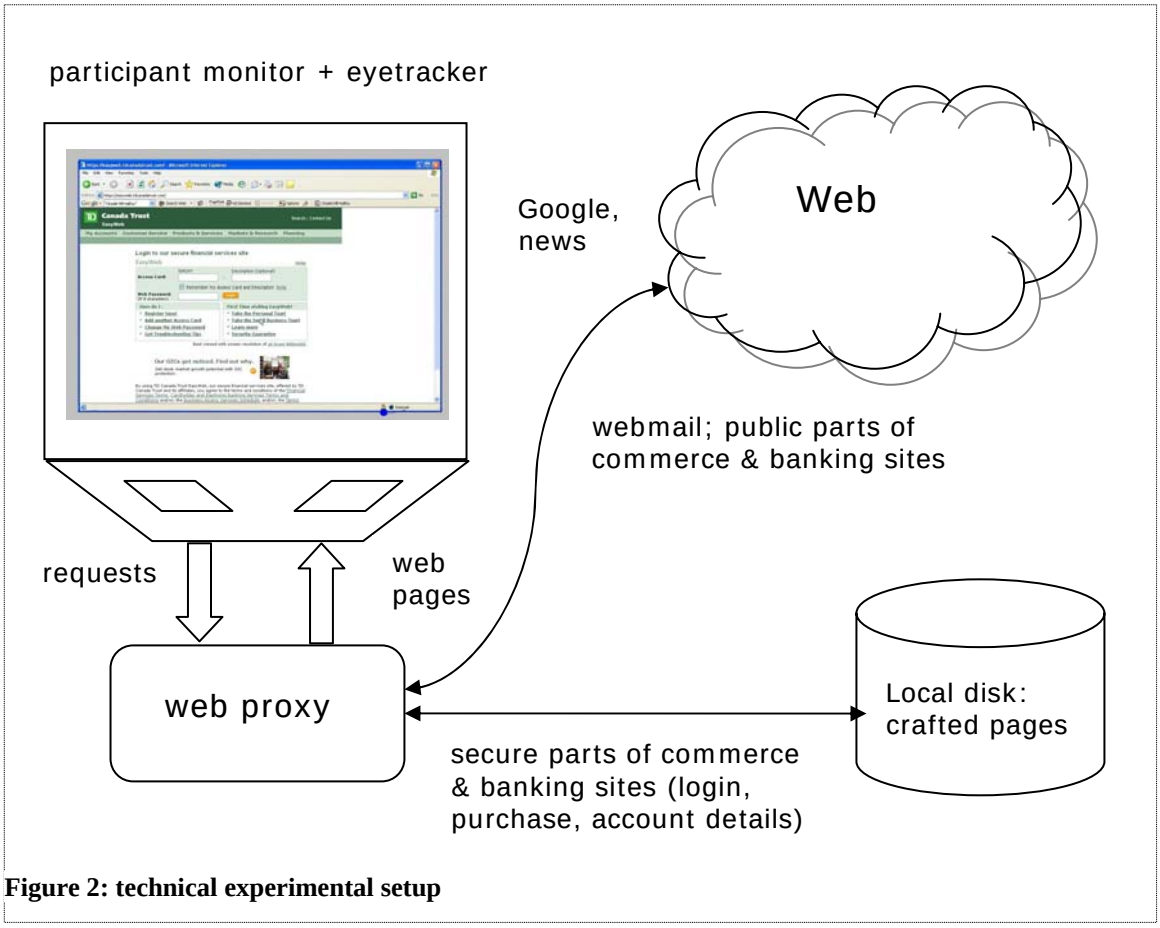

Figure 2: technical experimental setup

We had three basic types of secure transaction, and two non-secure tasks. For the non-secure tasks (use Google, go to a news site), the proxy directed the requests out to the regular web. The three secure transactions consisted of checking webmail, purchasing with a credit card, and checking a bank balance online. We had created real webmail accounts for this experiment, so these requests were directed to the actual web. However, we had to simulate portions of two commerce sites (well-known Canadian bookstore, and obscure cable supplier), as well as portions of five major banks. We used Httracker to mirror portions of the real sites, which could be returned as necessary. This was easy to do for the public portions of the website, but more complex for the private account portions.

For the commerce sites, we walked through a purchase and stopped short of the final step (actually purchasing the item). We created a "thank you" page from scratch, one that replicated the basic "look and feel" of the site. We had to carefully walk the participant through the purchase process: we placed a pre-selected item into the cart for them to buy. They could look at any part of the site, other than the secure purchase section, and would get the real web page. When they started the checkout process, they would see our mirrored and crafted pages instead. (This solution could be made more powerful; we took a very simple approach).

The bank pages were more complicated: if we wanted the participants to think they were really on their own bank's website, we had to make the account pages look exactly right. Again, we allowed them to browse most of the real site, but mirrored and crafted the login page and the account information pages. The trick was to find out what the real account pages looked like for all five major banks. We got help from people in our lab and department: people logged into their own accounts, downloaded the HTML, deleted and personal data (account balances, userIDs, etc.) and gave them to us for replication. Note that this is not a scalable solution; this really only worked because there were so few banks to choose from. It worked well enough for our purposes, however; the replication was very convincing. (The only good way to tell the difference was to look at the certificate.)

We generated self-signed certificates for the SSL connections. In the usual case, you would expect to see a warning message when users connected to our false local sites.
In our case, we suppressed the warning messages, because this was a confounding factor that was too complex to investigate in this specific study (i.e., how people respond to security pop-ups).

Notes on the Eyetracker

We used a Tobii x50 eyetracker to record gaze and eye movement data. This tracker is **not** head-mounted: it sits below the testing monitor. This allows the participant to have freedom of head movement and is less obtrusive than a head-mounted device; these features were useful for our tests, as we wished to simulate a natural browsing experience as much as possible. See Figure 3 for a picture of the eyetracking hardware device.

Figure 3: Tobii x50 eyetracker (photo from Tobii website)

The x50 contain a camera and near-infrared LEDs: the LEDs shine onto the participant's eyes and the reflection patterns are recorded by the camera. Calibration is required before each session, but this is generally a short and simple process taking less than one minute. A good calibration, along with the proper experimental setup, allows for approximately 0.5 degrees of accuracy to the target. (Note that one degree of accuracy means 1 cm difference, on average, between the actual and intended gaze point, when the person's eyes are 50 cm from the viewed object on the screen.) The x50 also tracks both eyes simultaneously (binocular tracking), a frame rate of 50 Hz, and a delay of approximately 35 ms. (Each gaze data point is timestamped, with an accuracy of +/- 5 ms.)

The Tobii x50 hardware is connected over Firewire and USB to a server, to which it sends all the eye tracking data it collects, which can then be analysed. Tobii have developed Clearview software for data analysis, which we used in our study. There are many features in this package, including Hotspot visualization and the ability to define Areas of Interest for tailored analysis. What we used was gaze replay, recorded over an entire session: all activity on the screen, for the entire session, was recorded, along with all gaze data. This created a video stream for replay: all actions that the user performed (including cursor movements) could be replayed, along with a superimposition of eye movements happening at that moment. Gaze data was shown as a blue spot: the spot got larger if the person looked at one spot, or became a line when the eye tracked across the screen. (See Figure 1, above, for a sample still.) We were then able to replay each participant's session, observing where they were looking on the screen as they performed each task. (This enabled us, for example, to see when people were reading elements on the screen, by watching the blue line streak from left to right across a line of text on a web site.) There are many other features available with this eye-tracker, and an API is provided to give the user even more flexibility. We used a simple approach, which sufficed to give us all the eyetracking information that we required for our particular study, but others may find that they need to tailor the software for their own needs.