Chapter 4

Human Multitasking

Multitasking takes place when someone tries to handle more than one task at the same time, switch from one task to another, or perform different tasks in (rapid) succession. Multitasking allows thus to coordinate multiple tasks cognitively, with the downside of redirecting the focus of attention away from the primary task and, like external interruptions, leading to disruptive shifts in thinking.

In this chapter we discuss the need to support multitasking while interacting with computers, with a clear focus on web browsing. We present MouseHints, a tool that aims to minimize the negative effects of interruptions on memory. By leveraging implicit interactions and using a combination of very basic infographics, the tool draws the user attention to the location of previously interacted areas on the screen. This way, we provide a method for adaptive memory cues that can facilitate task resumption.

Chapter Outline

4.1 Introduction

We now use the Web to multi-task the activities we do every day, to the extent that it is not unusual to see users with a dozen applications and browser instances open at a time; e.g., sharing pictures, listening to music, or shopping, just to name a few. Computers can display more tasks and more information than we can handle, and attention remains a finite resource [Fong, 2008].

Understanding how people browse the Web has been historically a subject of research, see, e.g., [Adar et al., 2009; Byrne et al., 1999]. Spink et al. [2004] reported that a single browsing session may consist of seeking information on single or multiple topics, and switch between tasks. Viermetz et al. [2006] noticed that the effect of viewing a website and branching the focus onto different windows was an increasingly popular web viewing methodology. Moreover, the tabbed browsing feature has boosted the acceptance of such a web viewing behavior. In fact, according to Dubroy and Balakrishnan [2010], tab switching is the second-most frequent action that people perform in their browser, after link clicking. This is interesting, because up to now it is been assumed that the primary thing that people do in their browser is clicking on links. And this may still be true (for some people), but tab switching is a close second. This means that the browser is used for navigation, but also as a task-management tool. People thus may cognitively coordinate multiple tasks through multi-tabbing, having many pages open at the same time and switching between them in any order.

Web browsing activities can be defined as high-level tasks, that is, users pursue an abstract or general concept (e.g., buy a book, learn to play a musical instrument, check the weather, etc.) and, to accomplish such a goal, tasks usually involve multiple steps or sub-rutines. Unfortunately, while we often maintain high level definitions of tasks in our minds, computer systems seldom support them [Humm, 2007]. Most UIs for switching between tasks require visual searches of candidates, namely *placeholders*—e.g., headlines, text paragraphs, or images—to retain spatial information about the UI and thus cognitively ease navigation as well as task resumption (see Section 4.2).

4.1.1 Preliminaries

Multitasking takes place when someone tries to perform two tasks simultaneously, switch from one task to another, or perform two or more tasks in (rapid) succession [APA, 2006]. König et al. [2005] refer to multitasking as the ability to accomplish multiple task goals in the same time span by engaging in frequent switches between individual tasks.

One may note that multitasking can involve, by definition, attentional branching between multiple tasks, both in the physical and the digital world; e.g., reading a book may require an online dictionary to search certain words and,

possibly, consulting some of the (interesting) book references on a search engine. The downside of multitasking is that the focus of attention is redirected away from their primary task and, as Hembrooke and Gay [2003] stated, our ability to engage in simultaneous task is, at best, limited, and at worst, virtually impossible.

Tabbed Interfaces

A tabbed interface is one that allows multiple documents to be contained within a single window, using tabs as a navigational widget for switching between sets of documents. That being said, there is a fuzzy boundary for distinguishing between a tabbed interface and an operating system taskbar, in the sense that both allow to group application instances and switch between them. From this definition, it is clear that one can interchange both "documents" and "window" by "pages" and "browser", respectively, to refer more precisely to the Web domain. Today all major web browsers feature a tabbed interface, so this figure is expected to be well understood by users worldwide.

Parallel Browsing

By providing tabs, web browsers have started supporting parallel browsing, allowing users to engage multiple concurrent pages simultaneously [Dubroy and Balakrishnan, 2010]. The current active tab is a *foreground task* and thus it has the user attention, while other tabs or windows may be loading in the background or contain information that is not yet needed [Huang and White, 2010]. Typical browsing flow may then be interrupted by tab switches to visit pages in other tabs. However, the notion of switching between sets of pages can be augmented to switching also between sets of (other) desktop applications. For example, when browsing for research purposes it is usual having also opened a PDF viewer, a file explorer, and a text editor; and alternate between them during the course of the browsing session. These activities, besides of not being explicit features of parallel browsing, may however influence our browsing behavior and therefore they should be taken into account. The effect of parallel browsing suggests that the user focus can no longer be simply seen as the difference in time between two successive page requests.

4.1.2 The Costs of Attention Shifts

There is a long history in the literature examining the allocation of attentional resources (e.g., Hansen [1991]; Janzen and Vicente [1998]; Ma and Kaber [2006]; McFarlane [1999]). Cutrell et al. [2000] summarized the field by outlining implications for design and discussing the perceived difficulty of switching back to tasks. Mark et al. [2005] discovered that more than a half of goal-oriented sessions are interrupted regularly by activities such as co-worker conversations, virus scanner pop-ups and instant messages. Iqbal and co-authors developed

tools for supporting interruption management by notification cues [Iqbal and Bailey, 2007], as well as detecting and differentiating breakpoints during task execution [Iqbal and Horvitz, 2007]. They found that task suspensions may result in more than two hours of time until resumption. Users are susceptible to overload, making thus user attention and workflow both delicate and difficult to maintain, especially when interruptions occur or work is divided across sessions [Humm, 2007].

Memory is highly selective, and the selection processes are determined by the interplay between task processing demands and UI design [Oulasvirta, 2004]. Interruptions lead to disruptive shifts in thinking, and understanding the hidden costs of multitasking may help people to choose strategies that boost their efficiency, such as the approaches we present in the next section or, depending on the application domain, related work like [Ashdown et al., 2005; Kern et al., 2010].

The findings that multitasking over different types of tasks can reduce productivity [Rubinstein et al., 2001] is further supported by the single channel theory, which suggests that the ability of humans to perform concurrent mental operations is limited by the capacity of a central mechanism [Kahneman, 1973; Schweickert and Boggs, 1984]. Therefore, multitasking may seem efficient at a first glance but it may actually take more time in the end and lends itself to more errors. Multitasking has been also studied on mobile devices [Karlson et al., 2010; Leiva et al., 2012; Oulasvirta et al., 2005]. Concretely, Leiva et al. [2012] looked into the cost of mobile application interruptions on task completion time at scale and "in the wild". They found that unintended interruptions caused by incoming phone calls can delay completion of a task by up to 4 times in comparison to when the user was not interrupted.

Returning to the Web domain, with the ubiquitous use of tabbed browsers, keeping multiple pages open in the same browser window has become possible, being an efficient alternative to switching between browser application instances [Gupta, 2009]. Although switch costs may be relatively small here [Mayr and Kliegl, 2000], sometimes just a few tenths of a second per switch, they can add up to large amounts when people switch repeatedly back and forth between tasks [APA, 2006]. What is more, often the greater the number of tabs or applications open at once, the higher the user's cognitive overload. To cope with this issue, we propose leveraging implicit interactions to guide visual search and therefore try to speed up the resumption of (browsing) tasks.

4.1.3 Strategies to Ease Multitasking

A clear approach to reach these goals is helping the user regain the context of the deferred application when it is resumed. Some authors, e.g., Johnson [2010], suggested to give pertinent visual cues as a help for easing the recovery from the interruption. Iqbal and Horvitz [2007] offered two directions in this

regard: reminding users of unfinished tasks and assisting them in efficiently recalling task context. In addition, we suggest either helping the user to maintain the context while switching to another application, or to support regaining context when returning to the interrupted application. In general, inspired by previous approaches of the interruptions community, we can distinguish between *preventive* (preparing the user for being interrupted, c.f. Trafton et al. [2003]) and *curative* (supporting the user after being interrupted, c.f. Iqbal and Horvitz [2007]) strategies.

Preventive: Preparation for Being Interrupted

This strategy states that, when a task interruption occurs, the user should be prepared to leave the current task. For instance, on mobile applications, when a incoming phone call occurs, the caller usually waits on the line for some seconds. Postponing the call a bit more (say, 500 ms) might provide time to give the user an auditory/visual/haptic signal that soon the phone application will pop-up [Leiva et al., 2012]. This way, the user would be able to save a mental state and keep in mind the recently interrupted application before he is interrupted.

In a similar vein, on desktop applications, notifications often appear at the corner of the screen, causing the user to move the focus of attention to the notification. Based on the previous idea, highlighting the window decorations might also provide the user with the possibility to take a subconscious snapshot of his most recent action before switching the current task.

Curative: Guidance for Going Back into Tasks

In this case, the user has been interrupted and as such there is no chance to provide feedback to leave the current task. Then, when the user resumes the previously interrupted application, she has to reallocate cognitive resources, which becomes increasingly difficult if the resource demands were high to begin with [Iqbal and Horvitz, 2007].

Therefore, this strategy states that the user should be given some help to be able to immediately (and easily) continue with the previous task. This can be achieved by automatically leaving a visual on-screen cue such that the user could remember at any time to which task she is switching back. For example, the system can show the last focus of interaction, in order to guide the user to the screen position before the interruption took place (see, e.g., Figure 4.1). Alternatively, when returning to the interrupted application, the system could replay the last N milliseconds of UI interactions, to give a hint of what she was doing before the interruption.

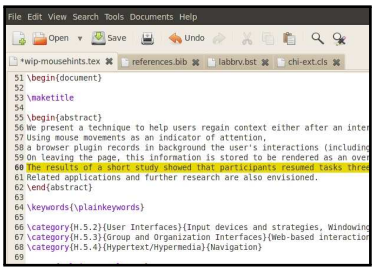


Figure 4.1: A usual approach for easing attention shifts in tabbed interfaces (in this case, a text editor). The last edited line is automatically marked by highlighting the text background, so when the user switches back to the current tab she can realize faster where she left writing.

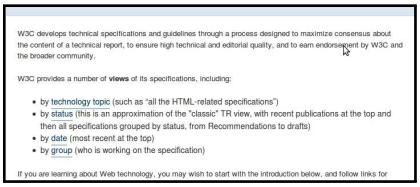
4.2 MouseHints

Kern et al. [2010] showed that some users, in order to keep track of where they were, tended to use the mouse cursor as a marker or to highlight the last line of a text paragraph. A similar approach is implemented in some text editors (see Figure 4.1). We exploit this notion in web browsing to remove the need of having to explicitly find a placeholder and/or actively manipulate it, without requiring additional hardware or any special setting.

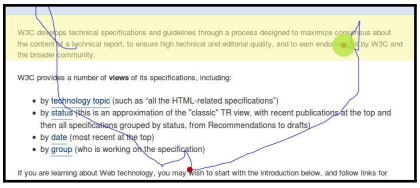
System Basis Only one web page and a corresponding tab representing it can be active at the same time in a browser window. Tapping this fact, our system tracks in the background the mouse activity in the current tab. Upon switching such a tab back, the system "hints" a subset of the last cursor movements (30 seconds by default), highlighting the last interacted element and the last cursor position (see Figure 4.2). Then, the rendered layer fades out in 500 ms (Figure 4.3).

User-System Interaction Protocol When the user selects a browser tab, a `focus` event is triggered and MouseHints records the position of the cursor every time she moves the mouse. When the user switches to another tab, two browser events are fired sequentially: a `blur` event from the old tab and a `focus` event from the new (now current) tab. MouseHints thus stops recording in the old tab and begins to track the activity in the current tab. When the user switches back to a previously visited tab, mouse data are overlaid on top of the HTML content. One may note that if the user switches to a desktop application, only a `blur` event can be detected. However, when switching back to the web browser, a `focus` event will be triggered, therefore enabling MouseHints again.

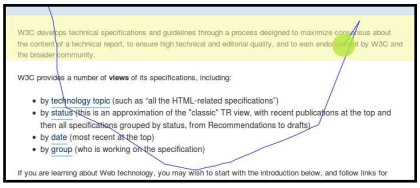
Implementation MouseHints was developed as a Firefox extension since such browser has a powerful mechanism that made it relatively easy to code and test. The browser interface was structured in XUL (XML UI Language). Both the logic and tracking algorithms were both written entirely in JavaScript. The visualization was coded in HTML5 throughout the `canvas` element, supported since version 1.5 of that browser.



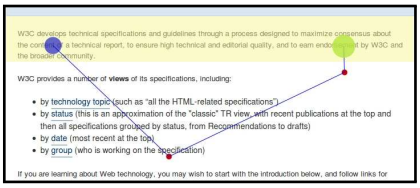
(a) Original test page.



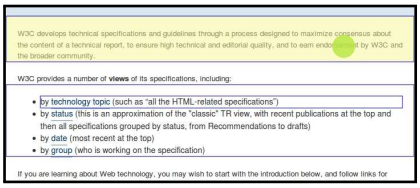
(b) Raw trajectory



(c) Digest



(d) Clustering



(e) DOM history

Figure 4.2: Visualization options for displaying the same mouse track. The right-most (green) circle represents the last cursor position, while smaller (red) circles represent mouse clicks. The bounding box of the last interacted HTML element is also highlighted. [4.2a] Original page, with no overlays. [4.2b] Event-based visualization. [4.2c] Velocity-threshold identification. [4.2d] WKM algorithm. [4.2e] An n-best list of hovering frequency.

Visualization We decided to represent the mouse cursor trail in a reasonable fashion while unobtrusively highlighting the last interacted HTML element. We developed a generic DOM selector that translated the mouse activity (e.g., hovering, clicking) into CSS selectors, so that the system could draw the corresponding bounding box of such interacted elements. Additionally, we implemented four different mouse path visualization options:

1. The raw mouse trail (Figure 4.2b).
2. A "digest" of the original trajectory (Figure 4.2c).
3. Clusters of mouse coordinates (Figure 4.2d).
4. A DOM-only visualization (Figure 4.2e).

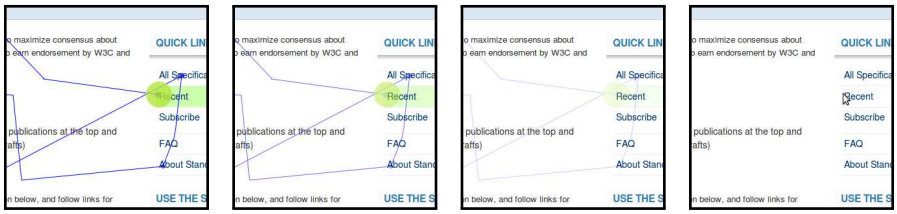


Figure 4.3: Visualization example. The overlay fades out in 500 ms, allowing for regular interaction with the page.

4.3 Evaluation

In order to evaluate our tool, we showed the visualization options (Section 4.2) to 6 participants and let them vote which one they preferred. The option that most people selected was number 2, so we used it for the test. Our hypothesis was that using MouseHints should benefit the users in terms of visual orientation in parallel browsing, i.e., faster task resumption and work completion by having the mouse interactions as a visual remainder.

Participants 36 unpaid volunteers (11 females) were recruited via email advertising. They were told to participate remotely in a study that would measure their reaction times while browsing. All of them were regular computer users accustomed to using browser tabs, aged 19 to 45 (M=25.5).

Apparatus We developed two Firefox extensions: the MouseHints application and a very basic logging system with the routines of the study. Half of the participants were asked to install both extensions on their computer. The other half of the users, who were not aware of the existence of MouseHints, installed the logging extension.

Design A between-subjects design was employed, with half of the subjects performing the tasks in only one condition (18 in the control group and 18 in the experimental group, respectively). The outcome measures were task success, time for task resumption, and time for task completion.

Procedure Each user performed two tasks, which were common to both groups. Each task took them about 5 minutes to perform in average, as it was dependent on each participant's browsing capabilities. The evaluation was done remotely, to allow subjects to browse in their own working environments. The tasks consisted of searching information for different topics (to mitigate possible learning effects between tasks); e.g., "what is the minimum number of face turns needed to solve a Rubik's Cube?" or "find the name of the last chapter of the book entitled El Quijote". Participants had to interrupt normal navigation

flow to play a popular game[1] in a dedicated browser tab. Such a game, despite being quite straightforward, required a lot of visual attention: the user had to click the last-born circle on each level (Figure 4.4). The conditions were browsing in a normal environment (control), and with the help of MouseHints (experimental).

(a) Level 12 **(b)** Transition **(c)** Level 13

Figure 4.4: While browsing, participants were eventually interrupted to play a game.

To measure how visual attention differed between both groups, at least two tabs had to be opened: one with the game and other with a regular web page. After a random delay between 20 to 40 seconds, the browser changed the focus of navigation from the current tab to the game tab, and users had to resume playing. After another delay, the browser changed the focus to another tab, which was randomly chosen from all opened tabs, to stress the users' cognitive load during the test. We measured the time for task resumption (first time to move the mouse inside the page) and time for task completion (total browsing time) for all opened tabs. Users were told to close their browser when a task goal was achieved—this allowed us to easily post-process their data.

In both conditions data were saved as timestamped event sequences in the local file system. In order to preserve the user's privacy, URLs were converted to MD5 hashes and data were stored in plain text format. This way, participants could verify that their data were sufficiently anonymized, and could also review what kind of information the extension was gathering. Then they were asked to submit the log files via email.

4.3.1 Results

We report measures on the three areas suggested by the ISO 9241-11 standard: *effectiveness* (completion rates and errors), *efficiency* (time on task resumption and completion), and *satisfaction* (subjective opinions on using the system).

Study on Effectiveness

We used a Pearson's chi-square test for this study. The nominal outcomes were task success/failure, measured by assigning 1 point each time the goal was achieved (based on the manual revision of user comments that were submitted

[1] http://tubegame.com/camera_mind.html

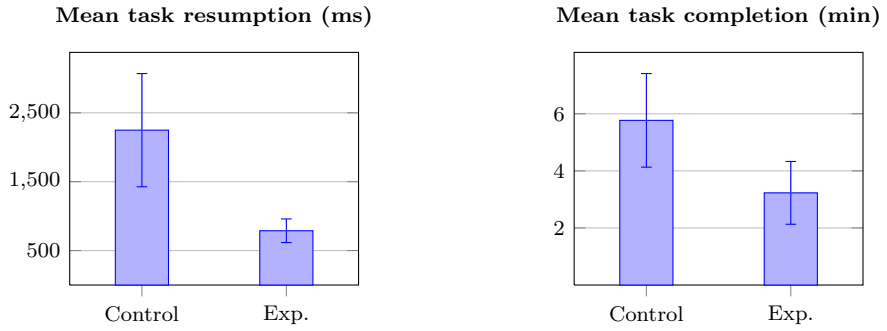


Figure 4.5: Between-groups efficiency comparison. Error bars denote 95% confidence intervals.

Study	Condition	M	SD	Mdn	Min	Max
Resumption (ms)	Control	2248.31	1778.89	2363	720	6566
	Experimental	788.11	371.42	791.5	345	1647
Completion (min)	Control	5.77	3.55	6.8	1.34	14.63
	Experimental	3.23	2.39	3.9	0.74	8.37

Table 4.1: Summary of efficiency results in both conditions.

by email). All participants excepting one user from the control group were able to finish the assigned tasks, concluding that there were no statistically significant differences in effectiveness between both groups ($\chi^2_{(1,N=36)} = 1.09$, $p = .29$, two-tailed). This result was not surprising. In fact, MouseHints is just an interaction assistant and, as expected, the user's success did not strongly depend on using this system for achieving their goals.

Study on Efficiency

In this case we used a Kolmogorov-Smirnov test, since normality assumptions did not hold. The continuous outcomes were time for task resumption and time for task completion. We used the median as central tendency measure for reducing the influence of outliers. As predicted, participants were found to be considerably faster in task resumption with MouseHints (Mdn = 791.5 ms) than without (Mdn = 2363 ms), $D = 0.72$, $p < .001$, two-sided hypothesis. We achieved similar conclusions regarding task completion (Mdn = 3.9 minutes with MouseHints; Mdn = 6.8 minutes without), $D = 0.5$, $p < .05$, two-sided hypothesis.

Study on Satisfaction

Participants from the experimental group submitted an online System Usability Scale (SUS) questionnaire [Brooke, 1996] after finishing the study. A Likert scale, from 1 (strongly disagree) to 5 (totally agree), was used to rank ten questions. SUS reported a composite measure of the overall usability of the system. The result was a score of 87.6, indicating that people indeed liked using MouseHints. (SUS scores range between 0 and 100).

The form attached to the online questionnaire allowed users to submit free comments and ideas. A frequently reported comment among participants in the experimental group was that MouseHints was considered helpful. Moreover, participants often mentioned the advantage of saving time and easing task resumption (12 users out of 18). Eight people liked the aid to memory of not having to remember what they previously did with the mouse in a page.

4.4 Discussion

Spink et al. [2004] raised the research question "how might multitasking be supported by web systems and interfaces?". MouseHints is an attempt to do so, although many other implications derived from (possible) further development are envisioned in this section.

Implications for Web Browsers MouseHints uses browser events to detect task switching and also track user interactions. However, our client-side implementation could provide the user with additional analysis features. Consequently, the browser could work as a personal organizer, prioritizing and reordering tabs according to browsing usage. What is more, rather than only dealing with explicit behavior information such as user history, web browsers could combine implicit interaction information of cursor data to suggest, e.g., already visited URLs when typing in the address bar.

Implications for Search Engines and Websites MouseHints could also have a number of implications for search engine design, in particular for inferring user interest. Our approach is a standalone client-side (offline) solution. We argue however that, by enabling some kind of server-side communication, cursor data could be sent for further analysis. In a public setting, the aggregation of other people's interactions may provide a valuable asset. Consequently, we could deploy large-scale studies about (contextualized) user behavior remotely, i.e., where the user is not physically present.

Furthermore, websites could also benefit from a rich understanding about their users. To date most theories on browsing behavior are based solely on the study of patterns from server's access logs [Leiva and Vidal, 2010]. However, the context of actions is a key issue for describing the surrounding facts that

add meaning to Web usage. Thus, combined with some analytic tools, we believe that MouseHints could contribute to achieve this goal.

Implications for User Interfaces Humans have remarkable perceptual abilities that are greatly underutilized in most current interface designs. Users can scan, recognize, and recall images rapidly, and can detect subtle changes in size, color, shape, movement, or texture [Schneiderman and Plaisant, 2005]. The visual elements together with the faded animations used in MouseHints serve as *bottom-up* stimuli that effectively capture user attention, improving reaction times and motor responses. So, these concepts can be applied to a broad range of UIs that could benefit from a user interaction model. For instance, it would be possible to implement a MouseHints-like agent in a tabbed application or even in the window manager of the operating system. We believe that incorporating related visual cues in traditional UIs should help the user while multitasking.

Implications for Electronic Devices MouseHints could also be used on mobile phones or tablets, e.g., in situations where the user should halt an application because of a phone call or a push notification. Additionally, in a higher level, one could implement our event detection method (Section 4.2) using accelerometer data, providing thus intelligent monitoring capabilities. For instance, it would allow mobile users to resume a game after leaving the device over a table because of an interruption.

Other Application Fields We believe this work is just a small though significant sample of the wide possibilities of tracking implicit interaction to ease task switching. Some related applications that could be implemented based on this technology include performance evaluation (e.g., compare motor skills or pointing abilities within a UI), user modeling (e.g., extract interaction features from the raw data and characterize user profiles), or self-adapting UIs (e.g., employ interaction data for rearranging layout elements based on each user's needs), among others.

Limitations First of all, participants performed tasks in an uncontrolled environment and without experimenter supervision. That could explain the variability in the gathered data (see Table 4.1), maybe due to potential outside distractions, or also because some tabs could not be relevant to the assigned task. Second, our approach is not suitable for the user that does not use the mouse (or a similar pointing device) at all while browsing the Web. In addition, there are situations where the eye and the mouse are not in sync; and we believe that our approach may not be much useful if such behavior happens frequently. Clearly, users who move the pointing device according to their focus of attention may be the most benefited target from MouseHints. Third, gathered data comprised about ten minutes of task execution data for each user. It would be interesting nevertheless to evaluate the effects of MouseHints in a large-scale study, where users will probably be more accustomed to the

system. Finally, users can assist web browsing by using more advanced I/O devices such as speech recognizers or eye trackers. Therefore, we encourage MouseHints to be used in combination with such systems, since we believe they all are complementary.

4.5 Conclusions and Future Work

We have presented MouseHints, a tool that aims to minimize the negative effects of interruptions while browsing, by providing adaptive memory cues about previous interactions and thus easing task resumption. MouseHints uses a combination of very basic infographics to draw user attention to the location of previously interacted areas on screen.

This chapter has described both the basic ideas behind our motivation as well as an implementation of this approach. Experimental results show that MouseHints is a promising technique for guiding visual search on complex interfaces. We believe the concept behind MouseHints may be used in different contexts that require multitasking and task switching, such as interacting with traditional (windowed or tabbed) desktop applications and even with mobile devices or electronic products. Our system may also be useful for visually complex tasks, such as scanning a busy display or navigating infographics in large screens.

Regarding the visualization of mouse trajectories, new strategies are being devised; concretely a hybrid method that incorporates clustering plus DOM history. This will be definitely a focus of future work.

Finally, MouseHints is by no means a supplement to any other methods to support multitasking, but rather an encouraging complementary tool. We believe that other sources based on implicit interaction should be taken into account, such as eye-gaze data or head movements. This topic, as well as exploring novel applications of MouseHints, will be considered for further research.

Bibliography of Chapter 4

E. Adar, J. Teevan, and S. T. Dumais. Resonance on the web: Web dynamics and revisitation patterns. In *Proceedings of the 27th international conference on Human factors in computing systems (CHI)*, pp. 1381–1390, 2009.

APA. Multitasking - switching costs. Available at http://www.apa.org/research/action/multitask.aspx, 2006. Retrieved August 1, 2010.

M. Ashdown, K. Oka, and Y. Sato. Combining head tracking and mouse input for a GUI on multiple monitors. In *Proceedings of extended abstracts on Human factors in computing systems (CHI EA)*, pp. 1188–1191, 2005.

J. Brooke. SUS: A "quick and dirty" usability scale. In P. Jordan, B. Thomas, B. Weerdmeester, and A. McClelland, editors, *Usability Evaluation in Industry*. Taylor and Francis, 1996.

M. D. Byrne, B. E. John, N. S. Wehrle, and D. C. Crow. The tangled web we wove: A taskonomy of WWW use. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pp. 544–551, 1999.

E. B. Cutrell, M. Czerwinski, and E. Horvitz. Effects of instant messaging interruptions on computing tasks. In *Proceedings of extended abstracts on Human factors in computing systems (CHI EA)*, pp. 99–100, 2000.

P. Dubroy and R. Balakrishnan. A study of tabbed browsing among mozilla firefox users. In *Proceedings of the 28th international conference on Human factors in computing systems (CHI)*, pp. 673–682, 2010.

D. Fong. Enhancing multitasking to enhance our minds. Available at http://daniellefong.com/2008/08/24/enhancing-multitasking-to-enhance-our-minds/, 2008. Retrieved August 1, 2010.

A. Gupta. Shiftbrowse: Context switch. Available at http://lcc.gatech.edu/%7Eagupta31/shiftbrowse/?p=33, 2009. Retrieved August 1, 2010.

C. M. Hansen. *Allocation of attention in dual pursuit tracking*. PhD thesis, Stanford University, 1991.

H. A. Hembrooke and G. K. Gay. The laptop and the lecture: The effects of multitasking in learning environments. *Journal of Computing in Higher Education*, 15(1):46–64, 2003.

J. Huang and R. W. White. Parallel browsing behavior on the web. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia (HT)*, pp. 13–18, 2010.

K. Humm. Improving task switching interfaces. Tech. Report COSC460, University of Canterbury, 2007.

S. T. Iqbal and B. P. Bailey. Understanding and developing models for detecting and differentiating breakpoints during interactive tasks. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pp. 697–706, 2007.

S. T. Iqbal and E. Horvitz. Disruption and recovery of computing tasks: field study, analysis, and directions. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pp. 677–686, 2007.

M. E. Janzen and K. J. Vicente. Attention allocation within the abstraction hierarchy. *International Journal of Human-Computer Studies*, 48(4):521–545, 1998.

J. Johnson. *Designing with the mind in mind*. Morgan Kaufman, 2010.

D. Kahneman. *Attention and Effort*. Englewoods Cliffs, Prentice Hall, 1973.

A. K. Karlson, S. T. Iqbal, B. Meyers, G. Ramos, K. Lee, and J. C. Tang. Mobile taskflow in context: A screenshot study of smartphone usage. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pp. 2009–2018, 2010.

D. Kern, P. Marshall, and A. Schmidt. Gazemarks: Gaze-based visual placeholders to ease attention switching. In *Proceedings of the 28th international conference on Human factors in computing systems (CHI)*, pp. 484–489, 2010.

C. König, M. Bühner, and G. Mürling. Working memory, fluid intelligence, and attention are predictors of multitasking performance, but polychronicity and extraversion are not. *Human Performance*, 18(3):234–266, 2005.

L. A. Leiva and E. Vidal. Assessing user's interactions for clustering web documents: a pragmatic approach. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia (HT)*, pp. 277–278, 2010.

L. A. Leiva, M. Böhmer, S. Gehring, and A. Krüger. Back to the app: The costs of mobile application interruptions. In *Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI)*, pp. 291–294, 2012.

R. Ma and D. B. Kaber. Presence, workload and performance effects of synthetic environment design factors. *International Journal of Human-Computer Studies*, 64(6):541–552, 2006.

G. Mark, V. M. Gonzalez, and J. Harris. No task left behind? examining the nature of fragmented work. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pp. 321–330, 2005.

U. Mayr and R. Kliegl. Task-set switching and long-term memory retrieval. *Journal of Experimental Psychology*, 26(5):1124–1140, 2000.

D. C. McFarlane. Coordinating the interruption of people in human-computer interaction. In *Proceedings of the IFIP Conference on Human-Computer Interaction (INTERACT)*, pp. 295–303, 1999.

A. Oulasvirta. Task-processing demands and memory in web interaction: A levels-of-processing approach. *Interacting with Computers*, 16(2):217–241, 2004.

A. Oulasvirta, S. Tamminen, V. Roto, and J. Kuorelahti. Interaction in 4-second bursts: the fragmented nature of attentional resources in mobile HCI. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pp. 919–928, 2005.

J. S. Rubinstein, D. E. Meyer, and J. E. Evans. Executive control of cognitive processes in task switching. *Journal of Experimental Psychology*, 27(4):763–797, 2001.

B. Schneiderman and C. Plaisant. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 4th edition, 2005.

R. Schweickert and G. J. Boggs. Models of central capacity and concurrency. *Journal of Mathematical Psychology*, 28(3):223–281, 1984.

A. Spink, M. Park, B. J. Jansen, and J. Pedersen. Multitasking during web search sessions. *Information Processing and Management: an International Journal*, 42(1):264–275, 2004.

J. G. Trafton, E. M. Altmann, D. P. Brock, and F. E. Mintz. Preparing to resume an interrupted task: effects of prospective goal encoding and retrospective rehearsal. *International Journal of Human-Computer Studies*, 58(5):583–603, 2003.

M. Viermetz, C. Stolz, V. Gedov, and M. Skubacz. Relevance and impact of tabbed browsing behavior on web usage mining. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pp. 262–269, 2006.