Universitat Politècnica de València
Departament de Sistemes Informàtics i Computació

Diverse Contributions to Implicit Human-Computer Interaction

by Luis A. Leiva

A thesis submitted in fulfillment for the
degree of Doctor of Philosophy in Computer Science

supervised by
Prof. Roberto Vivó and Prof. Enrique Vidal

November 8, 2012

PhD Thesis
Available online at http://personales.upv.es/luileito/phd/.

Typesetted in LaTeX (actually a mixture of TeX and LaTeX 2_ε).
Cover design by Luis A. Leiva. Iceberg photography © Ralph A. Clevenger (http://www.ralphclevenger.com, reproduced with permission).

Most parts of this work were supported by the Spanish Ministry of Science and Education (MEC/MICINN) under the research programme MIPRCV: "Consolider Ingenio 2010" (CSD2007-00018). Other parts have been also supported by the project TIN2009-14103-C03-03 and CasMaCat Project 287576 (FP7 ICT-2011.4.2).

Board Committee

Member, Reviewer .....

Prof. Albrecht Schmidt
Universität Stuttgart

Member, Reviewer .....

Prof. Antonio Krüger
Universität des Saarlandes

Member, Reviewer .....

Dr. Toni Granollers
Universitat de Lleida

President .....

Prof. Filiberto Pla
Universitat Jaume I

Secretary .....

Dr. M. Carmen Juan
Universitat Politècnica de València

Valencia, November 8, 2012

Abstract / Resumen / Resum

While interacting with computer applications, we submit an important amount of information unconsciously. By studying these implicit interactions we can better understand what characteristics of user interfaces add benefit (or not), thus deriving design implications for future systems.

The main advantage of processing implicit input data from the user is that every interaction with the system can contribute to enhance its utility. Additionally, such an input removes the cost of having to interrupt the user to submit explicit information that can be little related to the purpose of using the system. On the contrary, sometimes implicit interactions do not provide clear and concrete data. As such, how this source of information is managed deserves a special attention.

This research is two-fold: 1) to apply new perspectives both to the design and the development of tools that can take advantage from user's implicit interactions, and 2) provide researchers with a series of evaluation methodologies of interactive systems that are ruled by such implicit input methods. Five scenarios are discussed to illustrate the feasibility and suitability of this thesis framework. Empirical results with real users show that tapping implicit interactions is a useful asset to enhance computer systems in a variety of ways.

Al interactuar con aplicaciones informáticas, proporcionamos inconscientemente una cantidad importante de información. Mediante el estudio de estas interacciones implícitas es posible entender qué características de la interfaz de usuario son beneficiosas (o no), derivando así en implicaciones para el diseño de futuros sistemas interactivos.

La principal ventaja de procesar datos de entrada implícitos del usuario es que cualquier interacción con el sistema puede contribuir a mejorar su utilidad. Además, dichos datos eliminan el coste de tener que interrumpir al usuario para que envíe información explícitamente sobre un tema que en principio no tiene por qué guardar relación con la propia intención de utilizar el sistema. Por el contrario, en ocasiones las interacciones implícitas no proporcionan datos claros y concretos. Por ello, hay que prestar especial atención a la manera de gestionar esta fuente de información.

El propósito de esta investigación es doble: 1) aplicar una nueva visión tanto al diseño como al desarrollo de aplicaciones que puedan aprovechar consecuentemente a las interacciones implícitas del usuario, y 2) proporcionar una serie de metodologías para la evaluación de dichos sistemas interactivos. Cinco escenarios sirven para ilustrar la viabilidad y la adecuación del marco de trabajo de

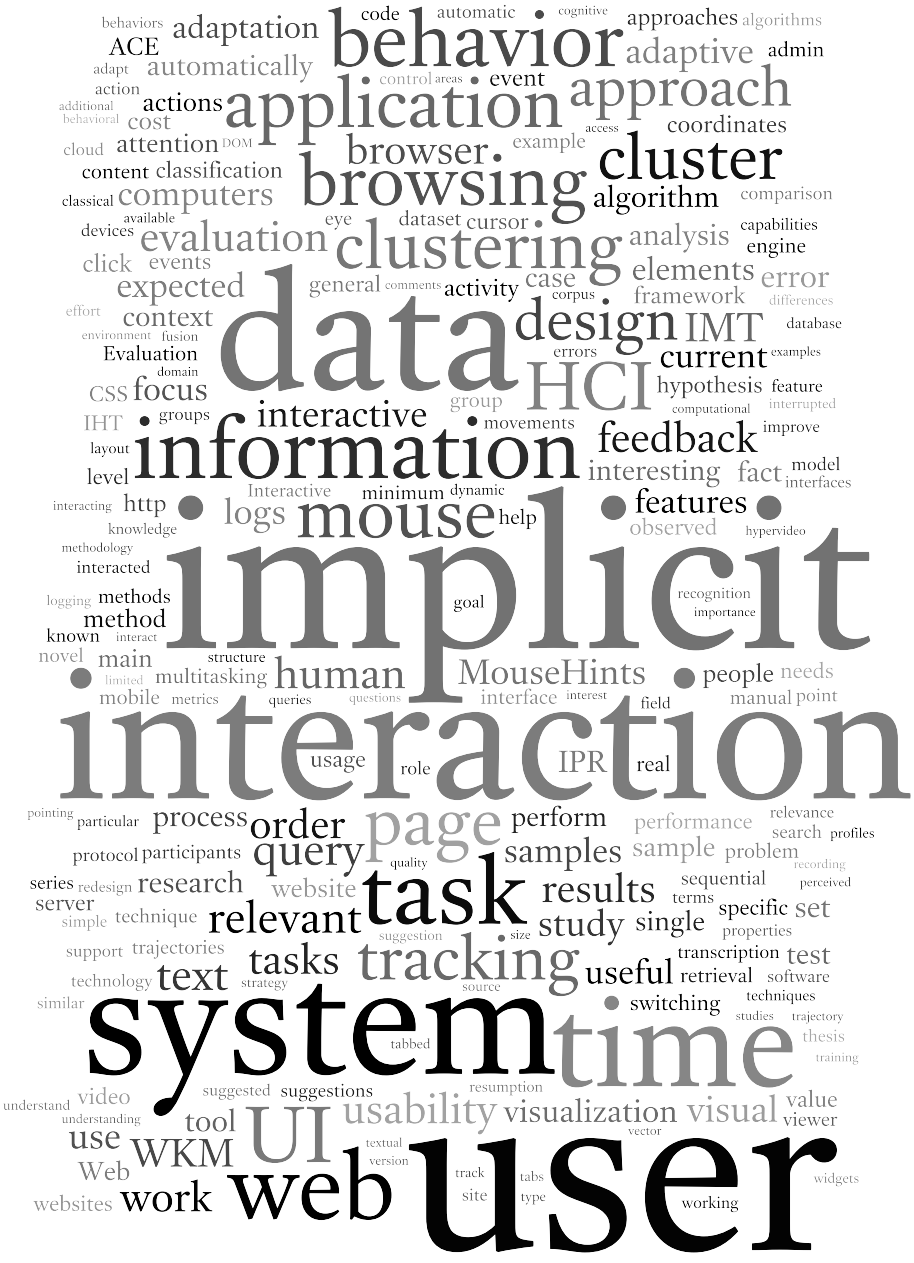
la tesis. Resultados empíricos con usuarios reales demuestran que aprovechar la interacción implícita es un medio tanto adecuado como conveniente para mejorar de múltiples maneras los sistemas interactivos.

Quan interactuem amb aplicacions informàtiques, proporcionem inconscientment una quantitat important d'informació. Mitjançant l'estudi d'aquestes interaccions implícites és possible entendre quines característiques de la interfície d'usuari són beneficioses (o no), i derivar així en implicacions per al disseny de futurs sistemes interactius.

El principal avantatge de processar dades d'entrada implícites de l'usuari és que qualsevol interacció amb el sistema pot contribuir a millorar la seua utilitat. A més a més, aquestes dades eliminen el cost d'haver d'interrompre l'usuari perquè envie informació explícitament sobre un tema que en principi no té per què guardar relació amb la pròpia intenció d'utilitzar el sistema. No obstant això, a vegades les interaccions implícites no proporcionen dades clares i precises. Per tant, cal prestar especial atenció a la manera de gestionar aquesta font d'informació.

El propòsit d'aquesta investigació és doble: 1) aplicar una nova visió al disseny i al desenvolupament alhora d'aplicacions que puguen reaccionar conseqüentment a les interaccions implícites de l'usuari, i 2) proporcionar una sèrie de metodologies per l'avaluació d'aquests sistemes interactius. Cinc escenaris il·lustren la viabilitat i l'adequació del marc de treball de la tesi. Resultats empírics amb usuaris reals demostren que aprofitar les interaccions implícites és un mitjà adequat i convenient alhora per a millorar de múltiples maneres els sistemes interactius.

Keywords


behaviors
adaptation
code
automatic
cognitive
approaches
algorithms
behavior
ACE
adapt
automatically
control
areas
event
adaptive
admin
action
additional
actions
application
approach
behavioral
cost
access
coordinates
cloud
attention
DOM
example
browser
cluster
content
classification
browsing
algorithm
comparison
classical
computers
available
devices
eye
dataset
cursor
capabilities
evaluation
clustering
analysis
engine
click
events
elements
error
expected
general
comments
activity
case
corpus
framework
differences
effort
context
data
design
IMT
database
environment
fusion
errors
current
examples
Evaluation
domain
CSS
focus
HCI
hypothesis
feature
group
computational
interrupted
IHT
groups
interactive
movements
improve
layout
information
feedback
level
interesting
fact
model
interfaces
Interactive
minimum
dynamic
interacting
http
logs
mouse
features
help
knowledge
observed
hypervideo
methodology
implicit
interacted
logging
methods
goal
recognition
method
importance
known
interact
novel
main
structure
human
MouseHints
people
needs
limited
multitasking
mobile
metrics
queries
questions
interface
interest
field
manual
point
interaction
usage
role
IPR
real
pointing
particular
process
order
page
perform
performance
relevance
search
profiles
protocol
participants
query
quality
samples
sample
problem
recording
series
redesign
research
website
task
results
sequential
perceived
server
terms
simple
technique
relevant
study
single
specific
set
suggestion
size
properties
support
trajectories
tasks
tracking
transcription
test
technology
text
strategy
useful
retrieval
software
similar
source
switching
techniques
system
time
studies
trajectory
tabbed
thesis
training
understand
video
suggested
suggestions
resumption
tool
UI
usability
visualization
visual
value
understanding
viewer
use
WKM
vector
textual
Web
version
track
tabs
widgets
websites
work
web
user
site
type
working

Acknowledgments

Ya han pasado 5 años desde que inicié mi andadura por el mundo de la investigación, al matricularme en el programa de doctorado que ha dado lugar a esta tesis. Unas cuantas publicaciones y un sinfín de anécdotas dan buena parte de una breve pero intensa trajectoria predoctoral que he tenido la suerte de completar. En verdad me considero afortunado al respecto por una larga serie de razones, de las cuales comentaré a continuación aquellas que considero especialmente relevantes.

En primer lugar, me considero afortunado por haber tenido de directores de tesis no uno sino dos catedráticos de la talla de Roberto Vivó y Enrique Vidal. He de agradecer a Roberto por haber depositado su confianza en mi desde el primer momento en que le propuse hacer el DEA bajo su tutela. Me ha dado un margen de maniobra sin precedentes que me ha permitido evolucionar favorablemente como investigador. Por supuesto también he de agradecer a Enrique por darme la oportunidad de trabajar en el grupo PRHLT, lo que ha supuesto y sigue suponiendo un apasionante reto profesional. Es una suerte tenerlo como co-director del grupo—junto con Francisco Casacuberta, otra persona de la que uno solo puede hablar bien.

Desde aquí, mi más sincero agradecimiento a toda la gente con la que he tenido la oportunidad de trabajar durante todo este tiempo. En especial, quiero agradecer a las siguientes personas por haberme permitido participar en el desarrollo de unos novedosos prototipos: Verónica Romero (CATTI), Daniel Ortiz (IMT), Ricardo Sánchez (IPP), Mauricio Villegas y Roberto Paredes (RISE) y Alejandro Toselli (KWS). Mención de honor para Vicent Alabau, cuya creatividad y buen hacer parecen no tener límites. Hemos compartido muy buenos momentos, y sobre todo numerosas y fructíferas discusiones que se han materializado en importantes publicaciones conjuntas.

También quiero mencionar al resto de mis compañeros del ITI/DSIC, porque gracias a ellos el día a día en el entorno de trabajo es más que reconfortante. Así de repente me vienen a la mente Jesús González, Nico Serrano, Elsa Cubel, Antonio Lagarda, Esperanza Donat, Germán Sanchis, Jesús Andrés, Jorge Civera, José Ramón "maestro" Navarro y las nuevas generaciones: Paco Álvaro, Dani Martín-Albo, Vicent Bosch, Mercedes García, Joan Albert y Miguel del Agua, entre otros. A los que se me olvidan, quedan agradecidos por extensión ;)

I would also like to thank the primary reviewers of this thesis: Albrecht Schmidt, Antonio Krüger and Toni Granollers, who accepted without hesitation to review and join the board committee on the defense day. I have to admit that Albrecht has played inadvertently an important role in this thesis. Thanks to his research work, I started to get interested in the topic of the thesis. But also he introduced me to Antonio Krüger, whom I shared a great stay with at the DFKI, together with the people from IRL. I would also like to thank the secondary reviewers: Fabio Paternó, Antti Oulasvirta and Nuria

Oliver. Althought they did not get the chance to actually review this thesis, I have been lucky enough to enjoy their support. And of course, I must thank the rest of the board committee: Filiberto Pla (president), Mari Carmen Juan (secretary) and the alternate members José Miguel Benedí and Miguel Chover.

Por supuesto, quiero agradecer a mis amigos y a mi familia, en especial a mis padres y a mi hermana, porque la distancia geográfica que nos separa no ha evitado que me sigan dando todo su apoyo incondicional. Por último, y no por ello menos importante (¡ni mucho menos!) quiero agradecer muy especialmente a Bea Alonso por su infinita paciencia, sobre todo en los últimos tramos de la tesis. Ella es ahora mismo una de las personas más importantes en mi vida. A ella le dedico esta tesis.

Luis A. Leiva
November 8, 2012

Contents

Nomenclature

ACE	Adaptive CSS Engine
AJAX	Asynchronous Javascript And XML
API	Application Programming Interface
CSS	Cascading Style Sheet
DOM	Document Object Model
HCI	Human-Computer Interaction
HMM	Hidden Markov Model
HTML	HyperText Markup Language
HTR	Handwritten Text Recognition
HTTP	HyperText Transfer Protocol
IDL	Interface Definition Language
IGP	Interactive Grammatical Parsing
IHT	Interactive Handwritten Transcription
IMT	Interactive Machine Translation
IPR	Interactive Pattern Recognition
JS	JavaScript
JSON	JavaScript Object Notation
MT	Machine Translation
NLP	Natural Language Processing
NN	Nearest-Neighbor
POI	Probability Of Improvement
PR	Pattern Recognition
RISE	Relevant Image Search Engine
SQE	Sum of Quadratic Errors
SUS	System Usability Scale
TS	Trace Segmentation
UI	User Interface
WER	Word Error Rate
WSR	Word Stroke Rate
XML	eXtensible Markup Language
XUL	XML UI Language

“You can discover more about a person in an hour of play than in a year of conversation.”

Plato, 427–347 BC

Chapter 1

Introduction

Understanding how users behave has been (and certainly *is*) a longstanding subject of study in a really wide range of disciplines in science. Often, behavior needs to be measured, usually by directly asking the users. When interacting with computers, though, the intention of the user is mostly hidden. What is more, direct user feedback is notoriously unreliable most of the time. For instance, feedback regarding feelings, opinions, threats, etc. is strongly biased toward an individual perception; and hence it is hardly generalizable.

Fortunately, despite of the heterogeneity and dynamism inherent in user behavior, some actions are common to many individuals, and hence they can be recognized automatically. This kind of information can provide useful hints when designing interactive systems, which is the foremost motivation of this thesis, as discussed in this chapter.

Chapter Outline

1.1 Preamble: On User Behavior

Behavior refers to the actions or reactions of an object or organism, usually in relation to the environment. Behavior can be (sub)conscious, (c)overt, and (in)voluntary. In Human-Computer Interaction (HCI), behavior is the collection of responses exhibited by people, which are influenced by a diversity of factors; e.g., culture, attitudes, emotions, values, and/or genetics.

According to humanism, each individual has a different behavior. Observations about individual differences can thus inform the design of interfaces that are tailored to suit specific needs [Hwang et al., 2004]. Nevertheless, humans often show certain behaviors recurrently. In fact, some actions can be recognized automatically and therefore can provide useful hints when designing interactive systems. For example, when browsing a web page, if many users highlight the same text paragraph and copy it, then that text is supposed to be interesting, and hence the webmaster could consider giving it more prominence, e.g., by typesetting it in boldface.

Additionally, user behavior is not static but rather *dynamic* per se: preferences and attitudes change frequently over time. This fact can easily invalidate methods or theories that were developed not so many time ago, because of the temporary dependence of the evaluations that once supported them—for instance, think of the findings on electronic mail usage analysis reported thirty years ago by Hersh [1982]. Instead, measuring natural behavior gives a much more accurate picture of a user's immediate experience rather than asking him after a task is complete [Hernandez, 2007]. This way, behavioral (or biometric or interaction-based) measurements are theoretically more accurate than relying on explicit user feedback. They are indeed theoretically more accurate because, similar to everyday life body language, a certain behavior does not indicate always and universally the same inner state [Gellner et al., 2004]. So, depending on the task or its context, we can safely rely on this kind of measures or, on the contrary, acknowledge their limitations and combine them with other data sources.

1.1.1 Historical Background

According to behaviorism, behavior can be studied in a systematic and observable manner with no consideration of internal mental states [Cherry, 2006]. So, intentions are evidenced by exertions: users first focus and then execute actions. But, can behavior be measured? If not, then it could not be scientifically analyzed. Fortunately, this is not the case. In fact, instrumentation, i.e., automatic recording of user behavior within a system, has a long history in psychology. Its use in simple systems such as operant chambers (c.f. the Skinner box) helped to advance the study of animal (and, later, human) learning, revealing new patterns of behavior. Instrumentation was a key milestone in

HCI, since the field draws on cognitive psychology at its theoretical base. Over the last 25 years researchers have used instrumentation to better understand users and, consequently, to improve applications [Kim et al., 2008]. Computers are now found in most aspects of our daily life, and for some it is hard to even imagine a world without them.

Today, user interfaces (UIs) are one of the main value-added competitive advantages of computer applications, as both hardware and basic software become commodities. People no longer are willing to accept products with poorly designed UIs. So much so that notions of software products have been revisited with generalized psychology and physiology concepts in mind. For example, the standard ISO/TR 16982:2002 addresses technical issues related to human factors and ergonomics, to the extent necessary to allow managers to understand their relevance and importance in the design process as a whole.

Interaction design is often associated with the design of UIs in a variety of media, but focuses on the aspects of the interface that define and present its behavior over time, with a focus on developing the system to respond to the user experience and not the other way around. Designing interactive systems is about designing technology to maximize aspects of the interaction toward some goal [Bongard, 2010]. Interactivity, however, is not limited to technological systems. People have been interacting with each other as long as humans have been a species [Sinclair, 2011]. Therefore, interaction design can be applied to the development of any software solution, such as services and events. Ultimately, the design process must balance technical functionality and aesthetics to create a system that is not only operational but also usable and adaptable to changing user needs. Therefore, it is necessary to consider a multidisciplinary point of view to understand the role of human beings in computer science.

Finally, to close this very succinct historical context[1], we should mention the contributions to HCI of notable organizations such as the Interaction Design Foundation and ACM SIGCHI in USA or AIPO in Spain. Organizations like these are providing an international discussion forum through conferences, publications, workshops, courses and tutorials, websites, email discussion groups, and other services. For many of us, HCI is therefore enjoying a privileged position compared to other fields in computer science.

1.2 Implicit Interaction

Often, in HCI, behavior needs to be measured. Otherwise, how could we figure out if an application is really being used as intended? It is clear that user feedback is invaluable and, as such, usually behavioral data are gathered by directly asking the users. When interacting with computers, though, the intention of the user is mostly hidden [Hofgesang, 2006]. The activation of automatic goals,

[1] [Carroll, 2009] is a must-read in this regard.

and the physical traits of stimuli in our environment all influence our thoughts and behavior considerably, and often without our awareness.

What is more, direct user feedback is notoriously unreliable most of the time. For instance, feedback regarding feelings, opinions, threats, etc. is strongly biased toward an individual perception; and hence it is hardly generalizable—unless the size of the user sample is fairly substantial, of course, which is rarely the case in HCI studies (see, e.g., [Henze, 2011] for a quantitative comparison). Moreover, this kind of feedback must be acquired through some in-lab based methods, e.g., surveys, usability tests, cognitive walkthroughs, etc., and therefore requires to invest both time and money, which are often finite resources that eventually should be optimized.

In addition, to learn a user's interests reliably, intelligent systems need a significant amount of training data from the user. The cost of obtaining such training data is often prohibitive because the user must directly label each training instance, and few users are willing to do so [Goecks and Shavlik, 2000; Zigoris and Zhang, 2006]. Meanwhile, users expect a system to work reasonably well as soon as they first use the system. Thus, it is supposed that systems should work well initially with less (or none) explicit user feedback.

The social psychologist John A. Barg (1955–) stated that *one of the functions of consciousness is to select behaviors that can be automated and become unconscious.* In this context, researchers have elucidated new ways of expanding this notion to computers. As such, many different definitions (that largely overlap each other) have been independently proposed worldwide and thus are diffusely spread in the literature. For instance, implicit interaction is related to some extent to the following terms:

- Ubiquitous Computing [Weiser, 1993]
- Calm Technology [Weiser and Brown, 1996]
- Proactive Computing [Tennenhouse, 2000]
- Ambient Intelligence [Hansmann, 2003]
- Attentive Interface [Vertegaal, 2003]
- Perceptual Interface [Wilson and Oliver, 2005]

In the literature, implicit interaction is found to be cited, among others, as:

- Untold Feedback [Tan and Teo, 1998]
- Subsymbolic Behavior [Hofmann et al., 2006]
- Subconscious Awareness [Yoneki, 2006]
- Passive Actions [Grimes et al., 2007]
- Implicit Intentions [Kitayama et al., 2008]

Consequently, as pointed out by Oulasvirta and Salovaara [2004], the topic now seems to be in a state of conceptual balkanization, and it is difficult to get an

overall grasp of the field. This fact poses an additional difficulty when defining the topic precisely. From my research, however, I would probably recommend (as being most adequate) the definition of Schmidt [2000]:

> *An action performed by the user that is not primarily aimed to interact with a computerized system but which such a system understands as input.*

Implicit interactions are thus those actions that the user performs with little (or no) awareness. And, unsurprisingly, humans have an abundance of experience with implicit interactions; we successfully employ them in a daily basis without conscious thought. For example, we laugh when someone tells a joke that we like. In doing so, we are communicating to that person that we appreciate such a joke. Humans constantly exchange information about their environment, and so can do computers. Figure 1.1 depicts a framework that summarizes quite well a modern view of implicit interactions in HCI.

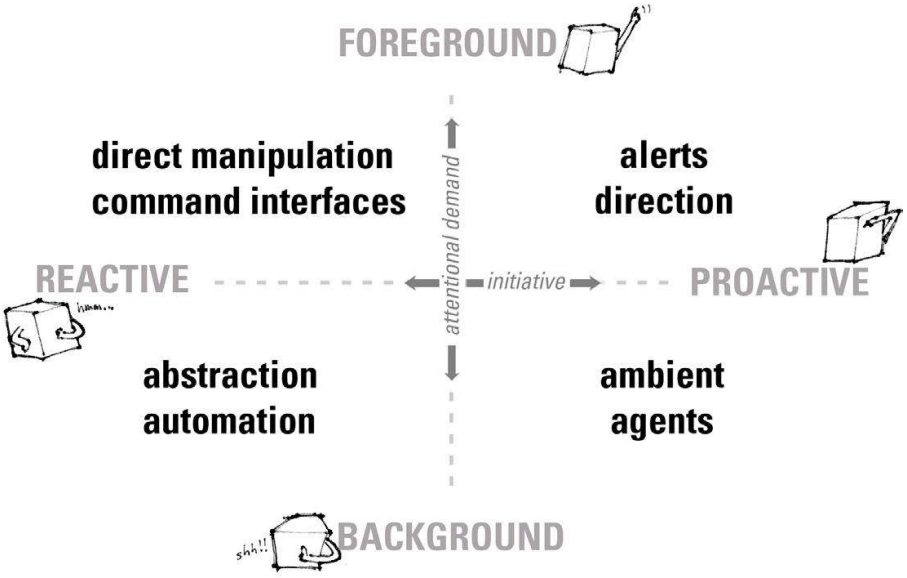


Figure 1.1: The implicit interaction framework [Ju and Leifer, 2008]. © Massachusetts Institute of Technology. Reproduced with permission.

As previously pointed out, the concept of implicit interaction is somewhat historically related to the ubiquitous computing (et al.) mantra: "the most profound technologies are those that disappear" [Weiser, 1999]. However, implicit interaction has a subtle but fundamental differentiation factor: is the user who takes the initiative to interact with the system. Therefore, ultimately the role of implicit interaction consist in leveraging as much information as possible derived from a natural user input, without requiring the user to be aware of the data the system needs to operate. This definitely has the capacity to make computers more useful and tailored to our needs.

1.2.1 Putting It All Together

The increasing use of technology—especially concerning to mobile devices and the Web—is changing our daily lives, not only in the way we communicate with each other or share information, but also *how* we relate to the environment. This entails new opportunities to transfer knowledge from one domain to another, by understanding that: *a*) implicit interactions offer a valuable source of information, and *b*) they can help to better manage user expectations.

By unobtrusively observing the user behavior we are able to learn functions of value. We can collect automatically generated training samples during a normal use, allowing for a collection of large datasets if deployed over the Web. This is interesting for many reasons. First, typical interactions with an application can involve many impasses, depending on the expertise of the user toward the application. Second, if such an application is intended to be used by an unknown user population, then it is very likely to involve ill-structured goals and tasks, and substantial influences from the content that is encountered while interacting [Card et al., 2001]. Third, classical approaches have relied on very simple measures such as time spent on a task or average number of clicks alone. These measures do not, however, provide any trace of the moment-by-moment cognition that occurs between regular interactions. If we are interested in developing detailed models of such cognition—for instance, to better understand how people's goals evolve, how people perceive and process the contents of an application, how and why they make decisions, and so on—then progress will be accelerated by having more detailed data of that cognition [Card et al., 2001].

Implicit interaction, as observed, requires no training and provides context for actions. As such, a wise knowledge of the limits, capabilities, and potential of implicit interaction in HCI provides an interesting theoretical basis for a systematic approach to analyzing, optimizing, and enhancing computer applications.

1.3 Aims and Goals of the Thesis

The central hypothesis of this research work is that *1*) there is a lot of information inherently encoded in user interactions, which *2*) can be measured and from which it is possible to extract meaningful knowledge, and therefore *3*) can be leveraged in a wide spectrum of applications and tasks. Virtually every chapter of the thesis is devoted to this notion, aiming to answer the same question: *How can implicit interaction be of help in computing systems?*

Other questions we try to answer include the following[2]. How can we exploit the potential of computer-based support to augment our daily activities? How

[2]See also http://www.ercim.eu/EU-NSF/DC.pdf

can we build systems in the face of uncertainty and partial knowledge? When do we try to predict the user and when do we let the user choose? How do we convey the system boundaries to the user?

This thesis is approached with a double-fold intent: *a*) researching on what characteristics can be inferred or leveraged from how users behave when interacting with computers, and *b*) deriving applications and implications to improve the utility of the systems that are meant to be used by people in a regular basis. There is a challenge, thus, in the way we can exploit this potential, in order to rethink how current technology may drive the dynamic environment of interactive systems. Through an exploratory research well beyond the classical (now interdisciplinary[3]) scope of HCI, this thesis will try to expand the body of knowledge on implicit interaction to related communities that rely to some extent on the user intervention, such as Cognitive Science, Infographics, Interactive Pattern Recognition, or Visual Design communities. This way, by exploring the role of implicit interactions in different domains and from different perspectives, not only a global vision of their importance is acquired; but specific solutions and working perspectives are proposed, discussed, and evaluated at different levels of understanding, depending on the specific task and the available resources. To do so, every chapter of this thesis has been conceived as a self-contained unit that in turn relates to the central topic of the thesis: the role of implicit interaction in HCI.

1.3.1 Organization and Contributions

This work has been divided into five illustrative scenarios, each one corresponding to a main chapter of this thesis, which are indeed the main contributions of the author to the field of implicit interaction. A brief overview of them is now advanced, although the reader can find a more detailed description in 'Thesis Overview' on page 9.

Chapter 2 showcases what probably is the most direct application to begin dealing with implicit interactions: visualization. An open source tool to understand browsing behavior is thoroughly described, providing also a real-world case study as an evidence of its utility. Most parts of this tool have been used to build other systems that helped to achieve the goals of this thesis. Chapter 3 presents a methodology designed to model the user in context, i.e., to find homogeneous groups of what a priori are different interaction behaviors, and also to automatically identify outliers. In addition, a novel revisitation of the K-means algorithm is presented to classify human actions in an unsupervised way. Chapter 4 discusses the problems when the focus of interaction changes from application to application, either unconsciously (e.g., a pop-up notification) or on purpose (e.g., multitasking). A technique to regain context is introduced in the domain of parallel browsing, and some directions are

[3] According to A. Oulasvirta, HCI has become so absurdly diverse and multi-multi-disciplinary that it is more aptly called *hyper-disciplinary*.

given to extend the same notion to mobile and desktop applications. Chapter 5 provides a novel approach to automatically redesign interface widgets. An appealing feature of such approach is that the method operates unobtrusively for both the user and the application structure. Although this is still ongoing work, with about a year of existence, the motivation of the technique has been empirically validated. Chapter 6 discusses the role of implicit interactions in Interactive Pattern Recognition applications, where the system and the user are expected to collaborate seamlessly. Four applications are examined: handwriting transcription, machine translation, grammatical parsing, and image retrieval. Finally, Chapter 7 wraps up the general conclusions of the thesis, remarking the main implications for design when implicit interaction is considered, and stating possible directions for further research. Last but not least, Appendix A enumerates the publications derived from this thesis.

1.3.2 Importance and Application Fields

Software applications in general and interactive systems in particular imply somewhat the understanding of their users. As previously discussed in Section 1.2, virtually any user-driven system can gain some benefit from implicit interaction analysis. Just to name a few of the possible application fields:

Usabiliy Testing Both remote and in-lab usability experiments are the primary source to evaluate the success of computer applications. Here, implicit interaction can help to unobtrusively analyze natural behaviors.

Data Mining If the experiments depicted above are, e.g., deployed over the Web, one can obtain vast quantities of data samples and perform readily prospective studies.

Performance Evaluation Related to the previous examples, a baseline control sample could be compared to a variety of test samples in real time, without interfering with the user experience.

Interface Analysis Determine which elements in the layout do attract the user interaction the most; again, without asking the users on purpose.

Gesture Recognition Use implicit features to convey meaning when drawing a picture (e.g., identify symmetries) or when handwriting (automatically isolate words or characters).

Usage Elicitation On the Web, spider bots behavior may greatly distort human usage patterns, hence it is critical to deal only with interaction data from real users.

Interaction Research Understanding human movement is a key factor to improve input devices as well as envision novel interaction techniques.

Behavior Prediction Usage data can presage not only how interfaces are likely to be used, but also which elements add value (or not) to the application.

Information Visualization Visualizing what users have done is a great aid to understand exactly how users behave and perform actions.

Biometrics Model behavior according to the usage of mouse, keyboard, eye-gaze, or other input devices for identifying users unequivocally.

Collaborative Filtering Discover usage profiles, involving the collaboration among multiple methods, viewpoints, data sources, and so on.

User Modeling Acquire information about a user (or a group of users) so as to be able to adapt their behavior to that user (or that group).

Multimodal Interfaces Leverage additional feedback signals that sometimes are unconsciously submitted to improve the utility of the system.

Self-Adapting UIs Use interaction data for re-arranging layout elements based on how users interact with them.

1.4 Thesis Overview

The following sections below introduce the contents that shall be later covered in the chapters of the thesis. It is worth mentioning that all systems developed in the context of this thesis are either web-based or closely related to the Web. The main reason is because currently people use web browsers more than any other class of desktop software on a daily basis. This situation has created a previously unparalleled level of user experience in a software niche [Edmonds, 2003]. Moreover, regarding to test new research methods and techniques, three reasons back up the need for driving research through web-based systems: *1*) the initial development time can be shorter, so the system is available to users earlier, *2*) continuous improvement is possible, without having to update or reinstall software, and *3*) real-world usage data can be obtained during the application life cycle.

1.4.1 Interactive Usability Evaluation

Besides conventional features such as performance and robustness, usability is now recognized as an important quality attribute in software development. Traditionally, usability is investigated in controlled laboratory conditions, by recruiting a (hopefully representative) user sample and often performing video recordings and surveys that are later reviewed. This requires an important investment in time and money, not to mention that processing user interaction data is, at a minimum, cumbersome. This chapter discusses the role of implicit

interaction when performing usability tests on websites; concretely, *a)* which kind of data can be gathered by observing the overt behavior of users, without relying on explicit feedback, *b)* how this data can be presented to the usability evaluator, and *c)* which questions can be answered by inspecting such data.

1.4.2 Behavioral Clustering

Behavioral clustering is a broad term that refers to the task of automatically labeling and classifying user behavior. Overall, clustering is a relevant method to identify sub-populations in a dataset, so that they can be represented by more compact structures for, e.g., classification and retrieval purposes. To this end, implicit interaction can provide current clustering methods with additional information. First, on the Web, fine-grained interactions can reveal valuable information (e.g., related to cursor movements, hesitations, etc.) that is not available in typical access logs. Second, in a general context, user behavior has an intrinsic *sequential* nature, which is not considered on current clustering analysis, that can be exploited to simplify the structure of the data. This chapter proposes two approaches to solve both drawbacks: *1*) a novel methodology to model websites, i.e., finding interaction profiles according to how users behave while browsing, and *2*) a novel clustering algorithm to deal with sequentially distributed data, whose suitability is illustrated in a human action recognition task.

1.4.3 Human Multitasking

We use different applications to multi-task the activities we do every day, even when browsing the Web; e.g. it is not unusual having multiple tabs or browser instances open at a time. People thus may cognitively coordinate simultaneous tasks through multiple windows or multi-tabbing, having many applications open at the same time and switching between them in any order. This chapter addresses how to reduce the overall cognitive load involved in switching among multiple windows during the course of typical information work. The chapter provides directions for designing mobile applications, where interrupted tasks usually have a high resumption cost. A method was implemented to illustrate a means to assist web browsing: using mouse movements as an indicator of attention, a browser plugin highlights the most recently interacted item as well as displaying (part of) the mouse path. An empirical study shows that this technique can help the user to resume and complete browsing tasks more quickly.

1.4.4 Adaptive User Interfaces

Adaptive systems accommodate the UI to the user, but doing so automatically is a non-trivial problem. Adaptation should be predictable, transparent, and

discreet, so that changes introduced to the UI do not confuse the user. Also, adaptation should not interfere with the structure of the application. This chapter presents a general framework to restyle UI widgets, in order to adapt them to the user behavior. The value of this methodology comes from the fact that it is suited to any application language or toolkit supporting structured data hierarchies and style sheets. As discussed, an explicit end user intervention is not required, and changes are gradually applied so that they are not intrusive for the user. The method is also extended as a technique to foster creativity, by suggesting redesign examples to the UI developer.

1.4.5 Interactive Pattern Recognition

Mining implicit data from user interactions provides research with a series of interesting opportunities in order to create technology that adapts to the dynamic environment of interactive systems. This chapter presents an iterative process to produce a user-desired result, in which the system initially proposes an automatic output, which is partially corrected by the user, which the system then uses to suggest a suitable hypothesis. Such iterative (and interactive and predictive) paradigm is the core of the MIPRCV project, a Spanish consortium of 10 universities and 7 research groups, which the author has been involved with since 2009. The main contribution of the author to the project has been the development (and later evaluation with real users) of interactive systems that implement the aforementioned paradigm, namely: *1*) Interactive Handwritten Transcription, *2*) Interactive Machine Translation, *3*) Interactive Grammatical Parsing, and *4*) Interactive Image Retrieval. According to user-simulated experiments and a series of real-world evaluations[4], results suggest that this paradigm can substantially reduce the human effort needed to produce a high-quality output.

Bibliography of Chapter 1

J. Bongard. Class notes on Human Computer Interaction. Available at http://cs.uvm.edu/~jbongard/2012_HCI/CS228_Class_02.pdf, 2010. Retrieved July 12, 2012.

S. K. Card, P. L. Pirolli, M. V. D. Wege, J. B. Morrison, R. W. Reeder, P. K. Schraedley, and J. Boshart. Information scent as a driver of web behavior graphs: Results of a protocol analysis method for web usability. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pp. 498–505, 2001.

J. M. Carroll. *Encyclopedia of Human-Computer Interaction*, chap. Human Computer Interaction (HCI). The Interaction Design Foundation, 2009.

K. Cherry. What is behaviorism? Available at http://psychology.about.com/od/behavioralpsychology/f/behaviorism.htm, 2006. Retrieved July 27, 2012.

A. Edmonds. Uzilla: A new tool for web usability testing. *Behavior Research Methods, Instruments, & Computers*, 35(2):194–201, 2003.

[4]Excepting Grammatical Parsing, all prototypes were empirically tested with real users.

M. Gellner, P. Forbrig, and M. Nelius. Results of mousemap-based usability evaluations – towards automating analyses of behavioral aspects. In *Proceedings of the 8th ERCIM Workshop UI4ALL: "User interfaces for all"*, 2004.

J. Goecks and J. Shavlik. Learning users' interests by unobtrusively observing their normal behavior. In *Proceedings of the 5th International Conference on Intelligent User Interfaces (IUI)*, pp. 129–132, 2000.

C. Grimes, D. Tang, and D. M. Russell. Query logs alone are not enough. In *Workshop on Query Log Analysis at the 18th International Conference on World Wide Web (WWW)*, 2007.

U. Hansmann. *Pervasive Computing: The Mobile World*. Springer, 2nd edition, 2003.

N. Henze. Analysis of user studies at MobileHCI 2011. Available at http://nhenze.net/?p=865, 2011. Retrieved October 2, 2011.

T. Hernandez. But what does it all mean? understanding eye-tracking results. Available at http://eyetools.com/articles, 2007. Retrieved November 8, 2009.

H. M. Hersh. Electronic mail usage analysis. In *Proceedings of the 1982 conference on Human Factors in Computing Systems (CHI)*, pp. 278–280, 1982.

P. I. Hofgesang. Methodology for preprocessing and evaluating the time spent on web pages. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pp. 218–225, 2006.

K. Hofmann, C. Reed, and H. Holz. Unobtrusive data collection for web-based social navigation. In *Proceedings of the 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, 2006.

F. Hwang, S. Keates, P. Langdon, and J. Clarkson. Mouse movements of motion-impaired users: A submovement analysis. In *Proceedings of the 6th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS)*, pp. 102–109, 2004.

W. Ju and L. Leifer. The design of implicit interactions: Making interactive systems less obnoxious. *Design Issues*, 24(3):72–84, 2008.

J. H. Kim, D. V. Gunn, E. Schuh, B. C. Phillips, R. J. Pagulayan, and D. Wixon. Tracking real-time user experience (TRUE): A comprehensive instrumentation solution for complex systems. In *Proceedings of the 26th annual SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pp. 443–452, 2008.

D. Kitayama, T. Teratani, and K. Sumiya. Digital map restructuring method based on implicit intentions extracted from users' operations. In *Proceedings of the 2nd International Conference on Ubiquitous Information Management and Communication (ICUIMC)*, pp. 45–53, 2008.

A. Oulasvirta and A. Salovaara. A cognitive meta-analysis of design approaches to interruptions in intelligent environments. In *Proceedings of Extended Abstracts on Human factors in computing systems (CHI EA)*, pp. 1155–1158, 2004.

A. Schmidt. Implicit human-computer interaction through context. *Personal and Ubiquitous Computing*, 4(2):191–199, 2000.

K. Sinclair. Creating interactions in building automation. Available at http://www.automatedbuildings.com/news/aug11/columns/110725014808emc.html, 2011. Retrieved October 2, 2011.

A.-H. Tan and C. Teo. Learning user profiles for personalized information dissemination. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pp. 183–188, 1998.

D. Tennenhouse. Proactive computing. *Communications of the ACM*, 43(5):43–50, 2000.

R. Vertegaal. Attentive user interfaces. *Communications of the ACM*, 46(3):31–33, 2003. Editorial note.

M. Weiser. Some computer science issues in ubiquitous computing. *Communications of the ACM*, 36(7):74–84, 1993.

M. Weiser. The computer for the 21st century. *Mobile Computing and Communications Review*, 3(3):3–11, 1999.

M. Weiser and J. S. Brown. The coming age of calm technology, 1996.

A. Wilson and N. Oliver. Multimodal sensing for explicit and implicit interaction. In *Proceedings of the 11th International Conference on Human-Computer Interaction (HCII)*, 2005.

E. Yoneki. Sentient future competition: Ambient intelligence by collaborative eye tracking. In *Proceedings of the European Workshop on Wireless Sensor Networks (EWSN)*, 2006.

P. Zigoris and Y. Zhang. Bayesian adaptive user profiling with explicit & implicit feedback. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 397–404, 2006.

Chapter 2

Interactive Usability Evaluation

Besides conventional features such as performance and robustness, usability is now recognized as an important quality attribute in software development. Traditionally, usability is investigated in controlled laboratory conditions, by recruiting a (hopefully representative) user sample and often performing video recordings and surveys that are later reviewed. This requires an important investment in time and money, not to mention that processing user interaction data is, at a minimum, cumbersome.

This chapter discusses the role of implicit interactions when performing usability tests on websites; concretely, *a)* which kind of data can be gathered by observing the overt behavior of users, without relying on explicit feedback, *b)* how this data can be presented to the usability evaluator, and *c)* which questions can be answered by inspecting such data.

Chapter Outline

2.1 Introduction

Determining how UIs are operated has aroused historically a lot of interest in many research fields such as product design and software engineering. For instance, detecting areas of interest or misused layout spaces, time to complete a task, etc. In a typical usability evaluation study, it is important for practitioners to record what was observed, in addition to why such behavior occurred, and modify the application according to the results, if needed. Observing the overt behavior of users provides useful information to investigate usability problems. Based on live observations, or analyses of video tapes, an evaluator constructs a problem list from the difficulties the users have accomplishing the tasks [Jacobsen et al., 1998]. However, 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. Such an approach is known to be costly (e.g., equipment, personnel, etc.) and rapid prototyping sometimes requires just preliminary studies. What is more, software applications usually have a life cycle extending well beyond the first release. Problems like these have led to consider alternate approaches. Concretely, in the field of web applications, remote activity tracking systems are today one of the main sources to evaluate the UI and analyze user behavior.

Processing user interaction data is thus, at a minimum, cumbersome. Fortunately, today there is a vast array of tools that can facilitate this task to the researcher. For instance, state-of-the-art usability systems employ client-side logging software, which include mouse and keyboard tracking, since these input devices are ubiquitous; therefore neither specific hardware nor special settings are required to collect interaction data remotely. The rationale that justifies these remote logging methods lies on the fact that there is a strong correlation to how likely a user will look at web pages [Chen et al., 2001; Huang et al., 2012; Mueller and Lockerd, 2001], and hence a mouse can tell us the user's intent and interests most of the time.

Modern cursor tracking systems usually support replaying the user interactions in the form of mouse tracks, a video-like visualization scheme, to allow researchers to easily inspect what is going on behind such interactions; e.g., *How many of the users did actually click on the "Buy" button? In which order did the user fill in the form fields? Do users ever scroll the web page? If so, how far exactly?* Nonetheless, traditional online video inspection has not benefited from the full capabilities of hypermedia and interactive techniques. We believe that mixing both channels is likely to better assist the usability practitioner. Therefore, our proposal is enhancing *hypervideo* technology to build a useful inspection tool for web tracking. Section 2.3 describes extensively the proposed system.

2.1.1 Lowering Usability Costs

Assessing the allocation of visual attention with conventional methods like click analysis, questionnaires, or simply by asking subjects where they have paid attention to, are limited to those processes which are part of conscious reflection and alive control. Relying exclusively on such methods will lead to a major validity problem, because attentional processes do not solely depend on user awareness. They are often driven beyond such awareness, and therefore are not reportable [Schiessl et al., 2003].

The eye movement is available as an indication of the user's goal before she could actuate any other input device [Jacob and Karn, 2003]. Unfortunately, an eye tracker is a very expensive hardware that requires exceptional calibration and needs to be operated in a laboratory with a small user sample, being not accessible to everyone [Nielsen, 2004]. Also, it has been shown that observers do not necessarily attend to what they are looking at and they do not necessarily look at what they are attending to [Toet, 2006]. On the contrary, measuring cursor activity is cheaper and quite affordable, since it does not require additional hardware, and enables remote data collecting. Moreover, in modern UIs, pointing devices such as pens, mice, trackpoints and touchpads, are ubiquitous [Ruiz et al., 2008]. Where there is a web browser, there is a mouse cursor [Chen et al., 2001].

Cursor tracking offers a series of interesting advantages when compared to traditional 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, offering insight information as new content is deployed. *4)* And most importantly, it is transparent to the users, so no experimenter bias or novelty effects are introduced, allowing users to navigate as they would normally do. One can argue that mouse movements are noisy, but also eye movements—actually even when looking at a point. Furthermore, the eye has higher error rate than the mouse, i.e., the coordinates reported by an eye tracker are often less accurate than those reported by most manual input devices. Finally, an eye tacker is an always-on device (which leads to the *Midas Touch* problem[1]), so distinguishing between intentional selection and simple inspection is more challenging with eye-gaze based devices.

2.2 Related Work

Automatic recording of user behavior within a system (also known as instrumentation) to develop and test theories has a rich history in psychology and UI design. One methodology that has recently begun to show promise within the HCI field is automated tracking or event logging to better understand user

[1] Eyes are never "off", so every gaze has the potential to activate an unintended command.

behavior. While users are interacting with an application, the system logs all UI events in the background. This event logging strategy enables the usability practitioner to automatically record specific behaviors and compute traditional usability metrics of interest (e.g., time to completion, UI errors, and so on) more accurately. Without these tools, these measurements would require researchers to meticulously hand-code behaviors of interest [Kim et al., 2008].

Mueller and Lockerd [2001] set a precedent in client-side tracking, presenting preliminary research on mouse behavior trends and user modeling. Arroyo et al. [2006] introduced the concept of collaborative filtering (that is, working with aggregated users' data), and the idea of using a web-based proxy to track external websites. Finally, Atterer et al. [2006] developed an advanced HTTP proxy that tracked the user's every move, being able to map mouse coordinates to DOM elements. Beyond the usefulness of these systems, only Atterer et al. [2006] could track complex Ajax websites, and visualization was solely the primary focus of Arroyo et al. [2006], although it was limited to an image overlaid on top the HTML pages. We argue that incorporating time-related information may enhance human interaction understanding, to replay exactly how users interact on a website. For instance, hesitations on a text paragraph may indicate interest about that content; or moving the mouse straight to a link of interest would show familiarity with the page. To this end, this is where video capabilities come into play, which, to some extent, have been lately implemented in industry systems.

Amongst the popular commercial systems at present, ClickTale[2], UserFly[3], and LuckyOrange[4] are deeply oriented to web analytics, with limited support for (non-interactive) visualizations. On the other hand, Mpathy[5] and Clixpy[6] are more visualization centered, but they use Flash sockets to transmit data, and so they only would work for users having the Flash plugin installed. Therefore, depending on the target audience of the website, it could lead to missing a huge fraction of the visitors that could provide valuable insights about their browsing experience. Finally, other approaches for visualizing user's activity are DOM based (Tag tracker[7]), or heatmap based (CrazyEgg[8]).

Basically, commercial systems work as "hosted solutions", i.e., a software-as-a-service delivery model. These systems require the webmaster to insert a tracking script in the pages to be targeted. Then such a tracking script transmits 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.

[2] http://clicktale.com
[3] http://userfly.com
[4] http://luckyorange.com
[5] http://m-pathy.com
[6] http://clixpy.com
[7] http://otterplus.com/mps
[8] http://crazyegg.com

2.3 Simple Mouse Tracking

Having looked at the literature, there are still some niches that are not fully covered by current tools. Mainly, there is no possibility to visualize the behavior of simultaneous users at the same time, and no system does report metrics related to user-centered data. These facts motivated the development of a new tool which, besides incorporating most of the state-of-the-art features, differs significantly from previous work, as stated in the next section. Now we shall describe SMT2 [Leiva and Vivó, 2012], our previous work, and how it differs from current systems. Then, we introduce a new version, SMT2ϵ, and show how it differs specifically from SMT2. Our tool is released as open source software, and can be downloaded and inspected at `http://smt2.googlecode.com`.

2.3.1 Overview of smt2

First of all, an important feature of our previous work regarding to state-of-the-art web tracking systems is the ability of compositing multiple interaction logs into a single hypervideo. This feature has been proved to be useful in assessing qualitatively the usability of websites, and also to discover common usage patterns by simply inspecting the visualizations (see Section 2.4).

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

Thirdly, the recording approach used in SMT2 is different regarding the ones described in current industry systems. Concretely, we perform a discretization in time of user interactions, following a simple event logging strategy together with the *polling* technique; i.e., taking a snapshot of the cursor status (mainly coordinates, clicks, and interacted elements) at a regular interval rate. This way, SMT2 tracks the user actions as they were exactly performed, allowing also to modify the speed at which movies can be replayed.

2.3.2 Introducing smt2ϵ

Regarding tracking capabilities, SMT2ϵ behaves almost identically as its predecessor, with the notable exception that SMT2ϵ features LZW compression to transmit the logged data, saving thus bandwidth. The actual improvements made to SMT2 that eventually derived in SMT2ϵ are focused on the server side.

To begin, our current effort goes toward interactive hypervideo synthesis from user browsing behavior. However, unlike conventional hypervideo, SMT2ϵ is

aimed to build 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**, i.e., a series of information layers that are rendered at runtime and provide the viewer with additional information. For instance, hovering over a click mark displays a tooltip showing the cursor coordinates, or hovering over a hesitation mark displays the amount of time the cursor was motionless.

SMT2ϵ extends this hypervideo technology with: 1. **hyperfragments**: videos can be linked to specific start/end parts, and 2. **hypernotes**: HTML-based annotations that point to specific video parts. These novel improvements are convenient in a tracking visualization scenario for a series of reasons. First, hyperfragments allow the viewer to select a portion of the video that may be of particular interest. Hyperfragments can be specified either with a starting or an ending timecode. This lets viewers quickly access desired information without having to watch the entire replay. Second, hypernotes allow the viewer to comment on the video at a specific point in time; e.g., to point out some video details or to let co-workers know that such video has been reviewed. When a hypernote is created, the viewer can click later on a note icon on the timeline that will seek the replay to the time indicated by the hypernote (Figure 2.3a). This provides viewers with indexing capabilities that can be extended to content searching. Fourth, the content of hypernotes is HTML, which enables rich-formatted text and insertion of links and images. This capability opens a new door to how visualizations can be later processed; e.g., it would be feasible to build narratives that summarize a user session.

In addition, SMT2ϵ features two installation modes: as an all-in-one solution (when website and admin site are both placed in the same server) and as a hosted service (website and admin site are both placed in different servers). SMT2 was limited in this regard, since to allow cross-domain communication, every website would require at least PHP support to forward the requests to the storage server (i.e., the admin site). With SMT2ϵ, 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 can use it.

Finally, SMT2ϵ features page classification according to user behavior in real time, by automatically mining the generated user and page models. The inclusion of this functionality was motivated by the fact that the viewer may find it useful to discover common interaction profiles as well as to easily identify outliers [Leiva, 2011] as new users access the website.

2.3.3 Architecture

As described below, SMT2ϵ is composed of three fundamental parts: recording, management, and visualization. On the server side, any web server (e.g., Apache, LightHTTPd, or IIS) supporting PHP and MySQL is able to run both

the admin site and the visualization application. The technology used to create such an interactive movies is a mixture of PHP (to query the database), HTML (to overlay the tracking data on top of it), JavaScript (to prepare the aforementioned tracking data), and ActionScript (to build the hypervideos).

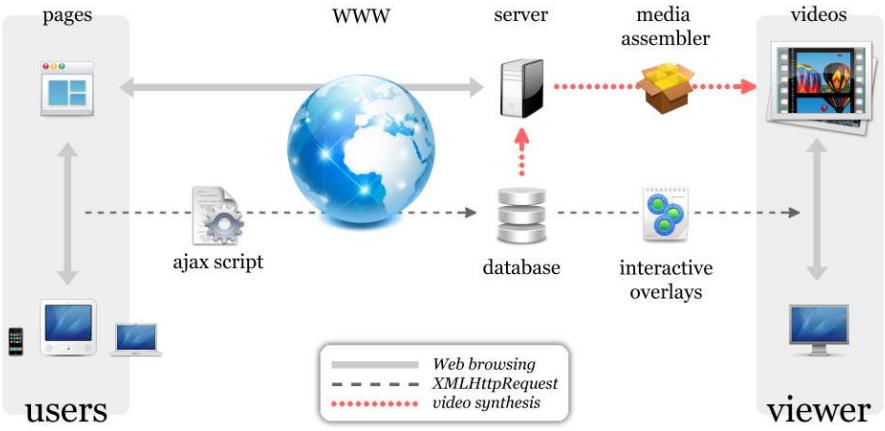


Figure 2.1: System architecture for acquiring users' activity and synthesizing interactive hypervideos.

2.3.4 Logging Users' Interactions

Every lower-level action can be recognized automatically, since the tracking script relies on the DOM event propagation model. We use the UNIPEN format [Guyon et al., 1994]—a popular scheme for handwriting data exchange and recognizer benchmarks—to store the mouse coordinates. This way, it is possible to re-compose the user activity in a reasonable fashion and to extract useful interaction semantics. While the user is browsing pages as she would normally do, an Ajax script logs the interaction data in the background. Tracking is performed in a transparent way for the users, either silently or by asking their consent.

It is worth pointing out that our strategy for transmitting the logged data do not rely on performing a server request each time a browser event is detected, as most tracking systems do. Instead, we store the data in a buffer, and we flush it at time-regular intervals. Doing so allows to reduce dramatically the number of HTTP requests to the web server, and hence lowering the overhead. Moreover, tracking can be *continuous* (default behavior) or *intermittent* (i.e., tracking stops/resumes on blur/focus events), letting the webmaster 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 use continuous recording, in order to keep mouse and eye coordinate streams synchronized.

Figure 2.2: A working example of inserted tracking code. Here we set the registration frequency to 24 fps and establish a maximum recording timeout of 1 hour. We also set random sampling for user selection, and ask consent to the chosen users for monitoring their browsing activity (they must agree to start recording).

```
<script type="text/javascript">
smt2.record({
  fps:       24,
  recTime:   3600,
  disabled:  Math.round(Math.random()),
  warn:      true
});
</script>
```

On the contrary, if the system is used on its own then the webmaster may want to save storage space in the database by enabling intermittent recording.

Another interesting logging feature is that the system can be invoked manually, if one have administrative rights to modify files in the web server, but it also can fetch external websites by using a PHP proxy that automatically inserts the required tracking code (Figure 2.2). We also take into account the user agent string to cache an exact copy of the page as it was originally requested, to avoid rendering differences due to different CSS being applied (e.g., on mobile devices compared to desktop computers). Additionally, it is possible to store interaction data from different domains in a single database, provided that each domain and the database are under the webmaster control.

2.3.5 Video Synthesis

The process to create an interactive hypervideo is composed of four main tasks: *1)* mining, *2)* encoding, *3)* rendering, and *4)* event dispatching. First, we query the database with the information that the viewer provides. Creating this kind of movies by using web technologies allows adding interactive information to on-screen visualizations, ranging from basic to more advanced playbacks. For example, she might request to visualize a single browsing session. The system will then retrieve the subsequent logs to make a video that will replay all tracks sequentially. On the contrary, though, the viewer might want to filter logs by operating system and page URL, in which case she uses a data mining form. In this case, data are retrieved according to the indicated filtering options, and logs will be merged into a single hypervideo when replaying (Figure 2.3). Different mouse trajectories will be normalized according to the original viewport of the user's browser and the current viewport of the viewer's browser. The normalization consists of a non-uniform affine mapping (either by scaling or translating the coordinates, depending on the type of layout: namely *fixed*, *centered*, or *liquid*). Then, a cached copy of the browsed page and the above-mentioned interaction data are bundled in a hypermedia player. This way, movies can be replayed within any web browser.

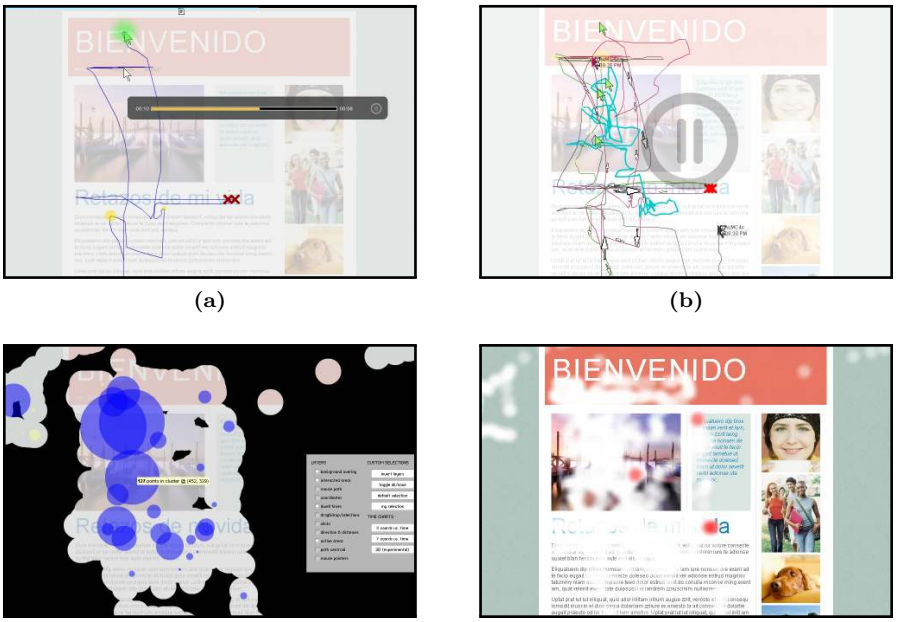


Figure 2.3: Some examples of our hypervideo visualization tool. [2.3a] single session with embedded media player. [2.3b] Replaying users' trails simultaneously, highlighting the average mouse track, and overlaying direction arrows. [2.3c] clusters of mouse movements, displaying also masked areas of activity. [2.3d] Dynamic heatmaps of mouse coordinates and clicks.

2.3.6 Interacting with the Data

On the server side, a multi-user admin site manages and delivers the hypervideos, allowing the viewer to customize a series of visualization options (Figure 2.3). The viewer can toggle different information layers interactively while she visualizes the videos by means of a control panel (Figure 2.4).

Automatic analysis of interaction features is also feasible for mining patterns within the admin site, since collected data are readily available in the database. This way, besides explicit metadata that is assigned to content, implicit knowledge can help to get a better picture on the nature of such content (see Section 2.5). Concretely, the metrics that SMT2ϵ computes for a given web page are described as follows.

Time Browsing time (in seconds) spent on the page.

Clicks Number of issued mouse clicks.

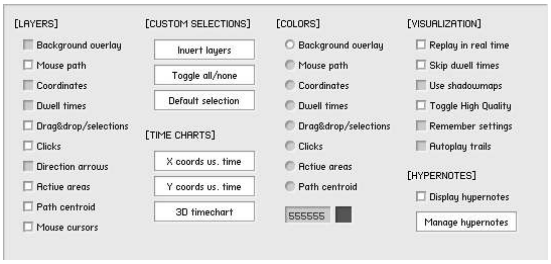


Figure 2.4: A draggable control panel is the main link between the viewer and the synthesized hypervideos. One can manipulate different visualization possibilities, which will be applied at runtime.

Activity Fraction of browsing time in which the cursor was moving, defined in $[0, 1]$. (0: no movements at all, 1: otherwise).

Length Cumulated sum (in px) of cursor distances.

Distance Average euclidean distance (in px) between coordinates.

Entry/exit points The first and last mouse coordinates, respectively.

Centroid Geometric center of all coordinates.

Amplitude Difference (in px) between maximum and minimum coordinates.

Scroll reach Percentage that informs how far did the user scrolled the page, defined in $[0, 1]$. (0: no scroll at all, 1: scroll reached the bottom of the page).

2.4 Applications

The following is a succinct list for illustrating the pragmatic utility of our system. We hope that the reader will be able to find other questions answered by examining other visualization marks.

- **Where do users hesitate? How much?** We followed the notion of *dwell time* introduced by Müller-Tomfelde [2007], i.e., the time span that people remain nearly motionless during pointing at objects. Dwell times are usually associated with ambiguous states of mind [Arroyo et al., 2006], possibly due to a thinking or cognitive learning process. In SMT2ϵ dwell times are displayed as circles with a radius proportional to the time in which the mouse does not move (Figure 2.5a). The system takes care of extremely large values of dwell times, by limiting the circle radii to a quarter of the viewport size.

- **Do users perform drag&drop operations? How?** Users perform drag and drop to select HTML content, or also to download an image to their desktop or to a file manager window. At a higher level, a web application

(a) **(b)** **(c)**

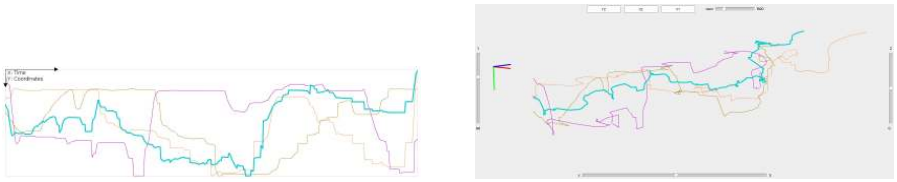
Figure 2.5: Combining visualization possibilities. [2.5a] Displaying hesitations (circles) and clicks (small crosses). [2.5b] Displaying entry/Exit coordinates (cursor bitmaps), motion centroids (big crosses), drag&drop activity (shaded fog), and interacted DOM elements. [2.5c] Analyzing a decision process; the user rearranged items in a list. Small circles represent 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 the red palette.

can support rearranging widgets to customize their layout, or also by adding objects to a list to be processed. Since we are using the UNIPEN format to encode each pair of mouse coordinates, the status of the click button can be easily represented, so SMT2ϵ provides a specific visualization type for these cases (e.g., Figure 2.5b).

- **Which elements is the user actually interacting with?** Thanks to the bubbling phase of JavaScript events, whenever a mouse event is dispatched (e.g., `mousemove`, `mouseover`) the tracking script traverses the DOM hierarchy to find if there is an element that relates to the event. Each tracking log holds a list of interacted DOM elements, sorted by time frequency (Figure 2.5c), so such list can be inspected either quantitatively (by looking at the numbers) or qualitatively (by looking at the colors). This visualization can be helpful to answer related questions, such as if the users go straight to the content or whether the mouse hovered over a link without clicking.

- **Which areas of the page do concentrate most of the interaction?** To answer this question, a K-means clustering of the coordinates is performed each time a mouse track ends replaying. So, focusing on the clustered areas allows 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 (Figure 2.3c). This visualization layer is notably appropriate when tracking data are rendered as a static image.

- **Do different mouse tracks correlate?** The viewer can select the 'time charts' option from the control panel (Figure 2.4) and compare multiple tracks simultaneously (see Figure 2.6). The coordinates are normalized in



(a) Normalized coordinates against time **(b)** Interactive 3D visualization

Figure 2.6: Time charts visualization. Bold line is the averaged mouse track, taking into account the selected users. The 3D view allows rotating the axes with 3 sliders (one for each direction), zooming, and projecting the lines in the YZ, XZ, and XY planes.

width and height according to the available chart size, to avoid possible visual biases.

- **What is the persistence of the page?** In this case, a 3D visualization might be useful (Figure 2.6b). The 3D chart renders the evolution of each pair of cursor coordinates x, y along the z axis, and provides simple interactive controls to ease further inspection. This way, for a given page, the viewer can observe at a glance the duration of each visit and how do they relate to the rest of them.

2.5 A Case Study

Here we provide empirical evidence for the efficacy of SMT2ϵ as a usability inspection tool. To test the system in a real-world scenario, the system was presented to a team of five graphic designers that were not usability experts. They wanted to redesign a corporative website, and they all used the tool for one month. One of them assumed the super administrator role, and the rest of them were assigned to the admin group. Thus, everyone could access to all admin sections without several restrictions; e.g., the difference between a user in the admin group and the super administrator is that admin users neither can download nor delete tracking logs, create user roles, or dump the database from the admin site.

2.5.1 Qualitative Results

Designers ran an informal usability test on their own. They configured SMT2ϵ as indicated in Figure 2.2, and gathered a representative user sample (near 5000 logs) in two weeks. Potential problems could be identified when visually inspecting the hypervideos, either for single users or by aggregating the logs from commonly browsed pages. Designers noticed that some areas of the home layout were causing confusion to most users; e.g., people hesitated over the main menu until deciding to click a navigational item. Designers could

also view that much of the interaction with the site was concentrated around the header section. Consequently, the team introduced some modifications to the web interface and gathered near 1000 logs in five days. This way, they could compare the generated interactions to previous data. Such updates had notable repercussions specially for first-time visitors (faster trajectories, less clicks overall). Figure 2.7 shows the appearance of the website before and after manually introducing the design updates. The reader can find more details of this study in [Leiva and Vivó, 2008].

(a) **(b)**

Figure 2.7: Website as it was designed initially (2.7a) and the redesigned layout (2.7b).

Overall, designers found the system very helpful. The main advantages suggested were being able to reproduce exactly what users did in a web page, and the speed with which a redesign could be verified. Concretely, the visualization layers (Figure 2.4) that the team found most useful were: mouse path, dwell times, clicks, direction & distances, and active areas. Designers also reported that there were two layers they found not relevant: path centroid and drag&drop/selections, mainly because *1)* the centroid was perceived as an imprecise indicator of the user interaction (i.e., designers stated that it was hard to derive meaningful conclusions by looking just at a single point on the screen) and *2)* only a few users performed drag&drop operations in the website. Designers liked the option of being able to switch to a static representation, specially when working with a large number of aggregated tracking logs.

2.5.2 Quantitative Results

Additionally, we asked permission to the team to download their gathered tracking logs for an offline study. They provided us with 4803 XML files. We processed them to build regression models of user activity and to create interaction profiles. We were able to predict with 71% of accuracy the expected time on a page based on the amount of mouse motion. Among other interesting findings, we noticed that the temporal evolution of mouse movements

follows a log-linear curve. This showed up that, instead of the idiosyncratic distinction between active (exploratory) and passive (lurker) users, there exists a wide continuum of in-between behaviors. More details of the above mentioned experiment can be found in [Leiva and Vivó, 2008].

Additionally, we used cursor data for a behavioral clustering experiment, which is detailed in Chapter 3. Eventually, 95% over all browsed pages could be explained by looking at 3 meaningful profiles. Designers could then review the pages belonging to each profile, focusing on the identified behaviors, and could continue iterating over the design-develop-test process. Similar experiments on this behavioral clustering methodology can be found in [Buscher et al., 2012; Leiva, 2011].

2.5.3 Limitations

Web-based activity tracking systems have inherent limitations, and of course SMT2ϵ is no exception to this rule. Although measuring page-level interactions is cheaper and enables remote data collecting at large, 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], Huang et al. [2011, 2012] have demonstrated that browsing time and user behavior have notable repercussions on gaze and mouse cursor alignment. Also, it has been shown that users do not necessarily attend to what they are looking at, and they do not necessarily look at what they are attending 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 their study.

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, 10000 concurrent users, then we believe the video visualization would not be much meaningful. Suffice to say it could be done, but at the cost of increasing the cognitive overload for the viewer (since visually inspecting too many users at the same time can be stressful), and only limited by the processing power of his 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 (for instance, by looking at the 'directions & distances' layer, Figure 2.4).

Additionally, besides the fact that our tool normalizes the mouse coordinates to avoid possible visual biases while replaying the hypervideos, we noticed that sometimes the visualization is not perfectly accurate, partly due to JavaScript rounding errors, partly due to discrepancies between how browsers render CSS. These browser discrepancies can be greatly minimized by using a reset

stylesheet on the web page. On the contrary, higher discrepancies are expected when the user access 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 trivial solution. For instance, the system could use the mobile user agent to fetch the page that the user visited, but it could happen that the page had changed since that visit, or even that it no longer exists. The same argument applies to the stylesheets of that page. Therefore, a more technically advanced approach should be taken into consideration, such as caching all assets for each user visits, at the cost of increasing the storage space.

2.6 Conclusions and Future Work

To better understand user behavior on the Web, modern tracking systems should rely on the browsing capabilities of the users, instead of the traditional server access logs. However, this approach has a clear trade-off, as moving to the client side involves having to process much more data. We believe that offering such data processed as a hypervideo can be considered as a promising idea and a specially helpful approach for assessing the usability of websites.

This article has described the design and implementation of SMT2ϵ, a web-based system for automatically gathering, mining, selecting, and visualizing browsing data in an interactive hypermedia presentation, either as a video or as a static visualization. The tracking system collects fine-grained information about user behavior, and allows viewers to control what they watch, when, and *how*, by selecting diverse types of infographics.

We have reported the main differences between our tool and previous web tracking systems, including the state of the art and highlighting our contributions to the field. We have shown the value of enhancing video visualizations with interactive techniques to present the viewer with complex information quickly and clearly. We have also described a real-world usage scenario proving that our system is a feasible and realistic implementation.

Tracking page-level browsing activity with SMT2ϵ 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 both qualitative and quantitative studies, being able to complement existing methodologies on web browsing and human behavior. Therefore, we believe that SMT2ϵ is ready to extend its scope to a broader, interdisciplinary audience.

One of our priorities for future work is working on scalability and performance limits especially concerning high-recording speeds. We also plan to enrich the system with other types of behavior analysis, for instance working with eye-tracking data, as hinted in the previous section, since user interaction is inherently multimodal.

2.6.1 Some Notes on Privacy

Monitoring the user interactions at a fine-grained level can be very useful to help shaping a more usable website, or making it more appropriate to the behavior 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 SMT2ϵ system with that notion in mind.

First, we believe logging keystrokes could be employed for unfair purposes, depending on the uses that one could derive from this tool. For that reason, we rejected to log raw keystroke data and track only keyboard events instead, without registering the associated character codes. Second, we believe 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 to ask always the user before tracking takes place. Furthermore, once a user has agreed to track, we advocate for asking her consent again after a prudential amount of time (e.g., a few hours, until the end of the browsing session, or when a tracking campaign finalizes). Third, we believe logged data should be stored in a server the webmaster owns, and not in one she cannot control. At least, it should be possible to let users access their (raw) data. We encourage commercial tracking systems to do so, since chances are there and current web technologies can support it. 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 usage at the site where SMT2ϵ is included. This way, we try to avoid an illegitimate abuse of our system (e.g., without advising 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.

Bibliography of Chapter 2

E. Arroyo, T. Selker, and W. Wei. Usability tool for analysis of web designs using mouse tracks. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*, pp. 484–489, 2006.

R. Atterer, M. Wnuk, and A. Schmidt. 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)*, pp. 203–212, 2006.

G. Buscher, R. W. White, S. Dumais, and J. Huang. Large-scale analysis of individual and task differences in search result page examination strategies. In *Proceedings of the fifth ACM international conference on Web search and data mining (WSDM)*, pp. 373–382, 2012.

M.-C. Chen, J. R. Anderson, and M.-H. Sohn. 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)*, pp. 281–282, 2001.

G. Daniel and M. Chen. Video visualization. In *Proceedings of the 14th IEEE Visualization (VIS)*, pp. 409–416, 2003.

Q. Guo and E. Agichtein. Ready to buy or just browsing? detecting web searcher goals from interaction data. In *Proceedings of SIGIR*, pp. 130–137, 2010.

I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. UNIPEN project of on-line data exchange and recognizer benchmarks. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pp. 29–33, 1994.

J. Huang, R. W. White, and S. Dumais. 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)*, pp. 1225–1234, 2011.

J. Huang, R. W. White, and G. Buscher. User see, user point: Gaze and cursor alignment in web search. In *Proceedings of the annual Conference on Human Factors in Computing Systems (CHI)*, pp. 1341–1350, 2012.

R. J. Jacob and K. S. Karn. *Eye Tracking in Human-Computer Interaction and Usability Research: Ready to Deliver the Promises*, chap. Section Comentary, pp. 573–605. Elsevier Science, 2003.

N. E. Jacobsen, M. Hertzum, and B. E. John. The evaluator effect in usability tests. In *CHI 98 conference summary on Human factors in computing systems*, pp. 255–256, 1998.

J. H. Kim, D. V. Gunn, E. Schuh, B. C. Phillips, R. J. Pagulayan, and D. Wixon. Tracking real-time user experience (TRUE): A comprehensive instrumentation solution for complex systems. In *Proceedings of the 26th annual SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pp. 443–452, 2008.

L. A. Leiva. Mining the browsing context: Discovering interaction profiles via behavioral clustering. In *Adjunct Proceedings of the 19th conference on User Modeling, Adaptation, and Personalization (UMAP)*, pp. 31–33, 2011.

L. A. Leiva and R. Vivó. A gesture inference methodology for user evaluation based on mouse activity tracking. In *Proceedings of Interfaces and Human-Computer Interaction (IHCI)*, pp. 58–67, 2008.

L. A. Leiva and R. Vivó. Interactive hypervideo visualization for browsing behavior analysis. In *Proceedings of the 21st international conference companion on World Wide Web (WWW)*, pp. 381–384, 2012.

F. Mueller and A. Lockerd. 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)*, pp. 279–280, 2001.

C. Müller-Tomfelde. Dwell-based pointing in applications of human computer interaction. In *Proceedings of the IFIP Conference on Human-Computer Interaction (INTERACT)*, pp. 560–573, 2007.

J. Nielsen. Capturing thoughts, capturing minds? from think aloud to participatory analysis. Available at http://openarchive.cbs.dk/bitstream/handle/10398/6501/14-2004.pdf, 2004. Retrieved November 8, 2009.

J. Ruiz, D. Tausky, A. Bunt, E. Lank, and R. Mann. Analyzing the kinematics of bivariate pointing. In *Proceedings of Graphics Interface (GI)*, pp. 251–258, 2008.

M. Schiessl, S. Duda, A. Thölke, and R. Fischer. Eye tracking and its application in usability and media research. *MMI Interaktiv*, 6(1):41–50, 2003.

J. Smith and D. Stotts. An extensible object tracking architecture for hyperlinking in real-time and stored video streams. Tech. Report 02-017, Univ. North Caroline and Chapel Hill, 2002.

A. Toet. Gaze directed displays as an enabling technology for attention aware systems. *Computers in Human Behavior*, 22(4):615–647, 2006.

Chapter 3

Behavioral Clustering

Behavioral clustering is a broad term that refers to the task of automatically labeling and classifying user behavior. In a general context, clustering allows to identify sub-populations in a dataset, so that they can be represented by more compact structures for, e.g., classification and retrieval purposes. To this end, implicit interaction can provide current clustering methods with additional information. For instance, on the Web, clustering is usually deployed by using a single data source, which is often browsing usage information derived from server access logs. However, when it comes to getting deep information about user behavior, this representation is inadequate in such a dynamic environment.

In this chapter, two opportunities are identified to enhance behavioral clustering through implicit interaction research. First, fine-grained interactions can reveal valuable information that is not available in typical access logs; e.g., cursor movements, hesitations before clicking, etc. Second, user behavior has an intrinsic *sequential* nature, which is not considered on current clustering analysis, that can be exploited to simplify the structure of the data. Therefore, we propose two approaches for both opportunities: *1*) a novel methodology to model the website, i.e., finding interaction profiles according to how users behave while browsing, and *2*) a novel clustering algorithm to deal with sequentially-distributed data, whose suitability is illustrated in a human action recognition task.

Chapter Outline

3.1 Introduction

A pervasive problem in science is to construct meaningful classifications of observed phenomena. Clustering can be seen as a compression technique to simplify the structure of the data, so that original objects can be represented by more compact structures that are better tailored for classification, storage, and retrieval purposes. The motivation to using these simplified structures can be as elemental as reducing the number of data samples to save space in large databases, such as web access logs, to more complex applications, such as detecting actions in hours of sensor data. The importance and interdisciplinary nature of clustering is evident through its vast literature; c.f. [Jain, 2010; Jain et al., 1999].

Two broad categories of clustering can be distinguished. In the first one, we have data from known groups as well as observations from entities whose group membership is unknown initially and has to be determined through the analysis of the data. On the other hand, the groups are themselves unknown a priori and the primary purpose of data analysis is to determine the groupings from the data, so that entities within the same group are in some sense more similar than those that belong to different groups. The latter category is the one we are tackling in this chapter.

We explore two novel approaches to (unsupervised) behavioral clustering, with a special emphasis on web page classification and human action recognition. On the one hand, in the context of page classification, currently the task of clustering web pages is approached in a similar way for both web documents and plain text documents. Even if it is known that web pages contain richer and implicit information associated to them [Poblete and Baeza-Yates, 2008], like the interactions that users perform while browsing. Thus, when facing a finer-grained understanding of user behavior and document analysis, server analytics are anything but accurate, being necessary to move toward the client side. As pointed out later, the first core contribution of this chapter is focused on this task.

On the other hand, the task of detecting actions from user behavior is not an easy one. Actions (or activities) are sequential by definition, and, while there are many works that solve sequential *supervised* machine learning problems (e.g. [Dietterich, 2002]), the unsupervised case had remained posing new challenges in the research community for years (e.g. [Trahanias and Skordalakis, 1989]). The second core contribution of this chapter consists in solving this problem.

3.1.1 Background

Cluster analysis provides an unsupervised classification scheme to efficiently organize large datasets [Duda et al., 2001]. Additionally, cluster analysis can

supply a means for assessing dimensionality [Agrawal et al., 1998] or identifying outliers [Leiva, 2011]. The fundamental data clustering problem may be defined as discovering "natural" groups, or clubbing similar objects together.

In this chapter, data clustering is seen as a data partitioning problem [Dubes, 1993; MacQueen, 1967; Yu, 2005] as opposed to the hierarchical approach [Fraley, 1996; Murtagh, 1984; Ward, 1963], since we are interested in a *partition* of the data and not in a *structure* (dendrogram) thereof.

Partitional clustering divides a dataset $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ of n d-dimensional feature vectors into a set $\prod = \{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$ of k disjoint homogeneous classes with $1 < k \ll n$. It is worth pointing out that the task of finding the optimum partition is formidable even for a computer, since this is an NP-hard problem. For example, if $k = 3$, we need to look at 3^{n-1} combinations. One way to tackle this problem is to define a criterion function that measures the quality of the clustering partition and then find a partition $\prod^*$ that extremizes such a criterion function.

The most popular algorithm for partitional clustering in scientific and industrial applications is by far the K-means (or C-means) algorithm, which can be considered as a simplified case of Expectation-Maximization (EM) clustering, and is described in the next section.

3.2 Revisiting the K-means Algorithm

The K-means algorithm is known for its simplicity, relative robustness, and fast convergence to local minima. K-means, including its multiple variants such as Fuzzy C-Means [Dunn, 1973], K-Medoids [Kaufman and Rousseeuw, 1990], etc., is based on the firm foundation of variance analysis. It requires the number of clusters k to be an input parameter, which is tightly coupled to the nature of the involved task, though there are many studies for choosing k automatically [Bezdek and Pal, 1998; Davies and Bouldin, 1979; Dunn, 1974; Hamerly and Elkan, 2001; Hubert and Arabie, 1985; Milligigan and Cooper, 1985; Sugar, 1998; Tibshirani et al., 2001]. The rough but usual approach is to try clustering with several values of k and choose the one that contributes most to the minimization criterion. Nonetheless, a simple rule of thumb is setting the number of clusters to [Mardia et al., 1979]:

$$k \approx (n/2)^{1/2} \tag{3.1}$$

The criterion function that K-means tries to minimize is the Sum of Quadratic Errors (SQE), denoted simply as Energy or J in the literature, which emphasizes the local structure of the data [Veenman et al., 2002]:

$$J = \sum_{j=1}^{k} H_j \tag{3.2}$$

where

$$H_j = \sum_{\boldsymbol{x} \in \mathcal{C}_j} \| \boldsymbol{x} - \boldsymbol{\mu}_j \|^2 \tag{3.3}$$

represents the heterogeneity (or distortion) of cluster $\mathcal{C}_j$, and

$$\boldsymbol{\mu}_j = \frac{1}{n_j} \sum_{\boldsymbol{x} \in \mathcal{C}_j} \boldsymbol{x} \tag{3.4}$$

is the cluster mean, with $n_j = |\mathcal{C}_j|$ being the number of samples in such cluster.

The most common implementation of this algorithm, generally attributed to Lloyd [1982], uses a minimum distance criterion, where in each iteration all samples are assigned to their closest cluster mean and convergence is achieved when the assignments no longer change. There exists, however, a more interesting version, often attributed to Duda and Hart [1973], which uses a sample-by-sample iterative optimization refinement scheme. At each step, the SQE is evaluated and the considered sample is reallocated to a different cluster if and only if that reassignment decreases J. Clearly, such a greedy optimization guarantees that the resulting partition corresponds always to a local minimum of the SQE. This refined version is explained as follows.

The variation in the SQE produced when moving a sample $\boldsymbol{x}$ from cluster j to cluster l can be obtained in a single computational step as [Duda et al., 2001]:

$$\Delta J(\boldsymbol{x}, j, l) = \frac{n_l}{n_l + 1} \| \boldsymbol{x} - \boldsymbol{\mu}_l \|^2 - \frac{n_j}{n_j - 1} \| \boldsymbol{x} - \boldsymbol{\mu}_j \|^2 \tag{3.5}$$

If this increment is negative, the new means, $\boldsymbol{\mu}'_j$, $\boldsymbol{\mu}'_l$ and the SQE, J', can then be incrementally computed as follows [Duda et al., 2001]:

$$\begin{aligned} \boldsymbol{\mu}'_j &= \boldsymbol{\mu}_j - \frac{\boldsymbol{x} - \boldsymbol{\mu}_j}{n_j - 1} \\ \boldsymbol{\mu}'_l &= \boldsymbol{\mu}_l + \frac{\boldsymbol{x} - \boldsymbol{\mu}_l}{n_l + 1} \\ J' &= J + \Delta J(\boldsymbol{x}, j, l) \end{aligned} \tag{3.6}$$

3.2.1 Sequential Clustering

When clustering sequential data there exists a strong constraint, often related to time, that can be exploited to a great advantage. Nonetheless, by ignoring this constraint, classical clustering techniques fail to cope with the underlying

sequential data structure, as illustrated in Figure 3.1. What is more, previous research on sequential data clustering considered the objects to cluster as whole data sequences or previously determined subsequences thereof; c.f., Guralnik and Karypis [2001]; Lee et al. [2007]. Instead, we are interested in discovering *subtrajectories* within a single trajectory, so that we can obtain a simplified data structure preserving the underlying data sequentiality. From this point of view, approaches based on Hidden Markov Models (HMMs) have been proposed; e.g., [Bashir et al., 2007]. The downside of HMMs, however, is that they require complex training, and also can be prohibitive if processing power is a restriction, e.g., working on mobile devices. As such, we propose here a closed-form solution having a low computational cost in terms of performance, which translates to really fast convergence times, and provides consistent results in terms of accuracy: each run for a given number of classes always yields the same (well-formed) sequential clustering configuration.

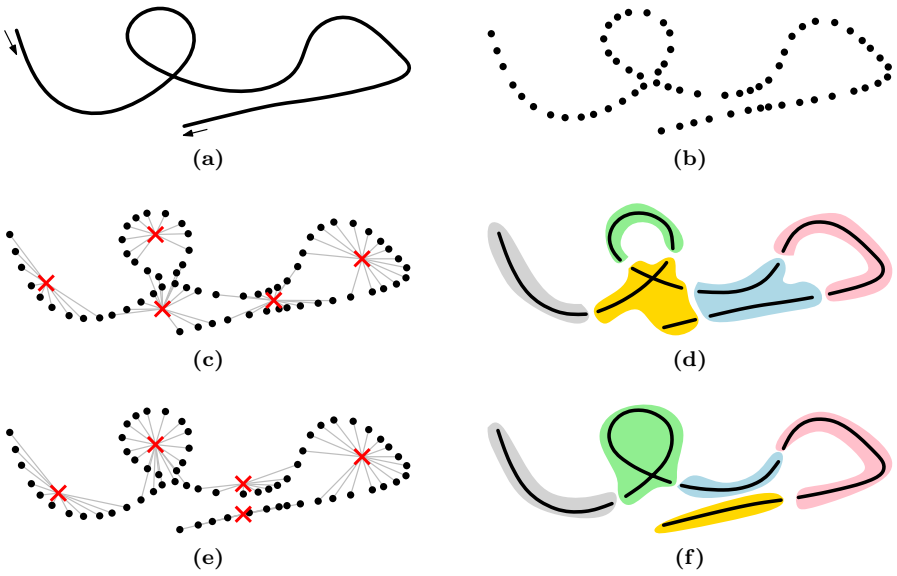


Figure 3.1: A 2D example. An arbitrary shape (3.1a) is digitized (3.1b) and reduced to 5 elemental units. Classical clustering algorithms do not deal with temporal information and, therefore, resulting units are ill-defined (3.1c), leading to an inconsistent configuration (3.1d). Our approach, however, provides a simple framework to easily cope with the sequentiality of the data (3.1e, 3.1f).

If the data in a dataset X are sequentially given, it can be said that such data describe a *trace* or *trajectory* in the d-dimensional vector space where samples are represented:

$$X = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \tag{3.7}$$

We define a sequential clustering into k classes as the mapping

$$\mathrm{b} : \{1, \ldots, k\} \mapsto \{1, \ldots, n\}$$

where b_j is the (left) *boundary* of cluster j; i.e., the index of the first sample in that cluster. See Figure 3.2 for a graphical example.

Using this convenient notation, the j-th (sequential) cluster of X can be written as follows:

$$\mathcal{C}_j = \{\boldsymbol{x}_{\mathrm{b}_j}, \boldsymbol{x}_{\mathrm{b}_j+1}, \ldots, \boldsymbol{x}_{\mathrm{b}_j+n_j-1}\} \tag{3.8}$$

where n_j can now be trivially computed as

$$n_j = \mathrm{b}_{j+1} - \mathrm{b}_j \tag{3.9}$$

This way, (3.3) and (3.4) can be rewritten as

$$H_j = \sum_{i=\mathrm{b}_j}^{\mathrm{b}_{j+1}-1} \| \boldsymbol{x}_i - \boldsymbol{\mu}_j \|^2 \tag{3.10}$$

$$\boldsymbol{\mu}_j = \frac{1}{n_j} \sum_{i=\mathrm{b}_j}^{\mathrm{b}_{j+1}-1} \boldsymbol{x}_i \tag{3.11}$$

and (3.5) and (3.6) can be directly used as such with this new formulation.

3.2.2 Warped K-Means

We propose a novel algorithm named *Warped K-Means* (WKM), inspired by the idea that the original data structure is delusively distorted, or "unfolded" (see Figure 3.2) to cope with the sequentiality restrictions. Our proposal is based on the trace segmentation (TS) technique for partition initialization (Figure 3.3), followed by a K-means-like optimization procedure (Figure 3.4).

As in classical K-means, WKM reallocates samples based on the analysis of effects on the objective function J, caused by moving a sample from its current cluster to a potentially better one. But now a hard sequentiality constraint is imposed. The first half of samples in cluster j are only allowed to move to cluster $j-1$, and, respectively, the last half of samples are only allowed to move to cluster $j+1$. A sample will be reallocated if and only if the corresponding SQE increment is beneficial (i.e., negative). This process is iterated until no transfers are performed.

Because of this constraint, along with the sequential ordering of samples within each cluster, typically only the samples close to the cluster boundaries get reallocated. To take advantage of this observation, we introduce an optional parameter $\delta \in [0, 1]$ which allows us to fine-tune the WKM behavior and at the

same time achieve further reductions in computational cost. It allows testing only those samples that are more or less close to cluster boundaries. In the extreme case of $\delta = 0$ the algorithm is conservative: all samples in a cluster are visited to see if they should be reallocated. In the other extreme, if $\delta = 1$ WKM is optimistic: only the boundary and the last sample in each cluster will be checked. In general, the effect of δ is illustrated in Figure 3.2.

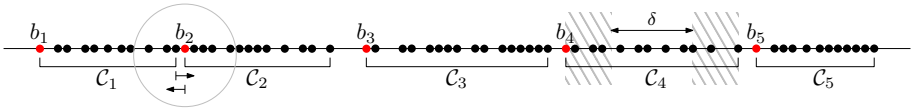


Figure 3.2: The basis of WKM. The algorithm provides an optional parameter δ which specifies the maximum amount of samples that will be inspected in each iteration. This way, if $\delta = 1$ only two samples are considered per iteration, while if $\delta = 0$ full search is carried out.

In sum, three key features differentiate our approach from other K-means based algorithms: initialization, visiting order, and sequentiality constraints.

Algorithm: TS Boundary Initialization

Input: Trajectory $X = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$; No. Clusters $k \geq 2$
Output: Boundaries $\mathrm{b}_1, \ldots, \mathrm{b}_k$

$L_1 = 0$
for $i = 2$ **to** n **do** // Accumulated trace length
  $L_i = L_{i-1} + \| \boldsymbol{x}_i - \boldsymbol{x}_{i-1} \|$
$\lambda = \frac{L_n}{k}$ // Segment length
$i = 1$
for $j = 1$ **to** k **do**
  while $\lambda\ (j-1) > L_i$ **do** // Interpolate
    $i{+}{+}$
  $\mathrm{b}_j = i$ // Define boundaries

Figure 3.3: Boundaries initialization. Each boundary is evenly allocated according to a piecewise linear interpolation on accumulated distances, resulting in a non-linearly distributed boundary allocation.

Algorithm Overview

In each cluster j, the samples close to its boundary b_j are first visited to see if they can be advantageously reallocated to the previous cluster, $j - 1$ ("reallocate backwards" loop). Then, the samples close to the boundary of the next cluster, $j + 1$, are similarly considered ("reallocate forwards" loop). It is worth noting that with $\delta < 1$ the proportion of samples processed in each backward or forward sequential chunk is typically less than the number corresponding to the given value of δ. This is because the reallocation process

Algorithm: WKM

Input: Trajectory X; No. Clusters $k \geq 2$ [; Proportion $\delta = 0.0$]
Output: Boundaries $\mathrm{b}_1, \ldots, \mathrm{b}_k$; Centroids $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$; Distortion J

```
Initialize boundaries b_1, ..., b_k                          // Use TS (Figure 3.3)
for j = 1 to k do
    Compute μ_j, n_j, J                          // Use Eq. (3.2), (3.9), (3.10), and (3.11)
repeat
    transfers = false
    for j = 1 to k do
        if j > 1 then                                        // Reallocate backwards 1st half
            first = b_j;   last = first + ⌊(n_j/2)(1 − δ)⌋
            for i = first up to last do
                if n_{j−1} > 1 and ΔJ(x_i, j, j − 1) < 0 then
                    transfers = true
                    b_j += 1;   n_j += 1;   n_{j−1} −= 1
                    Update μ_j, μ_{j−1}, J                   // According to Eq. (3.6)
                else break
        if j < k then                                        // Reallocate forwards 2nd half
            last = b_{j+1} − 1;   first = last − ⌊(n_j/2)(1 − δ)⌋
            for i = last down to first do
                if n_j > 1 and ΔJ(x_i, j, j + 1) < 0 then
                    transfers = true
                    b_{j+1} −= 1;   n_j −= 1;   n_{j+1} += 1
                    Update μ_j, μ_{j+1}, J                   // According to Eq. (3.6)
                else break
until ¬transfers
```

Figure 3.4: Warped K-Means. A sample $\boldsymbol{x} \in \mathcal{C}_j$ is only allowed to move either to cluster $\mathcal{C}_{j-1}$ or $\mathcal{C}_{j+1}$. If a move proves advantageous, that is, the increment in SQE is negative, the sample is reallocated and the two cluster means involved in such a reallocation are incrementally recomputed according to (3.6). Otherwise, the next cluster is inspected, in order to preserve the clustering sequentiality.

is aborted as soon as the SQE does not improve for that chunk, in order to preserve the sequentiality of our clustering procedure.

Note also that if some samples are reallocated during the forward processing of cluster j, then we do not need to re-check them in the backward processing of cluster $j + 1$. This is easily verifiable with an auxiliary variable that stores the index of the last reallocated sample. This detail, however, is not shown in the WKM pseudo-code for the sake of clarity.

The computational cost of a complete iteration of WKM over the whole sequence X, depends on the number of samples n, the sample vector dimension d, and the number of clusters k. As previously discussed, it can also depend on the value of δ. On the one hand, if $\delta = 1$, the complexity of WKM is reduced to $\Theta(kd)$ per iteration, in comparison to $\Theta(nkd)$ in the case of classical K-means.

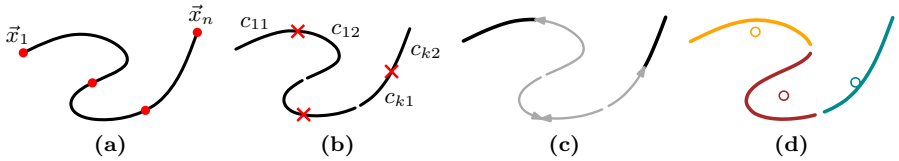


Figure 3.5: Graphical overview of Figure 3.4 for $k = 3$ clusters. [3.5a] Key points identification: the first and last key points always match the first and last data points ($\boldsymbol{x}_1$ and $\boldsymbol{x}_n$). [3.5b] Initial segmentation: crosses mark the k segments' middle points. [3.5c] Point visiting order for reallocation: notice that chunks c_{11} and c_{kn} do not need to be inspected. [3.5d] Final clustering configuration: circles represent each segment's centroid.

On the other hand, if $\delta = 0$, the best- and worst-case complexities are $\Omega(kd)$ and $O(nd)$, respectively. Therefore, for all values of δ and in all cases, each iteration of WKM is expected to be (much) faster than conventional K-means algorithms. Moreover, according to empirical observations, the convergence tends to require less iterations than such classical K-means algorithms.

Overall, the main advantages of our proposal can be summarized as follows:

- **Consistent results**: It always guarantees the convergence to a good local minimum, i.e., a low distorted partition of the original dataset that preserves sequence ordering.
- **Robust solution**: Each run for a given k always yields the same clustering configuration—thanks to the initialization algorithm and the minimization criterion for sample reallocation.
- **Low computational cost**: Much lower than that of classical K-means algorithms since, instead of the usual all-against-all search strategy, we only need to check two clusters in each step.
- **No extra mandatory parameters**: Our solution requires the same input data and parameters as in K-means, though an optional δ threshold can be specified to tune both the algorithm behavior and its cost.

As discussed by Leiva and Vidal [2011], the WKM algorithm is also suitable for online learning tasks over large datasets, due to the following facts: *1)* the computational cost of updating the centroids is independent of the number of samples and *2)* the final partition can be updated while new samples arrive without affecting too much the previous data structure.

3.3 Evaluation

In this section we evaluate behavioral clustering on two different tasks: web page classification and action recognition. In the former task, we are interested

in describing a website by how users interact within their contents. To this end, a classical clustering methodology is intuitively quite useful: different pages that trigger different behaviors should lie in different clusters, while pages with similar interactions are likely to be assigned to the same cluster. In the latter task, we are interested in characterizing human actions from raw sensor data. To this end, we look for data compression methods to reduce the number of samples for later action recognition. Here, a clustering methodology might also be quite useful, although preserving data sequentiality is of utmost importance—something that classical clustering methods fail to achieve.

Notice that the goal of the page classification task is to describe the website as a whole, so there is no need to preserve data sequentiality. However, the goal of the action recognition task is to discover the most informative number of elementary samples that define a human action. Therefore, in this case it is clear that a better outcome is expected if we employ our WKM algorithm instead of classical methods.

3.3.1 Clustering Browsing Interactions

In the same way as web clustering engines organize search results by topic or document relevance, our method aims to organize websites by users' interaction semantics. Such semantics of interaction are characterized by a series of metrics (16 in total), which are computed by our mouse tracking tool and were described in Section 2.3.6, say, 1D metrics: browsing time, number of clicks, motion activity, and path length; and 2D metrics (with X and Y components): distance, range, entry point, exit point, centroid, and scroll reach. We hypothesize that if such metrics are consistent, they should generate clusters of (approximately) same precision for a given typology of pages. In addition, it is important to remark that metrics should be normalized. For instance, time and scrolling are often reported as relevant metrics [Claypool et al., 2001; Holub and Bielikova, 2010]. However, it is clear that longer/bigger pages will require both more time and scrolling, and hence they could lead to misleading results if one does not consider data normalization. Usually whitening the data (i.e., ensuring a distribution of each metric with mean 0 and variance 1) may be enough.

Method

We gathered interaction data for approximately a month on three *informational* websites (Figure 3.6), i.e., they are dedicated to the purpose of providing information to the users (like, e.g., news portals or corporate blogs). Most websites could fit in this type of website to some extent, so evaluating our approach on this typology should ensure a broad generalization scope. The characteristics of each corpus are summarized in Table 3.1.

(a) OTH (habitat trends) **(b)** NM (rice company) **(c)** LAKQ (social club)

Figure 3.6: Example screenshots from the corresponding websites of each evaluated dataset (see also Table 3.1).

Codename	Size (MB)	# Logs	# URLs
OTH	25.5	4803	63
NM	33.5	5601	43
LAKQ	7.4	1232	28

Table 3.1: Overview of evaluated datasets (see also Figure 3.6).

Procedure

Users were selected by random sampling, which means that only a fraction of all visitors (with equal probability of selection) was collected. We set a tracking frequency of 24 fps. Each interaction log was stored in a MySQL database and then exported in XML format. Logs were modeled as normalized interaction-based 16-d feature vectors (see Section 3.3.1). We took into account visits that lasted 0.5 hours at most, in order to discard bogus or spurious logs beforehand. Then, we applied the classical K-means algorithm to automatically group the logs in each corpus, using random convex combination as initialization method [Leiva and Vidal, 2010] to accelerate convergence. The optimal number of clusters for each corpus was determined as the marginally less distorted grouping in terms of the SQE, which is proportional to the intra-cluster (or within-class) variance; see, e.g., Figure 3.7. Once we had each log assigned to a cluster, we extracted the mean and standard deviation for the tracked interaction features, for later comparison and further analysis.

Results

To illustrate the usefulness of the proposed framework we start by describing the profiles found in the OTH corpus. Table 3.2 summarizes the clustering results for this dataset. Then we discuss the main observations that relate to the other evaluated corpora.

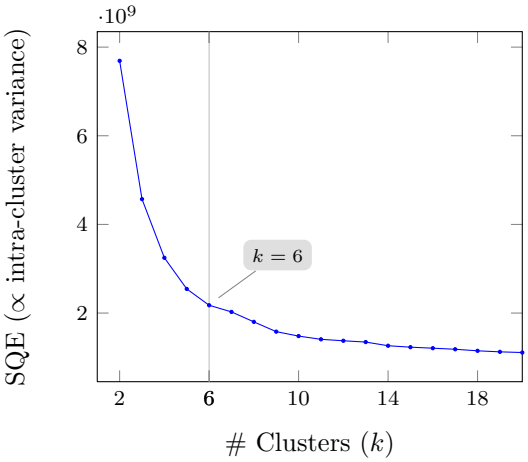


Figure 3.7: Clustering the OTH dataset. The intra-cluster variance (or energy, or SQE) decreases with increasing number of classes k. However, at some point the marginal gain will be smaller. Intuitively, this can be chosen as the number of clusters that better summarizes the dataset.

Cluster #	Population	%	Energy (SQE)	%	Variance
2	698	14	$3.6{\cdot}10^8$	16	$5.1{\cdot}10^5$
3	1347	28	$4.7{\cdot}10^8$	22	$3.5{\cdot}10^5$
6	2220	46	$5.4{\cdot}10^8$	25	$2.4{\cdot}10^5$
Avg. Total	4748	100	$2.1{\cdot}10^9$	100	$4.5{\cdot}10^5$

Table 3.2: Clusters found in the OTH dataset. Outliers were classified into three clusters (#1, #4, and #5), not reported here because they all represent near 10% of sample population.

Profiles in OTH corpus According to the 'elbow' criterion[1] (Figure 3.7), we found $k = 6$ to be the number of classes that better summarizes this dataset. However, three cluster were identified as outliers, which accounted for near 10% of the population. So actually we found three meaningful groups in this dataset. This fact reinforced the idea of using behavioral clustering for isolating sub-populations. Looking at these outliers we found that logs belonging to these clusters had unusual behaviors; e.g., 11.5 clicks on average (SD = 19.5), extremely long cursor trajectories of 12797.4 px (3130.9), and so on.

Pages in cluster #6 concentrated the biggest sub-population (46% of the data). We found short-term sessions of M = 30 s (SD = 132.9) with "one-click" browsing patterns of 1.1 clicks (0.6). Scrolling reached 40% (20) of the users' browser viewport and mouse range comprised 181.8 px (128.9) and 120.7 px (105.6) in horizontal and vertical axes, respectively. Thus, logs belonging to this cluster could be classified as "basic presence" pages, supporting somehow the evidence of the typology of the tracked pages (i.e., an informational website).

[1] In the literature, it is also mentioned as the 'gap statistic' [Tibshirani et al., 2001].

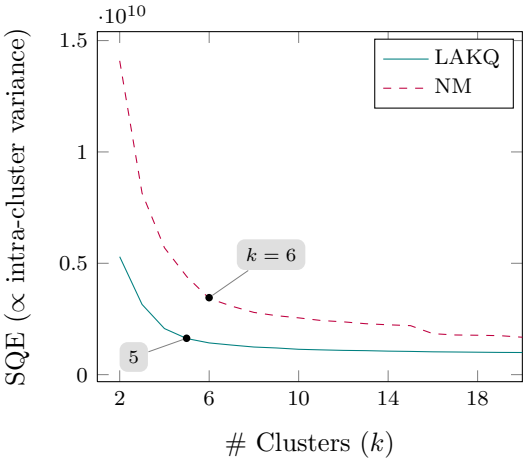


Figure 3.8: Clustering LAKQ and NM datasets, following the same criterion depicted in Figure 3.7. We identified 5 and 6 clusters to be the most informative number of classes to describe the pages in each dataset, respectively.

The population of cluster #3 was the least dispersed overall (16% of energy). In-page interactions lasted 45.7 s (125.96), issuing 1.5 clicks (1.6) per session on average. Pages belonging to this group were found to be browsed by relatively active users, e.g., mouse distance: 7.1 px (6.7), mouse motion: 16% (13), vertical scroll of 65% (23). Therefore, we hypothesize that these pages were be the most familiar for the users. Although we do not have such ground truth data to back up this claim.

Pages in cluster #2 showed metrics related to cluster #3, with similar power-law distributions. However, users in this cluster spent more browsing time, which was also more dispersed overall: 1.3 min (3.5), and clicked more: 2.33 (2.34). Pages were scrolled considerably more than the half of their browser's viewport: 76% (22). Together with the rest of considered metrics, this fact led us to conclude that pages in this cluster were the most interesting for the users.

Profiles in NM and LAKQ corpora Instead of performing a detailed analysis of each cluster found akin the OTH corpus as described above, we shall depict some interesting observations.

Cluster	Population	%	Energy (SQE)	%	Variance
1	159	13	$2.5{\cdot}10^8$	15	$1.6{\cdot}10^6$
4	632	53	$4.1{\cdot}10^8$	24	$6.4{\cdot}10^5$
5	346	29	$4.5{\cdot}10^8$	27	$1.3{\cdot}10^6$
Avg. Total	1178	100	$1.6{\cdot}10^9$	100	$1.3{\cdot}10^6$

Table 3.3: Clusters found in the LAKQ dataset. Two outliers (clusters #2 and #3) were identified.

Regarding Table 3.3, the biggest cluster (#4, 53% of the data) was surprisingly

not the most distorted overall. We observed that all meaningful clusters found were more or less similar in terms of dispersion, which is a convenient feature of K-means. What is specially interesting, however, is that vertical scrolling often overpassed 100% of the browser viewport. Taking also into account the mouse ranges, centroids, and entry/exit coordinates in these groups, we speculate that most visitors were using (moderately) large displays. This hypothesis was then verified by observing that the average screen resolution was 1208.9 (203.5) x 860.7 (118.8) px.

Cluster	Population	%	Energy (SQE)	%	Variance
2	1697	30	$8.9{\cdot}10^8$	26	$5.2{\cdot}10^5$
3	968	17	$5.8{\cdot}10^8$	17	$6.1{\cdot}10^5$
6	2132	38	$9.1{\cdot}10^8$	26	$4.2{\cdot}10^5$
Avg. Total	5557	100	$3.4{\cdot}10^9$	100	$6.1{\cdot}10^5$

Table 3.4: Clusters found in the NM dataset. Three outliers (clusters #1, #4, and #5) were identified.

As observed in Table 3.4, similar to the OTH dataset, we found three clusters (#2) in the NM dataset that were clear outliers. Again, we remark the usefulness of using behavioral clustering for isolating sub-populations in large datasets. On the other hand, though, the remaining clusters showed more consistent behaviors, comprising between 17% and 26% of the overall cluster energy. Overall, it was interesting to observe that the proposed metrics lead classical clustering to find the same number of classes as in the previously studied datasets. We elaborate more on this below.

Discussion

Our study threw some interesting suggestions. First, using this clustering framework allows to focus on a small number of groups to describe the vast majority of the pages of a website. For instance, in the OTH corpus the 3 main clusters found represent 95% of the browsed pages. Similarly, by looking at the same number of clusters, we can explain 89% and 93% of the pages in LAKQ and NM datasets, respectively. Second, as previously commented, our method allows to describe web pages in a completely different way, i.e., from the user interactions' point of view, instead of the usual structure/content/usage triad. This knowledge has an interesting potential to be used to compare cross-site browsing behaviors, or predict interest of non-browsed pages. Third, using the information implicitly embedded in user's interactions may help webmasters to redesign the most important pages, in terms of in-page interactions. This way, if individual personalization is not possible, users could browse the site at the same performance level to a greater or a lesser extent. Fourth, we found

that the user sample we tracked at each website was often a mixture of distributions. This evidence encourages to be cautious in using logging tools or intuitions that assume a normal distribution for all users.

As observed, exploiting the browsing context from user behavior may serve as a useful complement to current web mining techniques. Further suitability of this work relates to any system that taps knowledge about the user, e.g.: information retrieval, relevance feedback, document organization, or usage inference, just to name a few. Armed with this awareness, one could carry out novel research studies on user modeling and related applications.

3.3.2 Classifying Human Actions

In this case, we chose a straightforward classification task to test the WKM algorithm in isolation. We wanted to test how data sequentiality may affect the performance of a recognizer. To this end, we used the Localization Data for Person Activity dataset [Kaluža et al., 2010] form the UCI Machine Learning Repository [Asuncion and Newman, 2007]. In this corpus, 164860 data points were captured from 5 people wearing 5 active RFID tags (both ankles, belt, and chest). Up to 11 human actions were represented as a time series of `x,y,z` coordinates of such 5 body parts.

Note that, while there is an important number of works tackling the problem of classifying human actions, we chose this corpus to show the capabilities of WKM as a simple and accurate compression tool for a complex, real-world task. To this end, each human action is represented as a vector of a fixed number of "elementary actions", where each elementary action is, in turn, a cluster mean vector obtained by clustering the original sequence of action samples (`x,y,z` coordinates). Once each action is represented as a fixed dimension vector, many simple classifiers can be adequately used, among which we chose the well-known Nearest-Neighbor (NN) classifier.

Method

To characterize each activity, the `x,y,z` coordinates of all sensors were merged into a single 12-dimensional feature vector sample $\boldsymbol{x} = (x_1, y_1, z_1, \ldots, x_4, y_4, z_4)^T$. So a trajectory was defined as the sequence $X = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, where n is the number of samples in X.

Unfortunately, the dataset did not include the same number of instances per sensor. Therefore some of the composed trajectories had extremely different number of 12-dimensional vectors (e.g., some had just two vectors and others had more than 800). We needed thus to build a more comparable dataset; so, while composing each trajectory we verified that it had at least 10 samples.

Eventually we obtained 125 trajectories of 162 samples on average (SD=138.6), belonging to one of the following 5 classes: 'falling', 'lying', 'on-all-fours', 'sitting', and 'walking'. There were 25 trajectories per class. The features of the dataset used in the experiments are depicted in Table 3.5.

Trajectories	125
Mean samples per trajectory	162
Dimension of sample vectors	12
Classes (actions)	5
Number of trajectories per class	25

Table 3.5: Features of the dataset used in the WKM experiments.

Vector Representation We ran our implementation of WKM to cluster each trajectory into a variable number of segments ($k \in \{2, 4, \ldots, 20\}$) and with different cluster proportions ($\delta \in \{0, 0.2, \ldots, 1\}$). We also compared WKM with two well-known versions of K-means: the classical *Duda&Hart*'s algorithm [Duda and Hart, 1973] and the popular *Lloyd*'s version [Lloyd, 1982], using both random and TS initializations. When initializing randomly we performed up to 5 times each experiment, in order to mitigate the effects of chance, and computed the average values.

The cluster means obtained by k-clustering each action data sequence were stacked into a $3 \cdot 4 \cdot k$ dimensional feature vector, i.e., a $12\,k$-dimensional vector. For those trajectories with less samples than the desired number of segments, (i.e., when $k > n$) we used singleton clusters instead (i.e., $k = n$) and the missing dimensions were filled with zeros. As we will see below, this fact had clear repercussions when classifying some trajectories with $k > 10$ (ten was the minimum number of vectors in all trajectories), specially in terms of classification error.

Nearest Neighbor Classifier The simple and well-known 1-NN classifier with Euclidean distance was adopted to classify vector-represented action trajectories. As previously pointed out, each class was represented by a number of prototype trajectories. Each test trajectory was classified into the class of its nearest neighbor prototype.

In these experiments, we employed the C++ ANN library [Mount and Arya, 1998] for NN searching, with its basic, exact search option. Given the relatively small number of available trajectories overall, we adopted the leaving-one-out training and testing procedure.

Results

The first experiment was aimed at studying the behavior of different algorithms when minimizing SQE and increasing the number of clusters. Results are shown

in Figure 3.9. As expected, in all cases SQE decreases monotonically with increasing number of clusters. It is interesting to note that K-means algorithms achieve a (slightly) lower SQE than WKM, which is explained by the lack of sequentiality restrictions, that otherwise WKM imposes on the data.

In the next experiment we studied the ability of different clustering algorithms to behave as data preprocessors, in order to obtain simplified vector-represented trajectories for classification purposes. We considered the case when a sensor trajectory is segmented into just one single cluster ($k = 1$) as the baseline; that is, each trajectory is represented by a 12-dimensional vector corresponding to the average of all its trajectory samples. In that case, the classification error was as low as 9.6%, which is reasonable given the nature of the activities involved (e.g., the position of "lying" and "sitting" should differ greatly at least in the average `z` coordinate of each sensor).

Results for other values of k are shown in Figure 3.10. As expected, certain segmentations performed better than others for each algorithm, but a particularly adequate number of elementary actions seems to be 6 in most cases. Interestingly, WKM is the method that better puts this fact forward. We observed that accuracy degraded noticeably for $k > 10$, to the point that for $k = 20$ error rates were above 50% for all classifiers—for the reason explained in Section 3.3.2. Also, as observed, the randomly initialized versions were the worst performers.

In order to better understand the impact of the δ threshold of WKM, we repeated the previous experiment for different values of this threshold. Figure 3.11 shows the influence of δ in the recognition accuracy. We see that by tuning this parameter WKM results can be further improved, with a best result of 3.2% error rate for six elementary actions. Finally, regarding the computational cost of each algorithm, as shown in Figure 3.12, WKM behaves much better than its peers.

Table 3.6 summarizes the results discussed so far. Classical K-means algorithms do not help overcoming the trivial baseline (just one cluster). In contrast, WKM achieved a recognition accuracy of 97%, which represents a 66% improvement over the baseline. WKM is borderline statistically significantly better than all compared methods [$\chi^2_{(7,N=125)} = 4.44, p = .07$]. Most interestingly, the improvements introduced by WKM are achieved along a huge computational cost reduction (more than one order of magnitude) with respect to K-means algorithms. We can conclude that WKM was the best performer among its peers, and that results confirmed our expectations.

Discussion

As can be observed in the figures, WKM gives very competitive error rates at a low computational cost. Therefore, our experimental results show that WKM is

Figure 3.9: Sum of squared errors against number of segments. Each value is averaged for all trajectories (activity × person × trial). As expected, the segmentations achieved by WKM have higher distortion than those of classical K-means, since the former imposes a strong sequential restriction, while the latter does not.

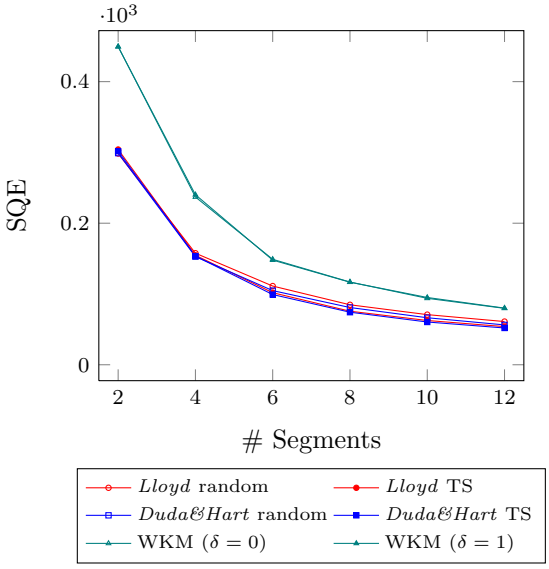
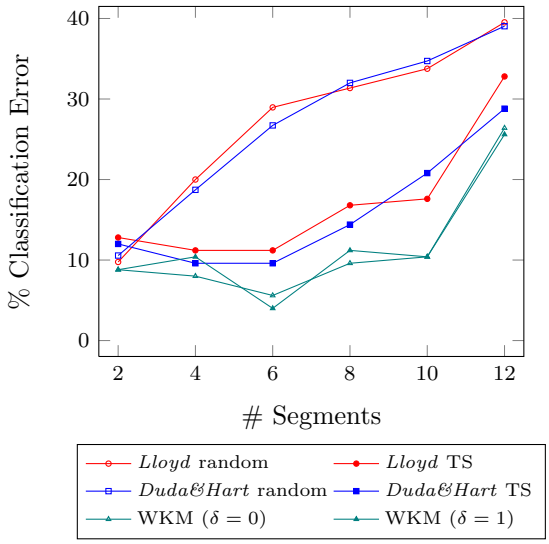


Figure 3.10: We performed variations to three alternatives for clustering trajectories: The *Duda&Hart*'s algorithm and the *Lloyd* version, using both random initialization and trace segmentation, and the WKM algorithm using two extreme distortion thresholds.



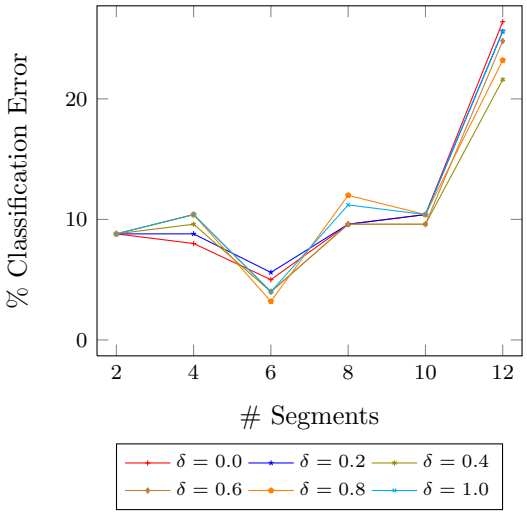


Figure 3.11: WKM classification error. We used different δ thresholds for each tested number of segments. The best accuracy was achieved when using $k = 6$ for all threshold values.

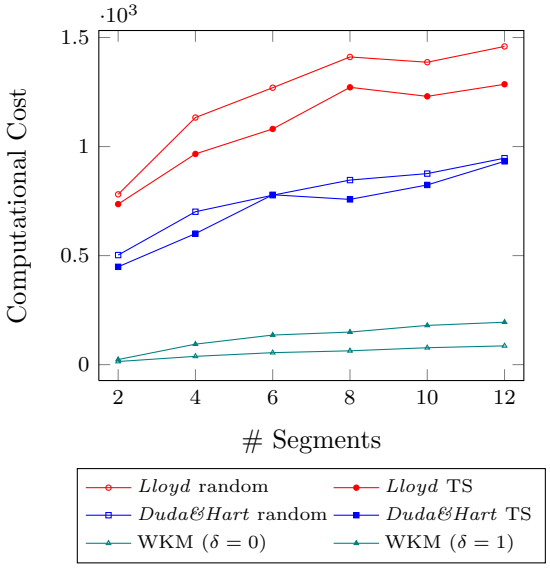


Figure 3.12: Computational cost against number of segments. Cost is estimated as number of times Eq. (3.5) is executed. For *Lloyd* versions, cost was computed as the number of times each algorithm tested if cluster means did change.

Algorithm	Best k	% Error	Cost
Baseline	1	9.6	—
Lloyd random	2	9.8	796
Lloyd TS	6	11.2	1080
Duda&Hart random	2	10.6	448
Duda&Hart TS	6	9.6	778
WKM $\delta = 0.0$	6	5.6	54
WKM $\delta = 0.8$	6	**3.2**	71
WKM $\delta = 1.0$	6	4.0	135

Table 3.6: Summary of sequential clustering results. Bold value indicates that it is the best result among all methods being compared.

an interesting approach for lowering both classification error and computational cost regarding to using other comparable clustering alternatives.

It is worth pointing out that all algorithms initialized with TS allow to find the "natural" number of classes. However, as shown in Figure 3.10, for WKM this number in turn corresponds to the lowest classification error rate in all cases (see also Table 3.6).

Additionally, we have shown that WKM ensures monotonic improvement and finite assignments in a sequential fashion, which translates to convergence to a good local minimum in which trajectory segments are well-defined. This can be leveraged in some interesting applications, as we shall expose as follows.

Online Handwriting Our clustering technique can be used as a preprocessing step for online text recognition. As illustrated in Figure 3.1, the obtained (well-formed) segments capture pen-stroke regularities which can be advantageously exploited by existing handwritten recognition approaches to increase character recognition accuracy [Leiva and Vidal, 2012].

Eye/Mouse Tracking This algorithm entails a reliable contribution to clustering eye movements on aggregated data; e.g., both heatmaps and areas of interets (AOIs) are computed by distance-based clusters, and therefore they do not distinguish between long-time fixations of a single person or short-time fixations of a group of people.

Motion Segmentation The storage and transmission of motion tracking content is a problem due to their tremendous size and the noise caused by imperfections in the capture process. Thus, one could use our method for a more compact representation of these (large) data.

In general, any discipline that would handle ordered data sequences could benefit from our approach; e.g., human motion classification from surveillance cameras or automatic video key frame extraction.

3.4 Conclusions and Future Work

This chapter has covered behavioral clustering, a broad term that refers to the task of automatically labeling and classifying user behavior, which was evaluated on two different tasks with a series of real-world datasets.

In the first task we were able to discover "hidden" profiles on websites, according to how users behave while browsing. We have demonstrated that this technique can be used to organize and describe websites from the user interactions' point of view. This technique can also be used as a measure of similarity between web pages, to evaluate their design in an automated fashion, or to discover outliers. We believe that this work opens a new door to novel approaches on web behavior studies.

Lines of future work regarding web page classification according to (implicit) interaction metrics include inferring behavior of non-browsed pages and finding related websites based on user interactions. The metrics we used for clustering are related to cursor activity, because cursor data are easy to collect and no special instrumentation is required on client side. However, user interaction is inherently multimodal. Thus, other related input signals such as eye movements could (and should) be taken into consideration, and be incorporated to more sophisticated web profiles. This way, one may complement studies of quantitative/qualitative nature, improving thus the usability and usefulness of websites, and being able to extend this methodology to related fields such as web applications or software products.

In the second task, we have presented a novel revisitation of the K-means algorithm, specially suited for sequentially distributed data. We have successfully used this approach to automatically identify human actions derived from raw sensor data. By taking into account that data are sequentially given, our proposal, WKM, behaves much better than classical clustering algorithms. One obvious reason why using a cluster representation may have advantages over working with raw sensor data is the evident size reduction, which in turn may enhance the ease of storage, transmission, analysis, and indexing. Moreover, extending this notion to the analysis of trajectories reverts in another significant advantage: having a good and compact representation of a data sequence makes it more invariant to noise or distortions in such data. This fact has been backed up by our experimental results, lowering both classification error and computational cost regarding to using other comparable clustering alternatives.

As stated in this chapter, a critical step for (adequately) clustering sequential data with WKM is the initialization of segment boundaries. We used the TS

technique, although other algorithms that ensure a sequential distribution may be also helpful. For instance, we could use an equispaced boundary initialization instead. Future work will be focused on removing the (optional) δ parameter from the algorithm, and instead learning automatically the best value for a given cluster configuration. Further research on WKM will be leaned toward an optimum procedure of choosing the number of clusters. We hope that our work may encourage researchers and practitioners to apply this algorithm to a wealth of new problems and/or domains.

Bibliography of Chapter 3

R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pp. 94–105, 1998.

A. Asuncion and D. J. Newman. UCI machine learning repository, 2007. Available at http://www.ics.uci.edu/~mlearn/MLRepository.html.

F. I. Bashir, A. A. Khokhar, and D. Schonfeld. Object trajectory-based activity classification and recognition using Hidden Markov Models. *IEEE Transactions on Image Processing*, pp. 1912–1919, 2007.

J. C. Bezdek and N. R. Pal. Some new indexes of cluster validity. *IEEE Transactions on System, Man and Cybernetics*, 28(3):301–315, 1998.

M. Claypool, P. Le, M. Wased, and D. Brown. Implicit interest indicators. In *Proceedings of the 6th international conference on Intelligent user interfaces (IUI)*, pp. 33–40, 2001.

D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(4):224–227, 1979.

T. G. Dietterich. Machine learning for sequential data: A review. In *Proceedings of the Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pp. 15–30, 2002.

R. Dubes. *Handbook of Pattern Recognition & Computer Vision*, chap. Cluster analysis and related issues, pp. 3–32. World Scientific Publishing Co., Inc., 1993.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*, chap. Unsupervised Learning and Clustering, pp. 517–599. John Wiley & Sons, 2001.

J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3:32–57, 1973.

J. C. Dunn. A cluster separation measure. *Journal of Cybernetics*, 4:95–104, 1974.

C. Fraley. Algorithms for model-based gaussian hierarchical clustering. Tech. Report 311, Department of Statistics, University of Washington, 1996.

V. Guralnik and G. Karypis. A scalable algorithm for clustering sequential data. In *Proceedings of IEEE International Conference on Data Mining*, pp. 179–186, 2001.

G. Hamerly and C. Elkan. Learning the k in k-means. In *Proceedings of the seventeenth annual conference on neural information processing systems (NIPS)*, pp. 281–288, 2001.

M. Holub and M. Bielikova. Estimation of user interest in visited web page. In *Proceedings of the 19th international conference on World wide web (WWW)*, pp. 1111–1112, 2010.

L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

A. K. Jain. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8): 651–666, 2010.

A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):1–60, 1999.

B. Kaluža, V. Mirchevska, E. Dovgan, M. Luštrek, and M. Gams. An agent-based approach to care in independent living. In *Proceedings of the International Joint Conference on Ambient Intelligence (AmI)*, pp. 177–186, 2010.

L. Kaufman and P. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, 1990.

J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory clustering: a partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data (SIGMOD)*, pp. 593–604, 2007.

L. A. Leiva. Mining the browsing context: Discovering interaction profiles via behavioral clustering. In *Adjunct Proceedings of the 19th conference on User Modeling, Adaptation, and Personalization (UMAP)*, pp. 31–33, 2011.

L. A. Leiva and E. Vidal. Assessing users' 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 and E. Vidal. Revisiting the K-means algorithm for fast trajectory segmentation. In *Proceedings of the 38th International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2011.

L. A. Leiva and E. Vidal. Simple, fast, and accurate clustering of data sequences. In *Proceedings of the 17th international conference on Intelligent User Interfaces (IUI)*, pp. 309–310, 2012.

S. Lloyd. Least squares quantization in PCM. *IEEE Transations on Information Theory*, 28(2):129–137, 1982.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, 1967.

K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, 1979.

G. W. Milligigan and M. C. Cooper. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50(2):159–179, 1985.

D. Mount and S. Arya. ANN: library for approximate nearest neighbor searching, 1998. Available at http://www.cs.umd.edu/~mount/ANN/.

F. Murtagh. A survey of recent advances in hierarchical clustering algorithms which use cluster centers. *Computing Journal*, 26(1):354–359, 1984.

B. Poblete and R. Baeza-Yates. Query-sets: Using implicit feedback and query patterns to organize web documents. In *Proceedings of the 17th International Conference on World Wide Web (WWW)*, pp. 41–50, 2008.

C. Sugar. *Techniques for Clustering and Classification with Applications to Medical Problems*. PhD thesis, Department of Statistics, Stanford University, 1998.

R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 63(2):411–423, 2001.

P. Trahanias and E. Skordalakis. An efficient sequential clustering method. *Pattern Recognition*, 22(4):449–453, 1989.

C. J. Veenman, M. J. T. Reinders, and E. L. Baker. A maximum variance cluster algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1273–1280, 2002.

J. H. Ward. Hierarchical grouping to optimize an objective function. *Journal of American Statistics Association*, 58(301):235–244, 1963.

J. Yu. General C-means clustering model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1197–1211, 2005.

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.

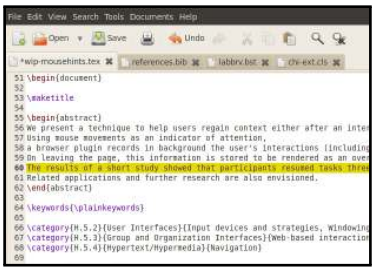


Figure 4.1: A usual approach for easing attention shifts in tabbed interfaces (in this case, a text editor). The last edited line is automatically marked by highlighting the text background, so when the user switches back to the current tab she can realize faster where she left writing.

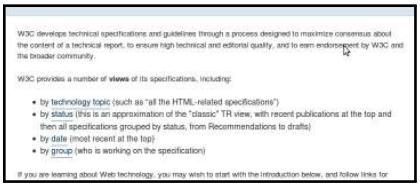
4.2 MouseHints

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.

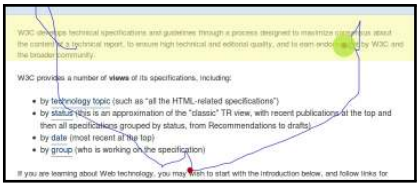
System Basis Only one web page and a corresponding tab representing it can be active at the same time in a browser window. Tapping this fact, our system tracks in the background the mouse activity in the current tab. Upon switching such a tab back, the system "hints" a subset of the last cursor movements (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.

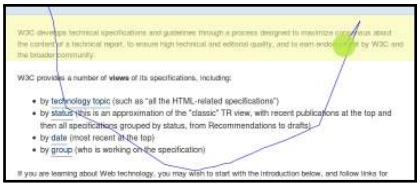
Implementation MouseHints was developed as a Firefox extension since such browser has a powerful mechanism that made it relatively easy to code and test. The browser interface was structured in XUL (XML UI Language). Both the logic and tracking algorithms were both written entirely in JavaScript. The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



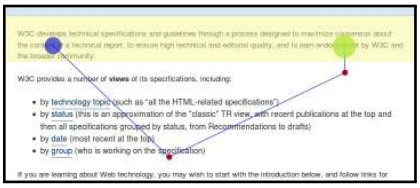
(a) Original test page.



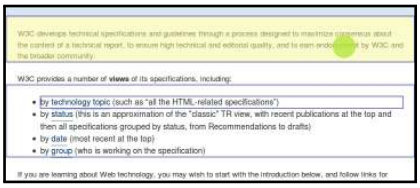
(b) Raw trajectory



(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).

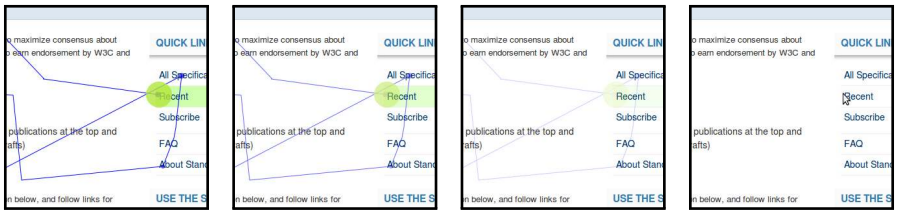


Figure 4.3: Visualization example. The overlay fades out in 500 ms, allowing for regular interaction with the page.

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

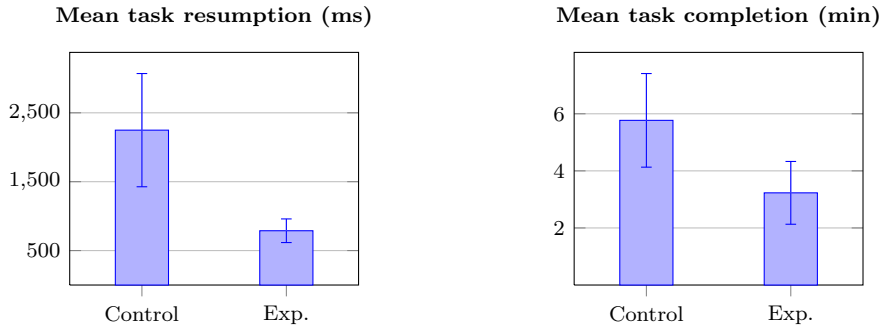


Figure 4.5: Between-groups efficiency comparison. Error bars denote 95% confidence intervals.

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.

Chapter 5

Adaptive User Interfaces

In computing systems, technology alone cannot survive without adequate user interfaces. To maximize the benefits that usable interfaces bring to users, often developers try to target as many people as possible. However, attempting to create UIs by following the *one-size-fits-all* approach is doomed to fail if an application is intended to be exposed to an arbitrary audience—take for instance web pages or mobile applications. Therefore, we must look for automated solutions.

This chapter proposes a novel approach to automatic UI adaptation that leverages implicit interactions to weight the importance of the information supplied with estimated priorities in user activity. This way, by analyzing information that is submitted with little or no awareness (e.g., mouse movements, clicks, keystrokes), elements where users focus their interaction are incrementally mutated. While this is still a work in progress, preliminary results indicate that this method has an interesting potential to build self-adaptive UIs.

Chapter Outline

5.1 Introduction

In computing systems, technology alone cannot survive without adequate user interfaces. Personalization and customization have been widely promoted by UI design theories but seldom few of them are put into practice. The vast majority of UIs are visually-oriented and assume that users do not have functional impairments of special requirements. Caveats, standards, and best practices have evolved on where to place layout widgets, navigation items, and body content. As such, to maximize the benefits that usable interfaces bring to users, often developers try to target as many people as possible. However, attempting to create UIs by following the *one-size-fits-all* approach is doomed to fail if an application is intended to be exposed to an arbitrary audience. Take for instance web pages or mobile applications, where, in addition, the range of screen sizes and the rendering possibilities are exceedingly large.

UI adaptation is about exploiting some features of the application and avoiding others. For instance, the mobile space has an incommensurable range of devices, and content often renders better when tailored to specific device characteristics. Another part of adaptation requires working around problems found in specific parts of the UI; e.g., elements that may cause confusion or frustration to first-time users and so on. A more drastic option is to build a separate UI for each user, but a manual approach is impractical and definitely not scalable. Also, continuously performing usability tests to assess new changes committed on the application is very time-consuming. Therefore, we must seek automated adaptation solutions.

Traditionally, UI adaptation techniques can personalize the layout presentation (e.g., modifying font sizes or applying some accessibility guidelines), but unfortunately the changes they perform operate from a global perspective. Some proposals that involve active end user manipulation have been considered; e.g., [Bolin et al., 2005]. Nonetheless, user-driven customization requires to perform additional activities beyond the main purpose of using the application. Some researchers [Arroyo et al., 2006; Atterer et al., 2006; Claypool et al., 2001] have demonstrated that every user interaction can contribute to enhance the utility of the system, therefore alternative adaptation approaches without burdening the user can be derived. What is more, as stated by Gajos and Weld [2004], the rendering of an interface should reflect the needs and usage patterns of their users. This work is inspired by these ideas.

We propose a novel approach that is based on implicit HCI to weight the importance of the information supplied with estimated priorities in user activity. This way, by leveraging information that is submitted with little or no awareness (e.g., mouse movements, clicks, keystrokes), elements (*widgets* from here onwards) where users focus their interaction are incrementally mutated. Specifically, due to the fact that exertions are preceded by attention most of the time (see Section 1.1.1), the importance of an interaction toward a specific widget

is measured as the proportion of UI-generated events on that widget between consecutive sessions, as described in Section 5.3.

5.2 Related Work

The idea of adapting the UI of applications or even full websites according to user interactions is not new (see, e.g., [Zhang, 2007]). However, practical examples have been too scarce so far. Despite considerable debate, automatic adaptation of UIs remains a contentious area [Gajos et al., 2008]. Commonly cited issues with adaptive interfaces include lack of control, predictability, transparency, privacy, and trust [Findlater and McGrenere, 2008].

It is commonly agreed that adaptive systems should accommodate the UI to the user, but also that doing so automatically is a non-trivial problem. We believe that adaptation should be both transparent and discreet, so that the changes introduced to the UI do not confuse the user. We also believe that adaptation should not interfere with the internal structure of the application.

Probably the major advances in the field of automatic adaptation of UIs are the ones carried out by Gajos and co-authors [Gajos and Weld, 2004; Gajos et al., 2007, 2008], where adaptation is approached as an optimization problem. However, their experiments were performed on form-based layouts, by modeling widget constraints, and choosing the best alternatives from a defined set of UI elements (e.g., sliders, combo boxes, radio buttons, etc.). Other types of applications such as web pages are nevertheless a completely different matter. Their dynamic nature per se makes the automatic adaptation a challenging task.

On the Web, with the exception of customizing font preferences, browsers do not provide end users with substantial control over how web pages are rendered. This way, researchers have proposed different approaches to layout adaptation that mainly involve user's manual work. Ivory and Hearst [2002] employed learned statistical profiles of award-winning websites to suggest improvements to existing designs; however, changes would be manually implemented. Tsandilas and Schraefel [2003] introduced an adaptive link annotation technique, although it required the user to perform direct manipulation of a middleware application. Notable approaches in this direction include the work of Bila et al. [2007], where the user must actively modify the layout contents. Kurniawan et al. [2006] proposed to override the visual tier of a web page with custom style sheets, but unfortunately updates had to be performed by hand. Now that web standards have minimized browser inconsistencies, this approach can be automatically exploited to automate the adaptation of web design (and other applications, as discussed later) without disrupting users' interaction habits.

(a) Original page **(b)** Automatically adapted design

Figure 5.1: An example of website design modifications. Changed parts are numbered in Figure 5.1b. ❶ headline text: `font-size`, `padding-top`; ❷ navigation menu: `font-size`; ❸ welcome paragraphs: `font-size`; ❹ 'read more' links: `color`; ❺ 'online booking' heading: `color`; ❻ submit button: `font-weight`; and ❼ 'special menu' div: `margin-top`.

5.3 ACE: An Adaptive CSS Engine

Our approach, being based on implicit interaction, allows to gather much usage data without burdening the user. On the other hand, though, collected data are potentially noisy and prone to some errors if not treated adequately. For that reason, the novelty of this approach is two-fold: *1)* to let the webmaster decide *which* elements are going to be adapted; and *2)* to automatically apply slight modifications to the rendering of UI elements based on *how* the user has interacted with them. This way, the system will try to invisibly improve the user-perceived performance toward a UI (Figure 5.1).

The main difference with other state-of-the-art interface adaptation techniques is ours relies on the developer (or webmaster) control to accommodate the appearance of the UI (or page) to the users in a transparent way. This way, our approach aims to focus rather than distract the user.

5.3.1 Rationale

With the growing popularity of web-based applications, the Cascading Style Sheets (CSS) paradigm has been widely adopted by several programming environments beyond the browser. For instance, it is possible to use CSS in Java[1], GTK+[2], and Qt[3]. CSS allows attaching styles to the application, decoupling

[1] http://weblogs.java.net/blog/2008/07/17/introducing-java-css
[2] http://gnomejournal.org/article/107/styling-gtk-with-css
[3] http://doc.qt.nokia.com/4.3/stylesheet.html

the data model and its presentation. This motivated us to develop ACE, an Adaptive CSS Engine in which adaptation operates by automatically overriding the rendering of widgets, by simply modifying their CSS. The technique was first introduced by Leiva [2011], and has been now reformulated to generalize to structured applications (e.g., document object models and scene graphs).

5.3.2 Overview

ACE leverages implicit interactions to incrementally mutate the appearance of interacted widgets (e.g., DOM elements). The importance of an interaction toward a specific widget is measured as the proportion of UI-generated events on that widget between consecutive sessions. Implicit interaction is used thus as a proxy of user attention. The idea is to introduce ephemeral changes that can be easily incorporated and do not alter the UI design in a way that it might confuse the user [Leiva, 2011, 2012a].

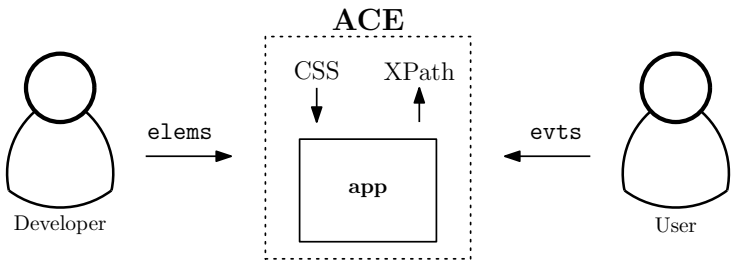


Figure 5.2: Workflow diagram. ACE tracks elements indicated by the developer. When the user access an application, UI events translate interacted elements into XPath notation (or a similar representation) for later storing. On returning to the application, the CSS properties of such stored elements are restyled accorded to computed scores.

ACE was written as a completely self-contained JavaScript (JS) program that restyles *numerical* CSS properties, i.e., those related to:

- **Dimensions** (e.g., `font-size`, `margin-top`). These properties often do have a unit of measure, e.g., `16px`, `2.5em`, or `20%`, which is preserved once they are adapted.
- **Colors** (e.g., `background-color`, `border-color`). These properties do have an hexadecimal representation,which is specified either by a keyword (e.g., `"red"`) or by a numerical RGB specification (e.g., `#RRGGBB` or `rgb(R,G,B)`).

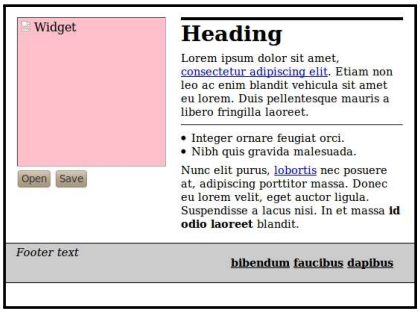
The main features of ACE are summarized in the following list:

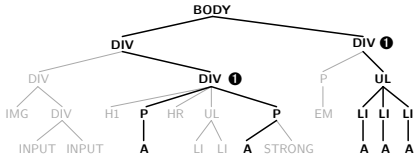
- Does not require end user intervention.
- Supports desktop, touch, and mobile web clients.

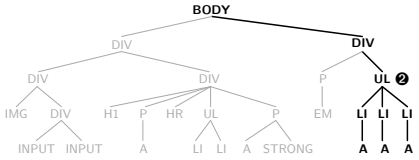
(a)

```
ACE.adapt({
  "div a": ["font-size", "color"], ❶
  "p ul" : ["font-weight", "margin"]
});
```

```
ACE.adapt({
  "div + a": ["font-size", "color"],
  "p + ul" : ["font-weight", "margin"] ❷
});
```







(b) pattern: `E F`

(c) pattern: `E + F`

Figure 5.3: Original page design (5.3a) with an overlaid mouse behavior that may cause different adaptation possibilities, according to the following CSS combinator patterns: [5.3b] F elements that are descendants of E elements; [5.3c] F elements immediately preceded by E elements; *Top row*: Sample JSON syntax. *Middle*: Corresponding page changes. *Bottom row*: DOM tree traversals, highlighting in bold the matched paths. Any combination of CSS selectors is supported, e.g., `"div + p.foo > span a:first-child"`.

- Any combination of CSS selectors can be used.
- Modifications are incrementally applied, ensuring that they are not intrusive for the user.
- Adaptation can be performed once the DOM is parsed or the application is fully loaded, so that third party or JS-controlled modifications are also

supported.
- Since the system has a *user interaction history*, it can populate adaptation to other widgets that share a similar structure.

5.3.3 Adaptation Protocol

Initially, the developer indicates which widgets and which properties can be restyled by the system, by means of straightforward JSON notation (see sample code snippets in Figure 5.3). Later, when the application is loaded, event listeners will track such widgets in the background. While using the application, the system "learns" from user interactions, so that the next time the application is loaded, the visual appearance of the widgets the user has interacted most with is subtly modified. Finally, when the user leaves the application, interaction data are serialized and stored into a local database. Figure 5.4 summarizes the architecture of this framework.

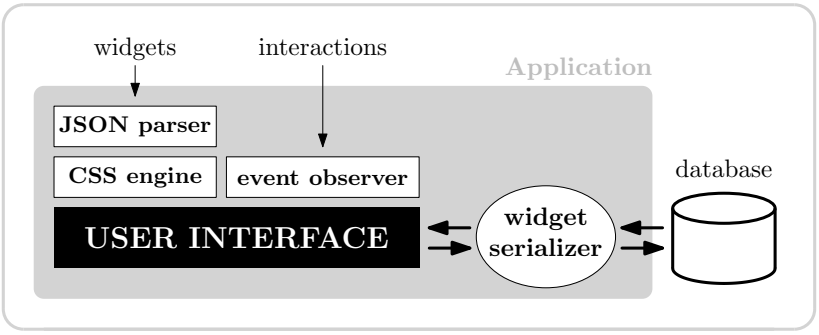


Figure 5.4: System architecture. Adaptable widgets are indicated by the developer, which will be modified according to how users interact with the application.

5.3.4 Implementation

A very simple API was designed to invoke the system. ACE exposes two public methods: `listen()` and `adapt()`. The former allows the developer to prioritize the importance of UI events (e.g., Should a `mousemove` event be assigned lower priority over a `click` event?). The latter takes two arguments (Figure 5.5): a configuration object and a context (the whole application by default).

Under the hood, the elements that were specified in the configuration object as CSS selectors are retrieved by means of the `querySelectorAll()` method or a similar alternative (depending on the programming language). Interaction data are then classified into different event lists, e.g., hovered, typed, scrolled, or tapped elements; where each list member is composed of a serialized widget representation as a key (to allow retrieving them later on subsequent user visits, see bottom rows of Figure 5.3) and an interaction score as a value. The scoring

```
interface ACE {
  void listen(Object eventList, Boolean keepOtherPriorities);
  void adapt(Object config, Object context);
}
```

Figure 5.5: ACE's API definition in Interface Description Language (IDL).

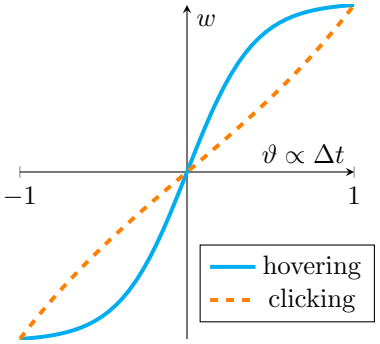


Figure 5.6: Weighting interactions example. Hovering is weighted according to $w = \tanh(\lambda\vartheta)$, while clicking is weighted as $w = \sinh(\lambda\vartheta)$. The parameter λ allows to tune the slope of both curves.

scheme is described in the next section. Basically, a score is proportional to the number of browser-generated events, or, in other words, how many times the user has interacted with UI elements.

Finally, data are persistently stored on the client side by means of an abstraction layer of different storage backends (e.g., `localStorage`, `IndexedDB`, or equivalents), so that the users' privacy is completely under their control; e.g., they may opt to configure their application or browser to restrict access to the storage context, or automatically delete stored data after some time.

5.3.5 Interaction Scoring Scheme

As commented above, each interacted element is assigned a score s, which depends on the event type. For instance, `mousemove` events are triggered in much more quantity than `mousedown` or `keyup` events, and as such they should be weighted accordingly. Let n_i be the number of times an event of type i was fired for a certain widget, and let N be the number of all fired events during application usage. The assigned score for that event is

$$s_i = \zeta(n_i/N) \tag{5.1}$$

where $\zeta(\cdot)$ is a symmetric sigmoid function. The idea is to get scores follow a non-linear distribution, in order to ensure that adaptation is smoothly applied.

Note that if an element receives different types of interactions (e.g., an `input` text field can listen to `click`, `focus`, or `keydown` events) then its scores need to be fused in order to compute a single value. ACE uses the weighted mean

as a fusion scoring method:

$$s = \sum_{i=1}^{m} w_i s_i \quad \text{with} \quad \sum w_i = 1 \tag{5.2}$$

where m is the number of computed scores for that element.

The value v of a CSS property is then modified based on the following style function:

$$v = v(1 + s) \tag{5.3}$$

On subsequent access to the UI, the new scores s'_i and how they will affect the CSS properties are both updated as follows:

$$\begin{aligned} s'_i &= \zeta(n'_i/N) - s_i \\ v' &= v(1 + s') \end{aligned} \tag{5.4}$$

According to equations (5.3) and (5.4), when a user loads an application for the first time, elements are rendered as they were designed, as the system has no information about previous interactions ($s_i = 0 \ \ \forall i$). Then, when returning to the application the system will react accordingly, i.e., modifying the value of those CSS properties specified by the webmaster based on the amount of user's interactions.

Given that scores are bounded to the interval $(-1, 1)$, a score of, say, 0.05 for a `margin-top` property will be interpreted as "increasing by 5% the value of the top margin." Conversely, a score of -0.1 for a `color` property will be interpreted as "decreasing by 10% (the contrast or saturation of) the font color." This way, it is not possible to alter the visual properties significantly, since adaptations are incrementally applied. Event lists are the only user information stored in the local database.

5.4 Fostering Creativity

ACE also introduces an interesting framework to find inspirational examples for redesigning UIs. Typically, the primary purpose of prototyping tools is to provide feedback to define a design earlier, when there is inadequate information to choose one solution over another. However, once the design of an application or website leaves the testing phase and moves to production, it hardly ever gets substantially modified. Rather, it follows a cycle of subtle iterative improvements. At this stage, surprisingly, few methods seldom support incrementally revisiting different versions of the *same* solution.

In this line, some work has been done in generating design *alternatives* to assist the user in the design process, i.e., to get the "right design", for instance,

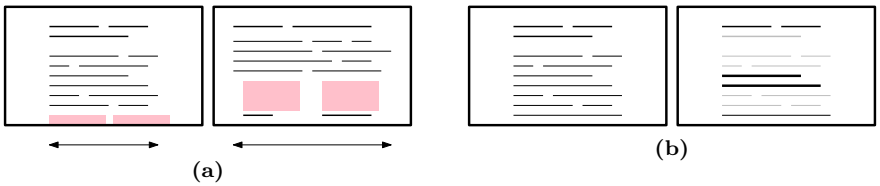


Figure 5.7: Some redesign considerations. [5.7a] Widening the central column of a web page allows the browser to display more information at a glance. [5.7b] Some parts of the UI can be altered according to its importance; e.g., changing the font sizes and colors of headings and text paragraphs.

Design Gallery [Marks et al., 1997], Side Views [Terry and Mynatt, 2002], or Adaptive Ideas [Lee et al., 2010]. However, there is little research toward tools that allow designers to explore design *refinements*, i.e., to get the "design right". Traditionally, current techniques to suggest improvements to an existing design imply a manual implementation (see Section 5.2). What would be interesting, though, is being able to automate the process to a greater or a lesser extent. In this regard, Masson et al. [2010] proposed using interactive genetic algorithms to add permutations to an existing design. The downside of this approach is that it relies on a user-task model and therefore it must be learned. In contrast, we propose to use ACE, which is model-free, and lets *all* users take part in the design process. However, instead of adapting a UI to an individual, the interactions of all users can be exploited to alter the design of an application or a whole website. Among other benefits, this may allow designers to:

1. Avoid having to recruit users for testing each time the application is updated: what you see is what users do.
2. Discover visually what behavioral patterns are consensus.
3. Find inspirational examples, by looking at how the appearance of the UI gets modified over time.

If subtle design modifications are needed to refine an existing layout—as it often happens when iterating over a design solution—then implicit user interaction can be valuable to this end [Leiva, 2012b]. For instance, on websites, if all users spend most of their browsing time on the home page 'above the fold', the designer could consider make wider the main body content, so that some parts could be accessed faster (Figure 5.7a). Similarly, if there is some paragraph that is commonly selected, if would be interesting to make such text more prominent, probably by increasing the font size or the color contrast, so that in subsequent visits users could realize easily where is the popular information (Figure 5.7b).

We believe therefore that ACE can exploit the collective users' behavior as an inspirational source for UI redesign. Implicit interactions can be gathered at scale on a daily basis, and without burdening the user. What is more, on the

Web, independent feedback is received from hundreds or thousands of remote anonymous users rather than being produced and interpreted in a small group or individuals working in isolation. This may help to achieve (hopefully) better design decisions, since it is possible to empirically validate how users react to a particular design update; e.g., by carrying out A/B tests. Additionally, this has the notable advantage that data acquisition and later processing can be both completely automated.

5.5 Evaluation

In terms of system performance, ACE takes a few milliseconds to complete the adaptation process. A series of JavaScript benchmarks were performed on the sample page shown in Figure 5.3 with different configuration objects and CSS properties. The machine was an i686 @ 2 GHz with 1 GB of RAM. The adaptation code was executed 100 times and benchmark results were averaged. Concretely, for 10 items (that were specified by different CSS level 3 selectors[4]) having at most 5 properties each, in all tested browsers (Firefox 7, Chrome 15, Opera 11, Internet Explorer 9, and Dolphin 2.2) the average times were below 20 ms, with standard deviations below 0.1 in all cases.

Regarding human evaluation, devising the most suitable evaluation method is still not completely clear. As a preliminary approximation, an informal study involving 12 users was carried out, in which participants where told to freely browse a mockup site (Figure 5.1) with the ACE system on an HTC Desire [Leiva, 2011]. At the end of the test, users answered three questions (see Figure 5.8); **Q1**: Do you think page elements are well laid out? **Q2**: Did you notice any change on the page, regarding the first time you visited it? **Q3**: If so, did you find distracting those changes?

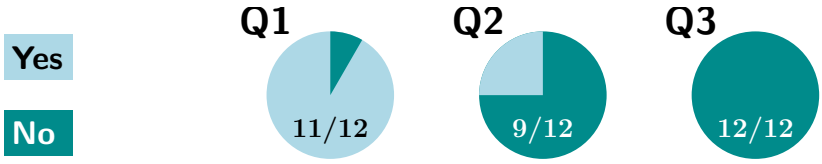


Figure 5.8: Results of the informal user questionnaire.

Overall, users' acceptability toward the method was perceived as positive. As observed, nine of them did not notice the automatic modifications, and none found distracting those changes while browsing. The informal pilot study, although being not conclusive, revealed that this adaptation technique has an interesting potential in building adaptive user interfaces.

[4] http://www.w3.org/TR/css3-selectors/

Regarding using ACE as a source of creative redesign, previous informal meetings with web designers have shown that this tool is perceived as a useful help [Leiva, 2012b]. People commented that they often want to determine how changes to a few page elements will affect the final appearance of the website. ACE satisfies this need, by letting them to inspect how user behavior would influence CSS rendering. Moreover, automatic redesign frees the web designer from the need to know what changes are possible, or how they can be effectively performed. Also, design refinements can offer pragmatic value as well as inspirational value. Figure 5.9 depicts some examples that this tool can produce.

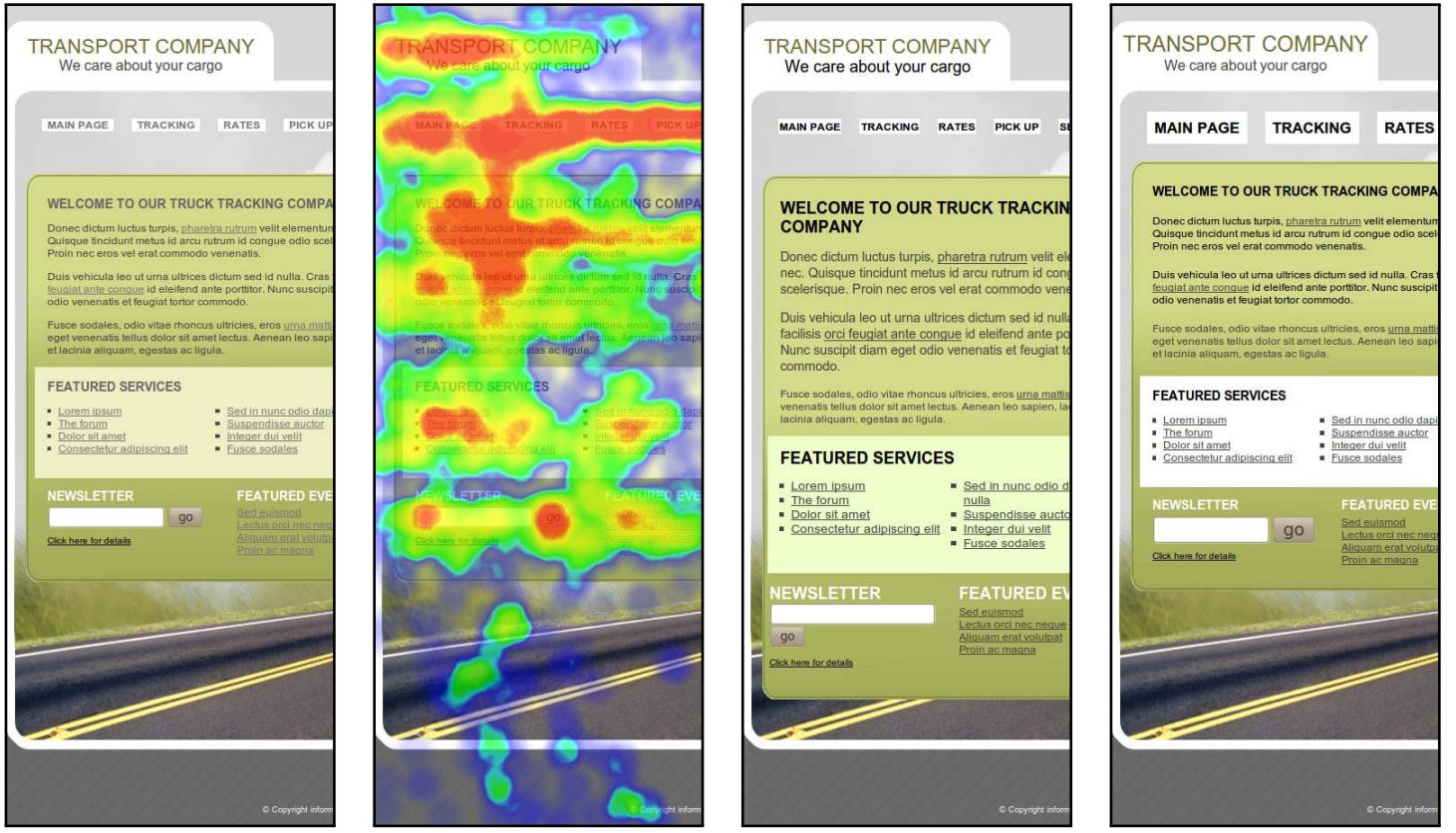
5.6 Discussion

Automatically mining implicit interactions for UI adaptation *and* redesign is a promising direction for future research. However, some work still remains to be done.

First of all, we feel that evaluating this kind of adaptation strategy is quite challenging, since no objective metrics can be consistently computed; e.g., in the absence of labeled samples, we cannot apply well-known measures such as precision and recall; and having to interrupt the normal navigation flow of users to ask them to vote is certainly not an option. We strongly believe, though, that implicit interactions inherently encode performance. Thus, if an adapted design works better than a previous iteration, it should be reflected somehow in the traces of movements, gestures, etc. Nevertheless, one needs to be cautious with this hypothesis, since learnability and familiarity with the UI could be introducing a serious bias. Therefore, an immediate follow-up work will consist in carrying out a formal in-lab evaluation study.

On the other hand, since content is automatically generated, it is likely to be of less quality than human-generated content. Thus, we believe that it would be interesting to assess the influence of such variations in layout design, or use different evaluation viewpoints; e.g., measure the reduction of user effort, compare to other adaptive systems, etc.

ACE has some implications for participatory design as well, since it aims to create applications that are more appropriate to their users. As previously commented, this frees the UI designer from the need to know what changes are possible; but more importantly, it helps to determine *how* such changes can be effectively performed. Also, system suggestions are expected to offer pragmatic value as well as inspirational value to the designer. ACE also can contribute to find "interaction agreements" between all users, which may be useful to detect whether if a design works as expected; e.g., how designs change through time according to the heterogeneous behavior of the users. Additionally, non-experienced designers can gain insights about what is going on with



(a) Original design **(b)** Movements + Clicks **(c)** Redesign suggestion #1 **(d)** Redesign suggestion #2

Figure 5.9: Redesign examples produced by ACE, taking into account multiple interaction logs and overriding a few CSS rules.

their designs, from the user interactions' point of view. This suggests implications for design practices from which the HCI community may well be able to benefit. Finally, collected data can be reused to support design decision making, or to improve understanding of how users interact at scale. Data can also be used for complementary analytics in traditional usability tests, or applied to infer new knowledge for future users.

A known limitation of ACE is that currently it can adapt only those properties that vary in a numerical range; e.g., `max-height` or `padding`. However, in a future it is expected to be able to map semantic properties. For instance, to adapt the `text-align` property of a text paragraph one could use:

$$v = \begin{cases} \texttt{"left"} & \text{if } s \in (-1, -0.5] \\ \texttt{"center"} & \text{if } s \in (-0.5, 0.5) \\ \texttt{"right"} & \text{if } s \in [0.5, 1) \end{cases}$$

Finally, redesign decisions are (by now) based on modifications of shape, position, and/or color attributes. Therefore, more advanced adaptation strategies such as re-arranging several page elements (beyond alignment) or inserting/removing content would require a technically more sophisticated approach.

All in all, this technology enables a straightforward means to invisibly enhance the utility of regular applications and web pages; e.g., in terms of usability, accessibility, readability, interactivity, or performance. Systems like ACE may allow applications to be flexible enough to meet different user needs, preferences, and situations.

5.7 Conclusions and Future Work

Dynamic and continuously changing environments like the Web demand new means of building UIs that are aligned to the skills of the users. We have presented an alternative to redesign interface widgets that operates unobtrusively for both the user and the application structure. Substantial improvements can be made at no cost, since the system is the only responsible of performing the adaptation, being delimited by the (implicit) user interactions and the restrictions imposed by the developer, so that not all events affect all styling.

Finally, we believe that this work opens a door to a wealth of applications that can be developed by tracking the user activity and dynamically restyling the appearance of the UI in response. For instance, integrating ACE with an eye-tracker would provide a finer-grained and potentially more focused analysis of user interactions. Moreover, other biometric inputs such as electrocardiogram signals would allow developers create "organic" UIs that are able to react to the emotions of the users.

Further research will pursue more ambitious results, such as inferring high-level behaviors from low-level events—for instance, reporting if a certain design causes users to get lost or incites them to being more active.

Bibliography of Chapter 5

E. Arroyo, T. Selker, and W. Wei. Usability tool for analysis of web designs using mouse tracks. In *Proceedings of extended abstracts on Human factors in computing systems (CHI EA)*, pp. 484–489, 2006.

R. Atterer, M. Wnuk, and A. Schmidt. 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)*, pp. 203–212, 2006.

N. Bila, T. Ronda, I. Mohomed, K. N. Truong, and E. de Lara. PageTailor: reusable end-user customization for the mobile web. In *Proc. MobySys*, pp. 16–29, 2007.

M. Bolin, M. Webber, P. Rha, T. Wilson, and R. C. Miller. Automation and customization of rendered web pages. In *Proceedings of the 18th annual ACM symposium on User interface software and technology (UIST)*, pp. 163–172, 2005.

M. Claypool, P. Le, M. Wased, and D. Brown. Implicit interest indicators. In *Proceedings of the 6th international conference on Intelligent user interfaces (IUI)*, pp. 33–40, 2001.

L. Findlater and J. McGrenere. Impact of screen size on performance, awareness, and user satisfaction with adaptive graphical user interfaces. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems (CHI)*, pp. 1247–1256, 2008.

K. Z. Gajos and D. S. Weld. SUPPLE: Automatically generating user interfaces. In *Proceedings of the 9th international conference on Intelligent user interfaces (IUI)*, pp. 93–100, 2004.

K. Z. Gajos, J. O. Wobbrock, and D. S. Weld. Automatically generating user interfaces adapted to users' motor and vision capabilities. In *Proceedings of the 20th annual ACM symposium on User interface software and technology (UIST)*, pp. 231–240, 2007.

K. Z. Gajos, K. Everitt, D. S. Tan, M. Czerwinski, and D. S. Weld. Predictability and accuracy in adaptive user interfaces. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems (CHI)*, pp. 1271–1274, 2008.

M. Y. Ivory and M. A. Hearst. Statistical profiles of highly-rated web sites. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pp. 367–374, 2002.

S. Kurniawan, A. King, D. Evans, and P. Blenkhorn. Personalising web page presentation for older people. *Interacting with Computers*, 18(3):457–477, 2006.

B. Lee, S. Srivastava, R. Kumar, R. Brafman, and S. R. Klemmer. Designing with interactive example galleries. In *Proceedings of the 28th international conference on Human factors in computing systems (CHI)*, pp. 2257–2266, 2010.

L. A. Leiva. Restyling website design via touch-based interactions. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI)*, pp. 91–94, 2011.

L. A. Leiva. Interaction-based user interface redesign. In *Proceedings of the 17th international conference on Intelligent User Interfaces (IUI)*, pp. 311–312, 2012a.

L. A. Leiva. Automatic web design refinements based on collective user behavior. In *Proceedings of the 2012 annual conference extended abstracts on Human factors in computing systems (CHI EA)*, pp. 1607–1612, 2012b.

J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber. Design galleries: A general approach to setting parameters for computer graphics and animation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques (SIGGRAPH)*, pp. 389–400, 1997.

D. Masson, A. Demeure, and G. Calvary. Magellan, an evolutionary system to foster user interface design creativity. In *Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems (EICS)*, pp. 87–92, 2010.

M. Terry and E. D. Mynatt. Side views: Persistent, on-demand previews for open-ended tasks. In *Proceedings of the 15th annual ACM symposium on User interface software and technology (UIST)*, pp. 71–80, 2002.

T. Tsandilas and M. C. Schraefel. User-controlled link adaptation. In *Proceedings of the fourteenth ACM conference on Hypertext and hypermedia (HT)*, pp. 152–160, 2003.

D. Zhang. Web content adaptation for mobile handheld devices. *Communications of the ACM*, 50(2):75–79, 2007.

Chapter 6

Interactive Pattern Recognition

Lately, the paradigm for Pattern Recognition (PR) systems design is shifting from the concept of full automation to schemes where the decision process is conditioned by human feedback. This is is motivated by the fact that many applications are expected to assist rather than replace human work; think for instance of systems for medical diagnosis or traffic control.

In this chapter, as an alternative to reviewing (or post-editing) the automatic output of PR systems, an interactive approach is proposed, where the human is placed "in the loop". This scenario leads the system to being able to leverage implicit information from user interactions, and use this information to improve its performance. Interactivity naturally entails multimodal operations, offering opportunities for even greater usability improvements. Multimodality arises when additional feedback signals are non-deterministic and, consequently, need to be decoded. Finally, interactivity offers an ideal framework for adaptive learning, which is expected to lead to further improvements in both performance and usability.

Chapter Outline

6.1 Introduction

Novel interfaces with high cognitive capabilities is a hot research topic that aims at solving challenging application problems in our society of information technology. The outstanding need for the development of such interactive systems is clearly reflected, for instance, in the MIPRCV[1] project, where these cognitive capabilities are included as one of the priority research challenges. Placing Pattern Recognition (PR) within an HCI framework requires changes to the way we look at problems in these areas [Vidal et al., 2007]. Classical PR minimum-error performance criteria should be complemented with better estimations of the amount of effort that the interactive process will demand from the user. As such, current existing PR techniques, which are intrinsically grounded on error-minimization algorithms, need to be revised and adapted to the new, minimum human-effort performance criterion.

Mining implicit data from user interactions provides research with a series of challenges and opportunities in order to rethink how Interactive PR approaches (IPR for short) may drive the dynamic environment of interactive systems. In this context, implicit interaction entails three types of opportunities in IPR:

- Feedback information derived from the interaction process can be used to significantly improve system performance.
- Interaction feedback signals are intrinsically multimodal, which means that we can study the synergy among different input modalities to enhance overall system behavior and usability.
- Each interaction generally yields ground-truth data, which can be advantageously used as valuable adaptive training data and tune system performance.

It should be noted that multimodal interaction may support two types of multimodality [Toselli et al., 2011]. One corresponds to the input signal itself, which can be a complex mixture of different data types, ranging, e.g., from conventional keystrokes to audio and video data streams. The other type, more subtle but also important, is derived from the often different nature of input and feedback signals. It is this second type the one that makes both multimodality and implicit interaction an inherent feature of human behavior.

Overall, the IPR framework proposes a radically different approach to correct the errors committed by a PR system. This approach is characterized by human and machine being tied up in a much closer loop than usually. That is, the user gets involved not only after the system has completed the production of its final recognition result, but also during the recognition process

[1] http://miprcv.iti.upv.es

itself. This way, errors can be avoided beforehand and correction costs can be dramatically reduced. Historically, this interactive-predictive approach was proposed by the so-called *conversation theory* from cybernetics, in which the system constructs its knowledge by means of a series of user interactions [Pask, 1975]. Currently, the Machine Learning community has renamed this approach to *corrective feedback* [Culotta et al., 2006], since every time the user amends an error, the system reacts by modifying the resulting hypothesis.

6.1.1 IPR Framework Overview

The IPR framework (Figure 6.1) is explained as follows [Vidal et al., 2007]:

- $\mathcal{X}$ is the system's input domain; i.e., the domain where input stimuli, observations, signals, or data come from.
- $\mathcal{H}$ is a theoretically infinite set of possible system outputs, results, or hypotheses. $h \in \mathcal{H}$ is a hypothesis which the system derives from a certain input $x \in \mathcal{X}$.
- $\mathcal{F}$ is the domain were feedback signals come from. $f(h, x)$, or just $f \in \mathcal{F}$ is a specific feedback signal which the user provides as a response to the system hypothesis h.
- $\mathcal{M}$ is any model which the system uses to derive its hypotheses.

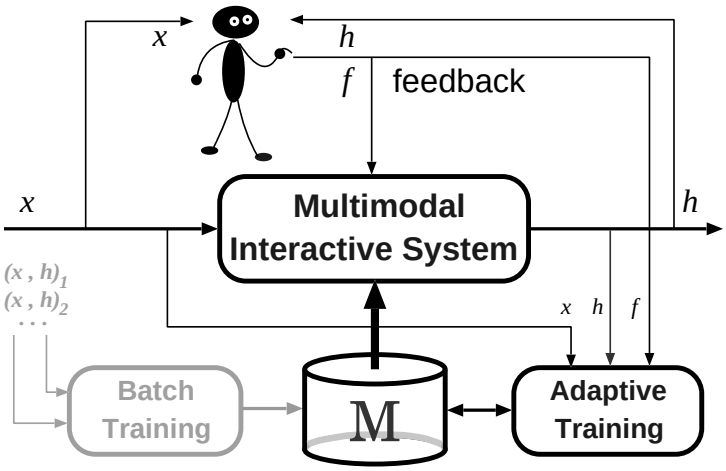


Figure 6.1: The IPR framework [Vidal et al., 2007]. Reproduced with permission.

Assume for simplicity that both the input x and the feedback f are unimodal. Interaction leads to the following modality fusion problem[2]:

$$\hat{h} = \arg\max_h \Pr(h|x,f) = \arg\max_h \Pr(x,f|h) \cdot \Pr(h) \tag{6.1}$$

In many applications x and f can be assumed to be independent given h. This allows for a naïve Bayes decomposition:

$$\hat{h} \approx \arg\max_h P_{\mathcal{M}_\mathcal{X}}(x|h) \cdot P_{\mathcal{M}_\mathcal{F}}(f|h) \cdot P_{\mathcal{M}_\mathcal{H}}(h) \tag{6.2}$$

Then, independent models, $\mathcal{M}_\mathcal{X}$, $\mathcal{M}_\mathcal{F}$ and $\mathcal{M}_\mathcal{H}$, can now be estimated separately for the input components and for the prior hypotheses distribution, respectively. This way, the resulting search problem accounts for the joint optimization of the conditional probability product.

6.1.2 Interaction Protocol

In the context of this thesis, the IPR framework has been successfully applied to four different IPR systems, where implicit interaction plays a crucial role: *1*) Handwritten Transcription, *2*) Machine Translation, *3*) Grammatical Parsing, and *4*) Image Retrieval. Indeed, the role of implicit interaction is crucial because the user can interact with an IPR system in an unimaginable number of ways. As such, the range of interaction possibilities has to be delimited or predicted in some way, so that the system can take maximum advantage of the expected user feedback. This leads to the creation of a user model, also known as an *interaction protocol*.

Depending on the application and the input modalities involved, very different types of protocols can be assumed for the user to interact with the system in a comfortable and productive way. But the chosen protocols must also allow an efficient implementation, because interactive processing is generally highly demanding in terms of response times [Toselli et al., 2011]. Eventually, the design of an efficient interaction protocol *and* an adequate UI are the most sensible design tasks for an IPR application. Concretely, once a specific interaction protocol is defined, it should be possible to apply decision theory in order to model the expected interaction effort of such protocol in terms of an adequate loss function. This would allow to search for a corresponding decision function that minimizes the loss; i.e., the expected interaction effort.

Within the two general types of interaction protocols identified in IPR [Toselli et al., 2011], we will focus in the *passive protocol*, that is, where the system requires human feedback to emit a hypothesis. This focus is motivated by the

[2]True probabilities are denoted as $\Pr(\cdot)$, while $P_\mathcal{M}(\cdot)$ or just $P(\cdot)$ denote probabilities computed with some model $\mathcal{M}$.

fact that it is a suitable scenario in which the system can take advantage of implicit interactions to a great extent. In contrast, under the *active protocol* it is the system, rather than the human, which is in charge of making the relevant decisions about the need of supervising errors. Clearly, this scenario is not as advantageous as the previous one to illustrate the role of implicit interactions in IPR.

In general, the way of interacting with an IPR system following a passive protocol is described as follows:

1. The system automatically proposes a draft of the output of the task; e.g., a text transcription or a collection of images.
2. The user then validates the parts of the output which is error-free; e.g., indicating the correct prefix in a text-oriented task or selecting those images considered as relevant in image retrieval.
3. The system then suggests a suitable, new extended consolidated hypothesis based on the previously validated parts and implicit information derived from user feedback.
4. Steps 2 and 3 are iterated until a final, perfect output is produced.

In the following sections we delineate a series of real-world implementations of the MIPR framework.

6.2 IPR Systems Overview

The following prototypes are focused on an interactive-predictive strategy, fully integrating the user knowledge into the PR process. The prototypes have been classified into two categories, depending whether the user feedback comes in the form of *structured input* or not. The former category includes three examples of Natural Language Processing (NLP) systems, where the order in which errors are corrected is determinant for the system. The latter category includes as an example an image retrieval system, where the user feedback comes in the form of *desultory input*, i.e., the order is not determinant for the system.

It is worth pointing out that these prototypes were not intended to be production-ready applications. Rather, they were developed to provide an intuitive interface which aims at showing the functionality of an IPR system in general, as well as illustrating the role of implicit interaction in particular.

6.2.1 Structured Input

In these systems, the user validates the longest prefix of the system hypothesis (e.g., a text transcription, a speech utterance, etc.) which is error-free. Such a

validation can be performed by using, e.g., a keyboard, a computer mouse, a touchscreen, a microphone, or an e-pen. Once the first error is corrected, the system predicts the most probable continuation of the partial input. This new extended hypothesis is strongly based on the previously validated prefix and the decoding of the corrections submitted by the user—for instance, if an e-pen was used to write down a word, those pen strokes must be decoded. For the sake of simplicity, let us assume in this subsection that the system is producing text-based hypotheses; for instance, transcriptions, translations, or parse trees.

As observed, under this protocol, the user is asked to correct the first error found. Then, the system can make the reasonable assumption that the user is reading the text form left to right (or vice versa for right-to-left languages, such as Arabic). With this assumption, the search process of the next (best) hypothesis is constrained to a smaller subset of words regarding the initial hypothesis, which allows the system to make a better prediction. Moreover, this assumption allows to automate the evaluation of these IPR systems, by simulating a user that will perform a series of error amendments in an ordered sequence.

However, the role of implicit HCI has much to offer to this protocol, as the system can place a series of (safe) constraints to improve its hypotheses even further. For instance, some editing operations are expected to be performed by the user beyond simple word substitution, e.g., insertion, deletion, or rejection (Figure 6.2). More specifically, when the user is going to insert (or delete) a word, the system can assume that the word at the right of the insertion (or deletion) is correct. This constrains to an even smaller subset of words regarding the previous hypothesis, and therefore it is expected that the next prediction will be much better, since the system has more information that is implicitly validated. Going further, to replace an incorrect word the user needs to place the cursor over a text field and then start typing the corrected word. Nevertheless, this information about cursor placement can be leveraged to emit the next hypothesis *before* the user starts typing, offering thus a (hopefully) better proposal, if not the one the user had in mind.

(a) Delete **(b)** Insert **(c)** Reject

Figure 6.2: Examples of editing operations in IPR systems. By deleting or inserting a word, the system can assume that the neighboring words are implicitly correct, allowing thus for a better prediction in the next hypothesis. By making a rejection, though, the system can only assume that the word at the left is correct.

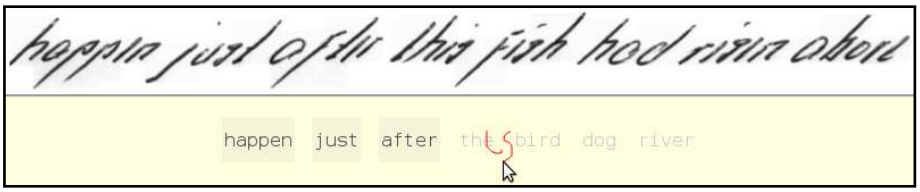


Figure 6.3: Interactive Handwritten Transcription prototype, an example of structured input. Some word-level editing operations that can be performed are substitution (shown in the image), insertion (6.2a), deletion (6.2b), or rejection (6.2c). In this example, the system assumes that the first 3 words plus the first 2 characters of the edited word are correct. This information is used to 1) decode the submitted pen strokes and 2) predict a suitable continuation of the implicitly validated segment: "happen just after this fish ···". Prototype available at http://cat.iti.upv.es/iht/.

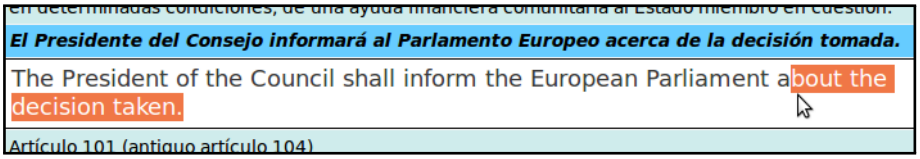


Figure 6.4: Interactive Machine Translation prototype, another example of structured input. Here, the system predicts a new hypothesis when the cursor is positioned over an erroneous character, before the user starts typing. As in Figure 6.3, the text at the left of the cursor is considered to be correct. Available at http://cat.iti.upv.es/imt/.

6.2.2 Desultory Input

As in the previous passive protocol, here the user is expected to supervise the system hypotheses in order to achieve a high-quality result. However, in this case the user can perform the amendments in a desultory order. This is especially useful when the elements of the output do not have a particular hierarchy. Many different scenarios can fall under this category. However, here we analyze the case of information retrieval, where the user initially submits a natural language description of an object she is looking for.

Under this protocol, the system outputs a set of objects matching the submitted query, so the user can select which ones fit her needs and which do not. The system then tries to fill the set with new objects taking into account the user preferences from the previous iterations. The procedure stops when the user chooses not to reject any further object from the set. The goal is to obtain such a set in the minimum number of interactions.

In image retrieval this protocol is known as *relevance feedback*, since the user typically categorizes the presented images into two (sometimes three) classes: relevant and non-relevant (and neutral in some cases). The role of implicit interaction in this scenario is particularly useful to unburden the user from having to think whether a particular image should be classified as non-relevant or neutral. As such, it is much easier for the user just to indicate which images

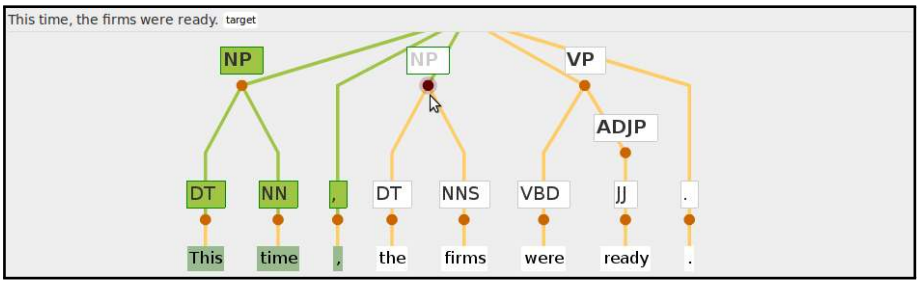


Figure 6.5: Interactive Grammatical Parsing prototype, an example of two-dimensional structured input. The same editing operations presented in Figure 6.2 can be performed. Tree visiting order is left-to-right depth-first, so resulting nodes at the left of (and above) the cursor are considered to be correct. Available at `http://cat.iti.upv.es/ipp/`.

are relevant. The system then classifies the rest of presented images into non-relevant, e.g., if they are very different to the ones the user has selected, or into neutral otherwise. Moreover, this strategy allows to automate the evaluation of these image retrieval systems, by simulating a user that will select only images considered as relevant in each iteration with the system.

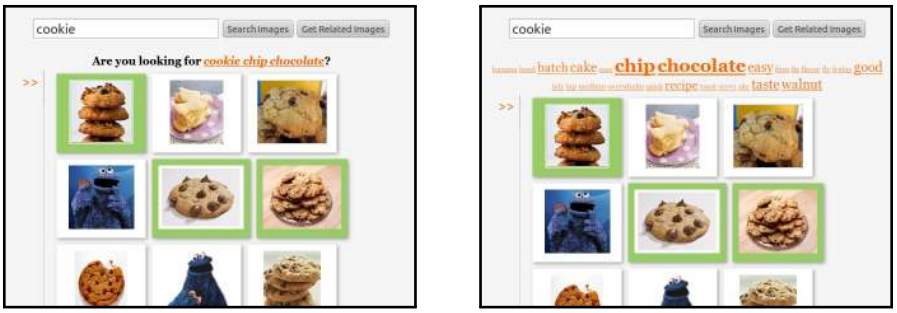
Again, the role of implicit HCI has much to offer to this protocol, as the system can take some initiative derived from user input to improve its hypotheses even further. For instance, using metadata from the presented images, it is possible to suggest a textual query that would allow the user to retrieve better images from scratch. In addition, the system can present a tag cloud to provide the user with a gist of the current set of images. Furthermore, when clicking on a tag, the system can refine the original query by adding the respective tag (or related information thereof) to the query.

6.3 Evaluation

Here we will focus on the evaluation of the IPR framework with real users. The IPR literature uses test-set-based estimates of user effort reduction, but only a few researchers have conducted controlled lab studies to verify whether the IPR framework proves to be superior to current baselines techniques [Alabau et al., 2012; Leiva et al., 2011a,b]. From the four applications previously examined, we will focus on three of them, which are the most mature technologies implemented so far.

6.3.1 Interactive Handwritten Transcription

The goal of this evaluation was aimed at improving Handwritten Text Recognition (HTR) technology. An Interactive Handwriting Transcription (IHT) system was used on a real-world task, and compared to a manual approach as



(a) Query refinement suggestion **(b)** TagCloud based suggestion

Figure 6.6: Interactive Image Retrieval prototype, an example of desultory input. The user must select which images are relevant, in no particular order, and the system will mark the rest as non-relevant or neutral; depending on the considered image features. Moreover, this information can be used to make suggestions to the user, e.g., as a refined query (6.6a) or as a tag cloud (6.6b), which may help to disambiguate intent or to retrieve hopefully better images. Available at `http://risenet.iti.upv.es/`.

baseline. We compiled a test corpus from a 19th century handwritten document identified as "Cristo Salvador" (CS), which was kindly provided by the Biblioteca Valenciana Digital[3].

Participants Fourteen users from our Computer Science department volunteered to cooperate, aged 28 to 61 (M=37.3). Most of them were knowledgeable with handwriting transcription tasks, although none was a transcriber expert. One user could not finish the evaluation, so the end user sample was 13 subjects (3 females).

Assessment Measures We used two well-known objective test-set-based measures: word error rate (WER)[4] and word stroke ratio (WSR)[5], both normalized by the number of words in the reference transcription. We also measured the time needed to transcribe completely each page with each HTR system. Additionally, we measured the *probability of improvement* (POI), which estimates if a system is *a priori* better than another for a given user [Bisani and Ney, 2004].

Design We carried out a within-subjects repeated measures design. We tested two conditions: transcribing a page with the manual and the IHT system, taking into account that each one was tested twice—to compensate the above-mentioned learnability bias. We used the (non-parametric) two-sample Kolmogorov-Smirnov test, since normality assumptions did not hold.

[3] `http://bv2.gva.es/`
[4] WER is the minimum number of editing operations to achieve the target transcription.
[5] WSR is the number of interactions needed to achieve the target transcription.

Apparatus We modified an IHT web-based prototype [Romero et al., 2009] to carry out the field study. We implemented two HTR engines to assist the document transcription on the same UI. In addition, a logging mechanism was embedded into the web application. It allowed us to register all user interactions at a fine-grained level of detail (e.g., keyboard and mouse events, client/server messages exchanging, etc.). Then, interaction log files were reported in XML format for later postprocessing.

Procedure Participants accessed the web-based application via a special URL that was sent to them by email. In order to familiarize with the UI, users informally tested each transcription engine with some test pages, different from the ones reserved for the real test. Then, people transcribed the two user-test pages with both transcription engines. These pages were selected according to their WER and WSR values, which were close to the median values of the test-set. To avoid possible biases due to human learnability, the first page (#45) was initially transcribed with the manual engine first; then the order was inverted for the second page (#46). Finally, participants filled out an online System Usability Scale (SUS) questionnaire [Brooke, 1996] for both systems. Such an online form included a text field to allow users submit free comments and ideas about their testing experience, as well as insights about possible enhancements and/or related applicability.

Results

In sum, we can assert that regarding *effectiveness* there are no significant differences, as expected, i.e., users can achieve their goals with any of the tested systems. However, in terms of *efficiency* the IHT system is the better choice. Regarding to *user satisfaction*, IHT again seems to be the most preferable option.

Quantitative Analysis Table 6.1 summarizes the main findings. We must emphasize that the daily use of any system designed to assist handwriting transcription would involve not having seen previously any of the pages (i.e., users would usually read a page once and at the same time they would transcribe it). Therefore, IHT seems to be slightly better than a manual approach in terms of WER, and clearly superior in terms of WSR.

Analysis of Task Completion Time We observed that, overall, there are no differences in transcription times [$D = 0.16$, $p = .75$, n.s.]. In general, the system used in second place always achieved the best time, because the user already knew the text. The remarkable result is that when the user reads a page in first place the chosen engine is not determinant, because one must spend time to accustom to the writing style, interpreting the calligraphy, etc. In this case the POI of IHT with respect to the manual engine is 53%.

System		Time	WER	WSR
Overall	Manual	11.1 (3.5)	8.6 (8.2)	97.8 (6.0)
	IHT	10.3 (3.7)	6.5 (3.7)	30.4 (6.1)
Page 45	Manual	12.8 (3.5)	12.8 (9.5)	97.3 (7.0)
	IHT	8.6 (3.2)	7.0 (4.1)	28.6 (4.1)
Page 46	Manual	9.4 (2.9)	4.1 (2.0)	98.4 (4.6)
	IHT	12.0 (3.4)	6.0 (3.3)	32.1 (7.1)

Table 6.1: Mean (and SD) per page for the measured variables: time (in minutes), WER (in %), and WSR (%).

Analysis of WER Overall, IHT performs better regarding WER [$D = 0.11$, $p = .99$, n.s.]. Although the differences are not statistically significant, the interesting observation is that IHT is the most stable of the systems—even better than when using the manual engine on an already read page. We must recall that the more stable in WER a system is, the fewer residual errors are expected and therefore a high quality transcription is guaranteed. In this case, considering the first time that the user reads a page, the POI of the IHT engine over the manual engine is 69%.

Analysis of WSR Interestingly, the WSR when using the manual engine was below 100%, since there are inherent errors (some users were unable to read all lines correctly). This means that some users wrote less words in their final transcriptions than they really should have to when using the manual engine. In both conditions IHT was the best performer, and differences were found to be statistically significant [$D = 1, p < .001$]. The POI of the IHT engine regarding the manual engine is 100%. This means that the number of words a user must write and/or correct under the IHT paradigm is always much lower than with a manual system. Additionally, this fact increases the probability of achieving a high-quality final transcription, since users perform fewer interactions and are prone thus to less errors.

Qualitative Analysis Regarding user subjectivity, the SUS scores could be considered normally distributed. Thus, a Welch two-sample t-test was employed to measure the differences between both groups. We observed a tendency in favor to IHT [$t(22) = 0.25$, $p = .80$, n.s.], since users generally appreciate the guidance of the IHT system to suggest partial predictions, considering the difficulty of the task proposed in the field study.

Limitations of the Study First, taking into account that our participants were not experts in transcribing ancient documents, a dispersed behavior was expected (i.e., some users were considerably faster/slower than others). Second, the pages were really deteriorated, making more difficult the reading for the users. For that reason, there is a strong difference between the first time that

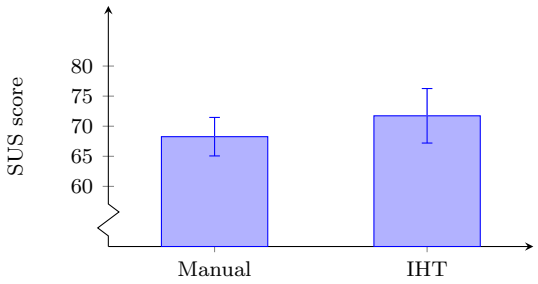


Figure 6.7: User satisfaction, according to the SUS questionnaire. Error bars denote 95% confidence intervals.

a user had to transcribe a page and subsequent attempts. Third, most of the participants had never faced neither any of the implemented engines nor the web UI before the study, so it is expected a learning curve prior to using such systems in a daily basis. Finally, a simplified starting level would minimize this effect for the task; however we tried to select a scenario as close as possible to a realistic setting.

Evaluation Discussion

Despite of the above mentioned limitations, there is a comprehensible tendency to choose the IHT paradigm over the manual system. Additionally, as observed, the probability of improvement of an IHT engine over manual transcription revealed that the interactive-predictive paradigm worked better for all users.

The advantage of IHT over traditional HTR post-editing approaches goes beyond the good estimates of human effort reductions achieved. When difficult transcription tasks with high post-editing effort are considered, expert users generally refuse to post-edit conventional HTR output [Toselli et al., 2009]. In contrast, the proposed interactive approach constitutes a much more natural way of producing correct texts. With an adequate user interface, IHT lets the user be dynamically in command: if predictions are not good enough, then the user simply keeps typing at her own pace; otherwise, she can accept (partial) predictions, thereby saving both thinking and typing effort.

6.3.2 Interactive Machine Translation

The goal of this evaluation was aimed to assess a Machine Translation (MT) system that is based on the IPR framework (IMT for short), and compare it to a state-of-the-art post-editing (PE) MT system. Translating manually from scratch was not considered, since this practice is being increasingly displaced by assistive technologies at present. Indeed, PE of MT systems is found frequently in a professional translation workflow [TT2, 2001].

Initially, we modified an IMT web-based prototype [Ortiz-Martínez et al., 2010] to carry out the evaluation. We targeted specific IMT features, e.g., confidence

measures in translated words or click-based operations. We will refer to this system as the advanced version (IMT-AV).

Evaluation of an Advanced Version

In addition to IMT-AV, a post-editing version of the prototype (PE-AV) was developed to make a fair comparison with state-of-the-art PE systems. PE-AV used the same interface as IMT-AV, but the IMT engine was replaced by autocompletion-only capabilities, as it is found in popular text editors.

Participants A group of 10 users (3 females) aged 26–43 from our research group volunteered to perform the evaluation as non-professional translators. They were proficient in Spanish and had an advanced knowledge of English. While none of them had worked with IMT systems before, all knew the theoretical foundations of the technology.

Assessment Measures Both systems were evaluated on the basis of the ISO 9241-11 standard (ergonomics of human-computer interaction). Three aspects were considered: efficiency, effectiveness, and user satisfaction. For the former, we computed the average time in seconds that took to complete each translation. For the second, we evaluated the BLEU[6] against the reference and a crossed multi-BLEU among users' translations. For the latter, we formulated 10 questions inspired by the system usability scale (SUS) questionnaire. Users would answer the questions in a 5-point Likert scale (1: strongly disagree, 5: strongly agree), plus a text area to submit free-form comments.

Apparatus Since participants were Spanish natives, we decided to perform translations from English to Spanish. We chose a medium-sized corpus, the EU corpus, typically used in IMT [Barrachina et al., 2009], which consists of around 200K sentences from legal documents. We built a glossary for each source word by using the 5-best target words from a word-based translation model. We expected this would cover the lack of knowledge for our non-expert translators toward this particular task. In addition, a set of 9 keyboard shortcuts was designed, aiming to simulate a real translation scenario.

Furthermore, autocompletion was added to PE-AV, i.e., words with more than 3 characters were autocompleted using a task-dependent word list. In addition, IMT-AV was set up to predict at character level interactions. We disabled complementary features for the evaluation to focus on basic IMT.

Procedure Three disjoint sentence sets (C1, C2, C3) were randomly selected from the test dataset. Each set consisted of 20 sentence pairs and kept the sequentiality of the original text. Sentences longer that 40 words were discarded. C3 was used in a warm up session, where users gained experience with the IMT system (5–10 min per user on average) before carrying out the actual

[6]BLEU is a standard measure of the quality of machine-translated text.

	PE-AV	IMT-AV
Avg. time (s)	62 (SD=51)	67 (SD=65)
BLEU	40.7 (13.4)	41.5 (13.5)
Crossed BLEU	77.4 (4.5)	78.9 (4.8)
User Satisfaction	2.5 (1.2)	2.1 (1.2)

Table 6.2: Summary of the results for the first test.

evaluation. Then, C1 and C2 were evaluated by two user groups (G1, G2) in a counterbalanced fashion: G1 evaluated C1 on PE-AV and C2 on IMT-AV, while G2 did C1 on IMT-AV and C2 in PE-AV.

Results Although results were not strongly conclusive (there were no statistical differences between groups), some trends were observed. First, time spent per sentence (efficiency) on average in IMT was higher than in PE (67 vs. 62 s). However, effectiveness was slightly higher for IMT in BLEU with respect to the reference sentence (41.5 vs. 40.7) and with respect to a cross-validation with other user translations (78.9 vs. 77.4). This suggested that the IMT system helped to achieve more consistent and standardized translations.

Finally, users perceived the PE system to be more adequate than the IMT system, although global scores were 2.5 for PE and 2.1 for IMT, which suggested that users were not very comfortable with none of the systems (Likert scores were comprised between 1 and 5). IMT failed to succeed in questions regarding the system being easy to use, consistent, and reliable. This was corroborated by the submitted comments.

Users complained about having too many shortcuts and available edit operations, some operations not working as expected, and some annoying common mistakes regarding predictions of the IMT engine (e.g., inserting a whitespace instead of completing a word, which would be interpreted as two different words by the UI). One user stated that the PE system "was much better than the [IMT] predictive tool". Regarding PE, users mainly questioned the usefulness of the autocompletion feature.

Evaluation of a Simplified Version

Results from the first evaluation were quite disappointing. Not only participants took more time to complete the evaluation with IMT-AV, but they also perceived that IMT-AV was more cumbersome and unreliable than PE-AV. However, we still observed that IMT-AV had been occasionally beneficial, and probably the bloated UI was the cause for IMT to fail. Thus, we developed a simplified version of the original prototype (IMT-SV).

Participants Fifteen participants aged 23–34 from university English courses (levels B2 and C1 from the Common European Framework of Reference for Languages) were paid to perform the evaluation (5 euro each). A special price of 20 euro was given to the participant who would contribute with the most useful comments about both prototypes. It was found that, by following this method, participants were more verbose when it came to reporting feedback.

Apparatus In this case, the editing interface was presented as a simple text area. In addition, the editing operations were simplified to allow only word substitutions and single-click rejections. Besides, we expected that the simplification of the interface logic would reduce some of the programming bugs that bothered users in the first evaluation. The PE interface was simplified the same way (PE-SV). Furthermore, the autocompletion feature was improved to support n-grams of arbitrary length. A different set of sentences (C1′, C2′, C3′) was randomly extracted from the EU corpus.

Procedure To avoid possible bias regarding which system was being used, sentences were presented in random order, and engine type was hidden to participants. As a consequence, users could not evaluate each system independently. Therefore, a reduced questionnaire with just two questions was shown on a per-sentence basis. **Q1** asked if system suggestions were useful. **Q2** asked if the system was cumbersome to use overall. A text area for submitting free-form comments was also included in the UI.

Results Still with no statistical significance, we found that IMT was perceived now better than PE. First, interacting with IMT-SV was more efficient than with PE-SV on average (55 s vs. 69 s). The number of interactions was also lower (79 vs. 94). Concerning user satisfaction, IMT-SV was perceived as more helpful (3.5 vs. 3.1) but also slightly more cumbersome (3.1 vs. 2.9). However, in this case differences were narrower. On the other hand, IMT-SV received 16 positive comments whereas PE received only 5. Regarding negative comments, IMT-SV accounted for 35 items and PE-SV 31 items. While the number of negative comments is similar, there was an important difference regarding positive ones. Finally, user complaints of IMT-SV can be summarized in the following items: *a)* system suggestions changed too often, offering very different solutions on each keystroke; *b)* while correcting one mistake, subsequent words that were correct were changed by a worse suggestion; *c)* system suggestions did not keep gender, number, and tense concordance; *d)* if the user goes back in the sentence and performs a correction, some parts of the sentence already corrected were not preserved on subsequent system suggestions.

Evaluation Discussion

Our initial UI performed poorly when tested with real users. However, when the UI design was adapted to user expectations, results were encouraging. Note that in both cases the same IMT engine was evaluated under the hood. This

	PE-SV	IMT-SV
Avg. time (s)	69 (SD=42)	55 (SD=37)
No. interactions	94 (60)	79 (55)
Q1 (Likert scale)	3.1 (1.2)	3.5 (1.1)
Q2 (Likert scale)	2.9 (1.2)	3.1 (1.3)

Table 6.3: Summary of the results for the second test.

fact remarks the importance of an adequate UI design when evaluating a highly interactive system as IMT.

In sum, the following issues should be addressed in IMT: *1)* user corrections should not be modified, since that causes frustration; *2)* system hypotheses should not change dramatically between interactions, in order to avoid confusing the user; *3)* the system should produce a new hypothesis only when it is sure that it improves the previous one.

6.3.3 Interactive Image Retrieval

In this scenario, it is desirable to retrieve as much precise images as possible in a few feedback iterations. To this end, Paredes et al. [2008] demonstrated that implicitly validating non-selected images as non-relevant is a safe and convenient assumption. Experiments on the well-known Corel/Wang dataset revealed that this method was able to retrieve 94.5% of the relevant images in just 2 iterations.

However, in an image retrieval system, there are generally available many different types of image features, and also there are textual features, such as metadata, annotations provided by users, or text surrounding the images from where they appear. Adequately leveraging all this available information is a major goal in order to obtain the best performance possible. In this section we study two approaches to achieve this goal: *1*) how to combine textual and visual information by using relevance feedback, and *2*) how to present this information to the user in a way that it may improve retrieval results.

Evaluation of Multimodal Fusion

We opted for *late fusion* as a fusion method of visual and textual features, since it is simple and easy to integrate in our previously developed prototype, a Relevant Image Search Engine (RISE) [Segarra et al., 2011]. Since only two modalities are considered, an $\alpha \in [0, 1]$ parameter is set to assign an importance weight to visual image descriptors. This allowed us to implement a linear combination of both features and let the system decide the best ranking of images according to:

$$R_\alpha(x) = \alpha R_v + (1 - \alpha) R_t \tag{6.3}$$

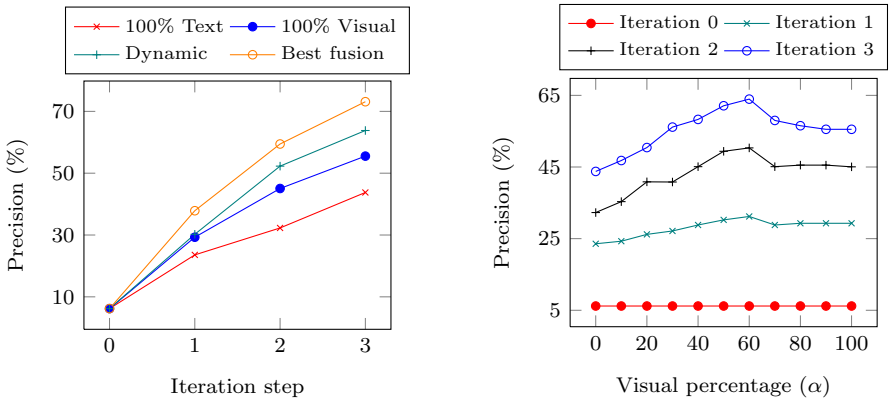
This way, when $\alpha = 0$, only textual features are considered (i.e., textual modality, R_t); while $\alpha = 1$ means that only visual features are considered (i.e., visual modality, R_v). Clearly, α should not be kept fixed for a given system, since it is known that in general some queries will perform better with visual information, or the other way around, and leaving this task to the user is too much burden. Hence, to deal with this dynamically variable weighting, we propose to take advantage of information derived from relevance feedback and solve an optimization problem: the system will try to rank relevant and non-relevant images as far as possible, also placing the relevant images in the top positions. We named this approach *dynamic linear fusion*.

To evaluate this approach, we manually labeled a subset of 21 queries with 200 images each from the RISE image database [Villegas and Paredes, 2012]. The reader may consult [Toselli et al., 2011] for a brief description of each query, together with their respective images.

Instead of recruiting users, as usual, we decided to do a preliminary evaluation first, which would eventually lead to a lab study in case results were promising. For consistency with the default RISE UI (Figure 6.6), we simulated a user who wants to retrieve $N = 10$ images, which were shown at a time. So, in each iteration, the user would see 10 images and judge which were relevant. Visual features were comprised of color histograms, while textual features were comprised of automatic image annotations (extracted from the web pages where images were located). Results are shown in Figure 6.8.

Evaluation Discussion Figure 6.8a shows the evolution of retrieval accuracy with the successive interaction steps for different retrieval strategies. As we suspected, both pure text and visual retrieval alone are worse performers. After one interaction step, the dynamic linear fusion approach performs better on average. The best fusion combination is just an upper bound, and therefore in practice it is unreachable.

It can be observed in Figure 6.8b that the system quickly gains accuracy with the progression of user interaction steps. That is, the more the information known about what is considered relevant in previous steps, the better it can predict the best fusion parameter α for the current step. In the first step, there is a clearly ascendant slope toward the visual strategy, achieving high precision when full visual search is used. However, in the following iterations the best precision is not obtained on the extremes, which shows the importance of having a dynamic user/query-adaptative α to achieve always the best precision.



(a) Precision vs. Iterations

(b) Precision vs. Fusion

Figure 6.8: Dynamic linear fusion results, for $N = 10$ images to be seen at a time. [6.8a] Comparison of image retrieval techniques. [6.8b] Precision as a function of α (visual percentage), for several feedback iteration steps.

Query Refinement Evaluation

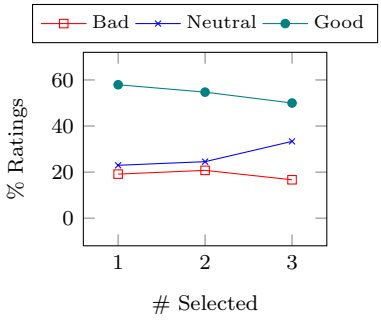
The image database used in RISE prototype was built from real data gathered from the Internet with completely unsupervised annotations, so there is no ground truth available, i.e., labeled samples. Furthermore, labeling a subset of the images in order to evaluate query refinement suggestions is rather challenging. The labeling would require to have a list of sample queries, and for each query, several subsets of selected relevant images corresponding to different subclasses of the original query. Moreover, for each of these subsets we would require a list of possibly correct query refinements. Thus, in order to evaluate the proposed approach, we opted to conduct an informal field study. The procedure was simple: to measure the user's subjectivity toward the query suggestion technique.

For the evaluation, we selected 81 out of the 99 concepts from the ImageCLEF 2011 dataset[7], and used these as the initial text search queries. The reason to remove 18 concepts was because they were related to specific image properties rather than high-level concepts, e.g., "Neutral Illumination", "No Blur", etc.

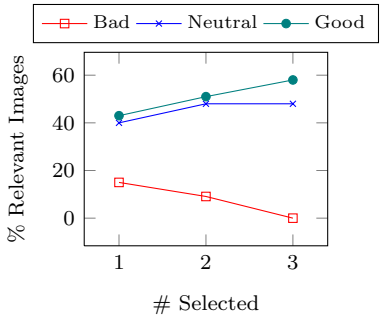
The evaluation task consisted of two stages [Leiva et al., 2011b]. First, users were presented with the first 10 ranked images for a given text query, e.g., "cat". Then the user would select a subset of images which had a common concept or relation among them, e.g., "all are black cats". If the system was able to derive a query refinement, the UI would show it and let the user rate whether the suggestion was either good, bad, or neutral. The number of times there was no query suggested (NQ) was also recorded (Table 6.4). In the second

[7] http://imageclef.org/system/files/concepts_2011.txt

stage of the evaluation, users were presented with the images after following the query suggestion, and they had to mark all of the images considered relevant to the concept they had in mind when selecting the images in the first stage of the evaluation. This two-stage process was repeated for all subsets of related images the user could identify. Three people from our department took part in the evaluation. Results are presented in Figure 6.9 and Tables 6.4 and 6.5, respectively.



(a) Stage 1: query ratings.



(b) Stage 2: retrieved images.

Figure 6.9: Query refinement evaluation results. [6.9a] Average rating of suggested queries against number of initially selected images. [6.9b] Percentage of images considered as relevant after following the query suggestions against number of initially selected images.

# selected	# samples	Bad	Neutral	Good	NQ
1	194	35	42	106	11
2	74	11	13	29	21
3	24	2	4	6	12
>3	30	0	0	3	27
Overall	322	48	59	144	71

Table 6.4: Results for stage 1 of query refinement evaluation, showing absolute ratings for suggested query refinements.

Evaluation Discussion Regarding the first stage of the evaluation, the first thing to note is that, as more images are selected, it is less probable that the system will suggest a query (see Table 6.4). This is understandable, since it is less likely that there will be common terms to all selected images. Moreover, terms associated to each image completely depend on the web pages where the image appears, thus not all images will be well annotated. Nonetheless, most of the suggested queries were rated as being good, which indicates that this approach of deriving suggestions based on selected (relevant) images can be quite useful.

# selected	# ratings	Bad	Neutral	Good
1	183	1.5 (1.5)	4 (3)	4.3 (3)
2	53	0.9 (1.4)	4.8 (3.2)	5.1 (3.3)
3	12	0 (0)	4.8 (4.8)	5.8 (2.8)
>3	3	0 (0)	0 (0)	9.6 (0.5)
Overall	226	1.3 (1.5)	4.3 (3.2)	4.7 (3.2)

Table 6.5: Results for stage 2 of query refinement evaluation, showing mean (and standard deviation) values of the number of relevant images retrieved after following suggested queries.

Regarding the second stage of the evaluation, as expected, query suggestions which were rated as being good or neutral retrieved more relevant images than bad query suggestions (see Figure 6.9b). This is convenient, since it is unlikely that a user will use a suggestion considered to be bad. A particular behavior that was also observed is that performance tends to be better for suggestions that were derived using more selected (relevant) images. Then, overall, as more images are selected, it is less likely that the system will suggest a query; however if there is a suggestion it tends to be a better one.

Another observation from the evaluation was that suggestion quality depends highly on the particular query. There are some queries where images presented to the user clearly belong to different subgroups, which, if selected, most of the time a query will be suggested that relates to that subgroup. An example of a query that provides good suggestions was shown in Figure 6.6.

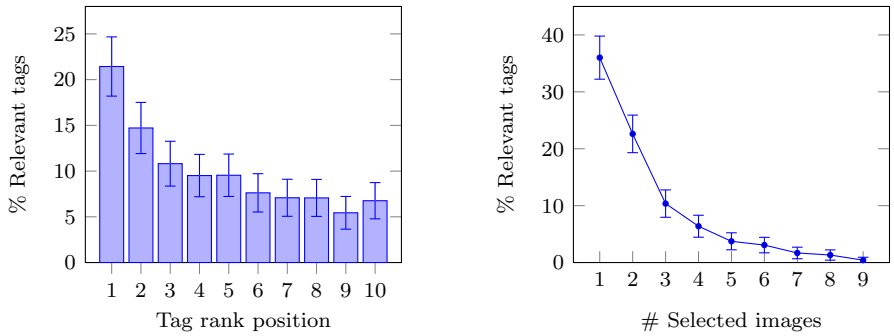
Tag Cloud Evaluation

In the same way as in query refinement evaluation, obtaining labeled data to be able to assess tag cloud suggestions is rather difficult. Thus, to perform the evaluation, we conducted again an informal field study, using the same database used in RISE prototype.

Fourteen users aged 31.42 (SD=5.34) were recruited via email advertising to participate in the evaluation study. They were told to assess the relevance of the $N = 10$ top scored tags suggested in the cloud for a series of queries (12 queries per person on average).

The list of queries was compiled by merging two lists from ImageCLEF 2012: Photo Annotation and Retrieval. Concretely, we merged concepts from 'Large-scale annotation using general Web data' subtask[8] and queries used in 'Visual concept detection, annotation, and retrieval using Flickr photos' subtask[9]. The final list comprised 164 search queries in total.

[8] http://imageclef.org/2012/photo-flickr
[9] http://imageclef.org/2012/photo-web



(a) Relevance vs. Tag position

(b) Relevance vs. Selected images

Figure 6.10: Evaluation results of the tag cloud, with 95% confidence intervals.

While evaluating each query, participants had to follow their given list of queries and select a subset of images for different subtopics from the presented set of 10 images. Participants had no restrictions on subtopic selection, e.g., a subtopic could have an arbitrary number of images, no minimum or maximum subtopics per query were imposed, etc.

Whenever a relevant image was selected from the presented set, a list of 10 tags was displayed in order of relevance (most relevant tags at the beginning of the list, in a left-to-right order). A check box was attached to each tag, so that users could mark whether the tag was considered relevant to the subtopic or not.

It is worth pointing out that no tag cloud in the strict sense was displayed, but a text-only tag list sorted by relevance, since we wanted to avoid any possible visual bias in the study. Figure 6.10 shows the evaluation results.

Evaluation Discussion Users reported that sometimes tags were found to be really useful and beneficial for the current query, but also sometimes they were found to be meaningless. This fact is explained by the noise due to the image indexing procedure, which was completely unsupervised and therefore the cloud may contain irrelevant tags for a particular query. This can be observed in Figure 6.10a, where each bar represents the average percentage of relevant tags (normalized by the number of selected relevant tags) given the rank position of each tag. Nonetheless, as expected, tags in the first positions of the cloud tended to be perceived more often as relevant. Differences between the first ranked tag and the other tags are statistically significant.

A study by Bateman et al. [2008] reported that tags with a larger number of characters tended to be selected less often. We investigated whether this could be observed in our study as well. We computed the tag length ratio as the division of the average length of selected tags by the average length of all

suggested tags, and obtained 1.03 (SD=0.16), which means that selected tags were around the average tag length overall. Furthermore, only 10% of the time a user chose a tag that had more than a 1.1 of tag length ratio. This suggested that the length of a tag was not determinant to assess its relevance toward a particular query, but also that users did not choose neither shorter or longer tags overall.

Figure 6.10b depicts the proportion of relevant tags according to the number of selected images. As observed, relevance differences between tags presented when selecting #1 or #2 images with respect to the rest of selections were found to be statistically significant. Similar conclusions to those observed in the query suggestion evaluation were derived: 1) as more images are selected, the overview the tag cloud provides about such a set of images tends to be more general; and 2) the quality of the tags depends highly on the particular query.

As observed, therefore, when a single image is selected, nearly half of the tags are considered as relevant, since the tag cloud is specifically tailored to such a single selection. Then, this proportion falls dramatically as more images are selected. This suggests that when many images are selected, a new strategy for generating tag clouds should be devised. Nonetheless, on average, 21.49% (SD=10) of the presented tags were considered as relevant at any time.

All in all, our study indicates that the tag cloud approach supports its intended goal, i.e., impression formation about a particular set of relevant images. Furthermore, the tag cloud provides the user with more options to refine the initial (textual) query. As such, we believe that a tag cloud has more potential than a query refinement suggestion, at least in an interactive image retrieval scenario.

6.4 Conclusions and Future Work

The IPR framework proposes a radically different approach to correct the errors committed by a PR system. This approach is characterized by human and machine being tied up in a much closer loop than usually. This way, errors can be avoided beforehand and correction costs can be dramatically reduced. We have characterized the interaction protocol that rules the IPR framework, and have introduced a series of prototypes that successfully illustrate it.

The literature had reported good experimental results in simulated-user scenarios, where IPR is focused on optimizing some automatic metric. However, user productivity is strongly related to how users interact with the IPR system and other UI concerns. For instance, in the NLP applications introduced in this chapter, a hypothesis that changes on every keystroke might obtain better automatic results, whereas user productivity may decrease because of the cognitive effort needed to process those changes. Therefore, the current IPR framework should be revised in order to optimize further these NLP systems

toward the user. In this regard, we have suggested some approaches, such as avoid modifying any user-submitted correction by any means, or deriving a new hypothesis only when the system is sure that it will improve the previous one.

Regarding IPR systems that deal with desultory user input, we have shown that implicit interaction can notably improve system performance. We have presented a series of image retrieval strategies to illustrate this fact, such as *1*) rethinking the classical retrieval protocol, in which users must indicate which images are relevant *and* non-relevant, to a much simpler one in which the system can assume that non-selected images are not relevant; *2*) combining multimodal information from selected images to provide better results; *3*) using this multimodal information to provide the user with optional suggestions, either in the form of a refined query suggestion or a tag cloud.

We have demonstrated that the techniques presented so far are both suitable and convenient. Each technique is based on a probabilistic model to handle user interaction, which allows IPR systems to take the lead in coordinating different user feedback signals. We hope these considerations will guide researchers to future developments that can have a significant impact both on academia and industry.

Bibliography of Chapter 6

V. Alabau, L. A. Leiva, D. Ortiz-Martínez, and F. Casacuberta. User evaluation of interactive machine translation systems. In *Proceedings of the European Association for Machine Translation (EAMT)*, pp. 20–23, 2012.

S. Barrachina, O. Bender, F. Casacuberta, J. Civera, E. Cubel, S. Khadivi, A. L. Lagarda, H. Ney, J. Tomás, E. Vidal, and J. M. Vilar. Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1):3–28, 2009.

S. Bateman, C. Gutwin, and M. Nacenta. Seeing things in the clouds: the effect of visual features on tag cloud selections. In *Proceedings of the nineteenth ACM conference on Hypertext and hypermedia (HT)*, pp. 193–202, 2008.

M. Bisani and H. Ney. Bootstrap estimates for confidence intervals in ASR performance evaluation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 409–12, 2004.

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.

A. Culotta, T. Kristjansson, A. McCallum, and P. Viola. Corrective feedback and persistent learning for information extraction. *Artificial Intelligence*, 170(14–15):1101–1122, 2006.

L. A. Leiva, V. Romero, A. H. Toselli, and E. Vidal. Evaluating an interactive-predictive paradigm on handwriting transcription: A case study and lessons learned. In *Proceedings of the 35th Annual IEEE Computer Software and Applications Conference (COMPSAC)*, pp. 610–617, 2011a.

L. A. Leiva, M. Villegas, and R. Paredes. Query refinement suggestion in multimodal interactive image retrieval. In *Proceedings of the 13th International Conference on Multimodal Interaction (ICMI)*, pp. 311–314, 2011b.

D. Ortiz-Martínez, L. A. Leiva, V. Alabau, and F. Casacuberta. Interactive machine translation using a web-based architecture. In *Proceedings of the 16th International Conference on Intelligent User Interfaces (IUI)*, pp. 423–425, 2010.

R. Paredes, T. Deselaers, and E. Vidal. A probabilistic model for user relevance feedback on image retrieval. In *Proceedings of the 5th international workshop on Machine Learning for Multimodal Interaction (MLMI)*, pp. 260–271, 2008.

G. Pask. *Conversation, cognition and learning: A cybernetic theory and methodology*. Elsevier Science, 1975.

V. Romero, L. A. Leiva, A. H. Toselli, and E. Vidal. Interactive multimodal transcription of text images using a web-based demo system. In *Proceedings of the 15th International Conference on Intelligent User Interfaces (IUI)*, pp. 477–478, 2009.

F. M. Segarra, L. A. Leiva, and R. Paredes. A relevant image search engine with late fusion: Mixing the roles of textual and visual descriptors. In *Proceedings of the 16th International Conference on Intelligent User Interfaces (IUI)*, pp. 455–456, 2011.

A. H. Toselli, V. Romero, M. Pastor, and E. Vidal. Multimodal interactive transcription of text images. *Pattern Recognition*, 43(5):1814–1825, 2009.

A. H. Toselli, E. Vidal, and F. Casacuberta, editors. *Multimodal Interactive Pattern Recognition and Applications*. Springer, 1st edition, 2011.

TT2. TransType2 - computer assisted translation. project technical annex, 2001. Information Society Technologies (IST) Programme, IST-2001-32091.

E. Vidal, L. Rodríguez, F. Casacuberta, and I. García-Varea. Interactive pattern recognition. In *Proceedings of the 4th Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, pp. 60–71, 2007.

M. Villegas and R. Paredes. Image-text dataset generation for image annotation and retrieval. In *II Congreso Español de Recuperación de Información (CERI)*, pp. 115–120, 2012.

Chapter 7

General Conclusions

The work presented in this thesis has focused on the topic of implicit interaction in HCI. An implicit interaction is an action a user performs with little (or no) awareness but which a computerized system can understand as input. As such, the role of implicit interaction consists in leveraging as much information as possible derived from a natural user input, without requiring the user to be aware of the data the system needs to operate, i.e., in a completely transparent procedure. By leveraging these implicit interactions, we can increase the richness of communication and make it possible to produce more useful applications and/or services. Implicit interaction can be considered as a consequence of ubiquitous computing, with the notable difference that is the user who takes the initiative to interact with the system. Implicit interaction, therefore, enables the ability to serve a person's information (or interaction) needs without becoming a burden. Finally, implicit interaction requires no training and provides context for actions. As such, it sets an interesting theoretical basis for a systematic approach to analyzing, optimizing, and enhancing a wide variety of computer applications.

7.1 Summary

Five chapters have illustrated the value of implicit interaction in a series of scenarios, namely activity tracking and video visualization (Chapter 2), behavioral clustering (Chapter 3), multitasking and task interruptions (Chapter 4), UI adaptation and redesign (Chapter 5), and interactive pattern recognition (Chapter 6). The main contributions of this thesis, thus, include:

1. A tracking plus hypervideo tool to understand user behavior through implicit interactions.
2. A method to classify web pages according to implicit interactions, together with a novel algorithm for clustering sequential data.

3. A method to ease multitasking that is based on implicit interaction cues as a visual remainder.
4. A method to transparently adapt a UI to the capabilities of the user (or a group of users) by mining implicit interactions.
5. A series of prototypes that implement a novel IPR framework that is guided by implicit interaction principles.

In sum, virtually any application can benefit from an implicit HCI framework. The main advantages include:

- Implicit interactions can be gathered for free, without burdening the user.
- Every user interaction may contribute to enhance system utility.
- Implicit interactions are useful to understand how people interact with computers.

Finally, the main drawbacks of dealing with implicit interactions can be summarized as follow:

- Implicit interactions do not provide always clear information.
- Some assumptions need to be made.
- Implicit interactions are a useful but complementary tier.

7.2 Future Outlook

While this thesis has researched implicit interaction in the context of web-based HCI applications, we have just barely scratched the surface when it comes to exploiting its truly potential. Invisible (or pervasive) computing is already all around us, making computers that fit the human environment and not the other way round [Kaushik, 2012]. Implicit interaction can only help to contribute in this regard by providing novel sources of perception and interpretations of the users within their computers and devices.

Tennenhouse [2000] claimed that, over the past 40 years, computer science has addressed only about 2% of the world's computing requirements. As stated by Cadez et al. [2003], arguably one of the great challenges in the coming century will be the understanding of human behavior in the context of "digital environments". In such a context, implicit interaction is certainly an important starting point. Devices that have (sometimes very limited) perceptional capabilities have *already* started the shift from explicit HCI toward a more implicit interaction with machines [Schmidt, 2000]. This being so, implicit interaction

will definitely gain more interest in HCI research, to make computers more useful and tailored to our needs.

It is clear that explicit interaction will continue having a primary presence in software applications, and that implicit interaction will be used as an additional source of (otherwise valuable) information. Nonetheless, probably in a (not so far) future, desires and intentions would be enough to get computers to act on our behalf.

Additional References

I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Model-based clustering and visualization of navigation patterns on a web site. *Data Mining and Knowledge Discovery*, 7(4):399–424, 2003.

P. Kaushik. Ubiquitous computing: Blurring the mind and machine gap. Available at http://www.huffingtonpost.co.uk/preetam-kaushik/ubiquitous-computing-blur_b_1520173.html, 2012. Retrieved July 23, 2012.

A. Schmidt. Implicit human-computer interaction through context. *Personal and Ubiquitous Computing*, 4(2):191–199, 2000.

D. Tennenhouse. Proactive computing. *Communications of the ACM*, 43(5):43–50, 2000.

Appendix A

Research Dissemination

One of the high-level goals of this thesis is to promote dissemination of its research results and the technologies discussed so far, in order to contribute to the body of knowledge both in academia *and* industry.

With respect to dissemination amongst the scientific community, to date, this thesis has generated +30 publications, most of which have been presented in top-tier venues[1]. To sum up, these contributions include:

- 23 conference papers,
- 3 journal papers,
- 3 workshop papers,
- 3 book chapters, and
- 2 research awards plus 1 mention.

Regarding dissemination amongst the industry, the contributions of this thesis include:

- 4 technology transfer projects, and
- 3 issued patents[2].

Moreover, most of our results have been disseminated through videos, demonstrations, informational brochures, and informative articles in the news and press. Some of the prototypes have been awarded in national and international competitions, and others have raised the interest of some ICT companies. This

[1] Conference rankings and acceptance rates are reported for the year of publication.
[2] Not listed here because of pending status.

shows a clear awareness from the society to start embracing these novel technologies. Furthermore, almost all developed prototypes are now part of the catalogue of technology supply at our university: the CARTA programme[3].

Finally, a number of follow-up activities should be performed, among which we highlight: technology watch, search for additional funding sources, deploy R&D management activities, and establish strategic alliances. It is expected that, in the medium term, these strategies will lead to novel emerging technologies and more tech transfer projects related to a greater or a lesser extent to this thesis.

List of Publications

V. Alabau, J. M. Benedí, F. Casacuberta, L. A. Leiva, D. Ortiz-Martínez, V. Romero, J. A. Sánchez, R. Sánchez-Saez, A. Toselli, and E. Vidal. CAT-API framework prototypes. In *Workshops on Database and Expert Systems Applications (DEXA)*, pp. 264–265, 2010a.

V. Alabau, F. Casacuberta, L. A. Leiva, D. Ortiz-Martínez, and G. Sanchis-Trilles. Sistema web para la traducción automática interactiva. In *Actas del XI Congreso Internacional de Interacción Persona Ordenador (INTERACCION)*, pp. 47–56, 2010b. 37% acceptance.

V. Alabau, L. A. Leiva, D. Ortiz-Martínez, and F. Casacuberta. User evaluation of interactive machine translation systems. In *Proceedings of the European Association for Machine Translation (EAMT)*, pp. 20–23, 2012. **CORE: B**. 54% acceptance.

L. A. Leiva. MouseHints: Easing task switching in parallel browsing. In *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems (CHI EA)*, pp. 1957–1962, 2011a. **CORE: A**. 42% acceptance.

L. A. Leiva. Mining the browsing context: Discovering interaction profiles via behavioral clustering. In *Adjunct Proceedings of the 19th conference on User Modeling, Adaptation, and Personalization (UMAP)*, pp. 31–33, 2011b. **CORE: B**. 20% acceptance.

L. A. Leiva. Restyling website design via touch-based interactions. In *Proceedings of the 13th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI)*, pp. 599–604, 2011c. **CORE: A**. 23% acceptance.

L. A. Leiva. Interaction-based user interface redesign. In *Proceedings of the 17th international conference on Intelligent User Interfaces (IUI)*, pp. 311–312, 2012a. **CORE: A**. 23% acceptance.

L. A. Leiva. ACE: An adaptive CSS engine for web pages and web-based applications. In *Proceedings of the WWW Dev Track*, 2012b. **CORE: A**. 45% acceptance.

L. A. Leiva. Automatic web design refinements based on collective user behavior. In *Proceedings of the 2012 annual conference extended abstracts on Human factors in computing systems (CHI EA)*, pp. 1607–1612, 2012c. **CORE: A**. 45% acceptance.

L. A. Leiva and V. Alabau. *Multimodal Interactive Handwritten Text Transcription*, chap. A Web-based Demonstrator of Interactive Multimodal Transcription. World Scientific Publishing, 2012.

[3] http://www.upv.es/carta/

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*, pp. 277–278, 2010. **CORE: A**. 35% acceptance.

L. A. Leiva and E. Vidal. Revisiting the K-means algorithm for fast trajectory segmentation. In *Proceedings of the 38th International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, p. 86, 2011. **CORE: A**. 19% acceptance. [**ACM SRC semi-finalist award**].

L. A. Leiva and E. Vidal. Simple, fast, and accurate clustering of data sequences. In *Proceedings of the 17th international conference on Intelligent User Interfaces (IUI)*, pp. 309–310, 2012. **CORE: A**. 23% acceptance.

L. A. Leiva and R. Vivó. (smt) Real time mouse tracking registration and visualization tool for usability evaluation on websites. In *Proceedings of the IADIS International Conference on WWW/Internet*, pp. 187–192, 2007a.

L. A. Leiva and R. Vivó. (smt) Herramienta de registro y visualización de mouse tracking en tiempo real para evaluación de usabilidad en sitios web. *Novática*, 189(1):53–60, 2007b.

L. A. Leiva and R. Vivó. A gesture inference methodology for user evaluation based on mouse activity tracking. In *Proceedings of Interfaces and Human Computer Interaction (IHCI)*, pp. 58–67, 2008.

L. A. Leiva and R. Vivó. Interactive hypervideo visualization for browsing behavior analysis. In *Proceedings of the 21st international conference companion on World Wide Web (WWW)*, pp. 381–384, 2012. **CORE: A**. 45% acceptance.

L. A. Leiva, D. Ortiz-Martínez, E. Cubel, G. Sanchis, V. Romero, R. Sánchez-Saez, V. Alabau, A. H. Toselli, J. A. Sánchez, J. M. Benedí, E. Vidal, and F. Casacuberta. Nuevas tecnologías interactivo-predictivas multimodales para el procesamiento de lenguaje natural sobre internet. Valencia IDEA research competition, ICT category, 2010. [**Accesit award**].

L. A. Leiva, V. Alabau, V. Romero, F. M. Segarra, R. Sánchez-Sáez, D. Ortiz-Martínez, L. Rodríguez, and N. Serrano. *Multimodal Interactive Pattern Recognition and Applications*, chap. Prototypes and Demonstrators. Springer, 2011a.

L. A. Leiva, V. Romero, A. H. Toselli, and E. Vidal. Evaluating an interactive-predictive paradigm on handwriting transcription: A case study and lessons learned. In *Proceedings of the 35th Annual IEEE Computer Software and Applications Conference (COMPSAC)*, pp. 610–617, 2011b. **CORE: B**. 20% acceptance.

L. A. Leiva, M. Villegas, and R. Paredes. Query refinement suggestion in multimodal interactive image retrieval. In *Proceedings of the 13th International Conference on Multimodal Interaction (ICMI)*, pp. 311–314, 2011c. **CORE: B**. 39% acceptance.

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. **CORE: A**. 25% acceptance.

D. Ortiz-Martínez, L. A. Leiva, V. Alabau, and F. Casacuberta. Interactive machine translation using a web-based architecture. In *Proceedings of the 16th International Conference on Intelligent User Interfaces (IUI)*, pp. 423–425, 2010. **CORE: A**. 30% acceptance.

D. Ortiz-Martínez, L. A. Leiva, V. Alabau, I. García-Varea, and F. Casacuberta. An interactive machine translation system with online learning. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pp. 68–73, 2011. **CORE: A**. 52% acceptance.

V. Romero, L. A. Leiva, V. Alabau, A. H. Toselli, and E. Vidal. A web-based demo to interactive multimodal transcription of historic text images. In *Proceedings of the 13th European Conference on Digital Libraries (ECDL)*, volume 5714 of *LNCS*, pp. 459–460. Springer-Verlag, 2009a. **CORE: A**. 21.7% acceptance. [**Best demo award**].

V. Romero, L. A. Leiva, A. H. Toselli, and E. Vidal. Interactive multimodal transcription of text images using a web-based demo system. In *Proceedings of the 15th International Conference on Intelligent User Interfaces (IUI)*, pp. 477–478, 2009b. **CORE: A**. 29% acceptance.

R. Sánchez-Sáez, L. A. Leiva, J. A. Sánchez, and J. M. Benedí. Interactive predictive parsing using a web-based architecture. In *Proceedings of the 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pp. 37–40, 2010a. **CORE: A**. 35% acceptance.

R. Sánchez-Sáez, L. A. Leiva, J. A. Sánchez, and J. M. Benedí. IPP-Ann: An interactive tool for probabilistic parsing. In *Workshops on Database and Expert Systems Applications (DEXA)*, pp. 255–259, 2010b.

R. Sánchez-Sáez, L. A. Leiva, J. A. Sánchez, and J. M. Benedí. Seamless tree binarization for interactive predictive parsing. In *Proceedings of VI Jornadas en Tecnología del Habla (FALA)*, pp. 47–50, 2010c.

R. Sánchez-Sáez, L. A. Leiva, J. A. Sánchez, and J. M. Benedí. Interactive predictive parsing framework for the spanish language. *Sociedad Española para el Procesamiento del Lenguaje Natural*, 45(1):121–128, 2010d.

F. M. Segarra, L. A. Leiva, and R. Paredes. A relevant image search engine with late fusion: Mixing the roles of textual and visual descriptors. In *Proceedings of the 16th International Conference on Intelligent User Interfaces (IUI)*, pp. 455–456, 2011. **CORE: A**. 44% acceptance.

M. Villegas, L. A. Leiva, and R. Paredes. *Multimodal Interaction in Image and Video Applications*, chap. Interactive Image Retrieval based on Relevance Feedback. Springer, 2012. In press.

List of Figures

List of Tables

Index

T

U

V

W

X

About the Cover of this Thesis

Perhaps the most famous image of an iceberg is the one produced in 1999 by Ralph A. Clevenger, which actually is not a real image. As he pointed out in a personal communication:

> *The iceberg image is a composite image that I created many years ago, from four of my images, to illustrate the concept of the unseen portion of an iceberg. The two halves of the iceberg are 2 separate shots that I took in Alaska and in Antarctica (neither is underwater). The only underwater part is the background that I took off the coast of California. The sky is the last component. It took a lot of research on lighting and scale to get the iceberg to look real.*

Some time later, the poster company Successories captioned the image as "The Essence of Imagination", with the following accompanying text: *"What we can easily see is only a small percentage of what is possible. Imagination is having the vision to see what is just below the surface; to picture that which is essential, but invisible to the eye".*

This image thus illustrates the concept of "what you see is not necessarily what you get". For such a reason it was chosen for the cover of this thesis. When we interact with someone we often just hear their words and see their (partial) behavior. It is like seeing only the tip of the iceberg: there is much more below the surface that we may not even be aware of. I often use this concept to illustrate the role of implicit interaction in HCI.