Blending Speech and Graphical User Interfaces

An empirical study on multimodal mobile interaction

Manolis Perakakis

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in the Department of Electronics and Computer Engineering
Technical University Crete

Chania, November 2011

Thesis Committee:

Assoc. Prof. Alexandros Potamianos (advisor)
Prof. Vasilios Digalakis
Assis. Prof. Aikaterini Mania

Acknowledgements

First, I would like to thank my supervisor Assoc. Prof. Alexandros Potamianos, for giving me the opportunity to work on such an interesting research topic. Additionally, I am thankful to him for being patient and encouraging, providing constant support to my work. I really feel fortunate to have the privilege of working with him.

I would also like to express my gratitude to the members of my doctoral committee Prof. Vassilios Digalakis, and Assist. Prof. Aikaterini Mania. Many thanks also go to rest committee for participating in the examination of my thesis.

I would also like to thank all my colleagues and especially Michalis Toutoudakis (who contributed parts of Chapter 6), Theofanis Kannetis and Valia Kouloumenta, all members of the multimodal team in our lab, for their cooperation. Special thanks also go to all the participants who volunteered to the evaluation experiments. Among many of my friends to whom I am grateful for their support all these years I need to thank Kyprianos, Konstantinos and Kalliopi.

Finally I would like to thank my parents Tilemachos and Galini. They gave me constant support in pursuing my dreams. This work is dedicated to them.

This thesis is part of the 03ED375 research project, implemented within the framework of the Reinforcement Programme of Human Research Manpower (PENED) and co-financed by National and Community Funds (25% from the Greek Ministry of Development-General Secretariat of Research and Technology and 75% from E.U.-European Social Fund).

Abbreviations

HCI	Human Computer Interaction
GUI	Graphical User Interface
ZUI	Zooming User Interface
MVC	Model View Controller
PDA	Personal Digital Assistant
MM	Multi-modal
GO	Gui-Only interaction mode
SO	Speech-Only interaction mode
CT	Click-to-Talk interaction mode
OM	Open-Mike interaction mode
OMSI	Open-Mike Speech Input interaction mode
MS	Modality-Selection interaction mode
VUI	Voice User Interface
VAD	Voice Activity Detection
SDS	Spoken Dialogue System
DM	Dialogue Manager
ASR	Automatic Speech Recognition
WER	Word Error Rate
CER	Concept Error Rate
TTS	Text To Speech synthesis
NLP	Natural Language Processing
NLU	Natural Language Understanding
NLG	Natural Language Generation
HMM	Hidden Markov Model
FSM	Finite State Machine
BCI	BrainComputer Interface
EEG	Electro-Encephalo-Graphy
ERP	Event-Related Potential
ANOVA	Analysis Of Variance

Contents

List of Figures

List of Tables

Abstract

Mobile phones have already outnumbered personal computers. Although until recently the majority of phones were used mainly as voice communication devices, the emergence of powerful application-centric mobile devices such as personal digital assistants (PDAs) and smart-phones, has created excitement for the future of mobile computing. Despite the recent explosion of advanced mobile applications such as web browsing and video consuming, constraints such as reduced display size and limited input interaction methods pose new challenges for interaction designers. The use of more than one interaction modalities has been proposed as a possible solution to overcome these limitations. Multimodal interfaces process two or more combined user input modalities such as speech, pen or touch, in a coordinated manner with multimedia system output and can potentially offer more rich, robust and adaptive interaction experience.

This dissertation investigates multimodal interface design and evaluation with a focus on mobile interaction. One of the main aims is to showcase how to design information-filling multimodal systems that combine speech and graphical user interface (GUI) input (e.g. pen or touch). From the interaction design standpoint, the main focus is on identifying and exploiting the synergies resulting from the mixing of modalities in order to create robust and effective interfaces. The system designed and implemented, allows both unimodal and multimodal interaction and can be used across different platforms such as PCs, PDAs and mobiles such as the popular iPhone device.

For the evaluation of multimodal interaction both established and novel metrics are employed. Two new metrics were devised that measure the relation of input modality preferences to unimodal efficiency and the synergies found in a multimodal system. The proposed metrics, relative modality efficiency and multimodal synergy, can provide valuable information to the interaction design process of multimodal systems. Furthermore affective evaluation incorporating biosignals such as skin conductance and brain waves (EEG) has provided a rich amount of data not previously available. Use of such physiological channels and their elaborated interpretation is a challenging but also a potentially rewarding direction towards emotional and cognitive assessment of multimodal interface design.

Evaluation results show that multimodal systems can potentially outperform unimodal systems in terms of both performance and user satisfaction when designed to maximize the synergies between the modalities. Overall this research entails significant implications for designing efficient mobile interfaces.

Chapter 1

Introduction

Although graphical user interfaces (GUIs) have been the dominant user interface technology for the past two decades, today's computing platforms (ranging from mobile devices to large wall displays) call for new, more natural and efficient ways of interaction. Recently, there has been much interest in investigating alternative input/output interaction modalities that go beyond the traditional keyboard and mouse input, and text and graphics output.

GUIs are the dominant interface technology in part due to the high bandwidth they provide on the output side (matching our advanced vision processing capabilities). In general, information can be better organized and presented to the user using graphical output compared to other modalities. Thus GUIs today are used not only in personal computers (PCs) but also in every computing based platform such as intelligent information kiosks, portable and mobile devices and automated teller machines (ATMs). On the input side, GUIs use for selection, pointer devices such as mouse on desktop computers or pen devices on portable systems with touch-screens. Some touch-screens support *multi-touch sensing* allowing input through one or more fingers which is considered more natural than using a pen. For text input, desktop computers use keyboard, while portable and mobile devices use methods such as keypads, miniaturized physical keyboards, virtual keyboards, or various handwriting recognition methods such as graffiti input (see Section 2.6.2).

Speech is the most natural form of communication among humans, but it has several limitations when used in HCI. Although speech recognition technology has been studied actively during the past decades and highly sophisticated recognizers have been constructed, machines are far from matching human speech recognition performance, especially in adverse recording conditions.

Speech and GUI interfaces have been extensively studied and compared in the literature, e.g., [8, 9]. With GUIs everything the user wants to do at any given time must be presented at the screen, while speech interfaces lack visual information and require users to memorize

all meaningful information. In addition, the sequential nature of speech loads the short-term memory and takes up the linguistic channel, which makes speech interfaces unsuitable for some tasks. As an output channel, speech is slow because of its sequential nature, while GUIs convey information in parallel thus making them suitable for presenting a large amount of information. Speech output may be more appropriate for grabbing attention and offering an alternative feedback mechanism to the user, rather than conveying a large amount of information [9].

Spoken interaction may be faster when users immediately say what they want to achieve without going through GUI menu hierarchies. Spoken messages may also be more expressive and convey richer information compared to GUI actions, such as the selection of similar objects among a large number of them. However the freedom and efficiency that speech offers to the user, makes speech more difficult for the computer to handle. It is also hard for users to know the limitations of what they can say and how to explore the set of possible tasks they can perform [9]. Finally, users interacting with speech interfaces do not have the same feeling of control usually allowed by GUI interfaces. This is because speech input may be *inconsistent* due to recognition errors, i.e., the recognition result may be different for the same sentence spoken twice. Handling speech errors efficiently is a key issue for successful speech applications. Well designed spoken dialogue systems or the use of additional modalities in multimodal systems can alleviate these problems and allow for efficient and natural speech interaction.

Multimodal interfaces [10, 11, 12] that combine speech input with other modalities have been hailed as the solution to the speech robustness problem. Multimodal interfaces offer increased *robustness* and error correction capabilities against error-prone modalities due to both user behavior and system design. It was found [13, 14] that for multimodal dialogue systems users tend to use simpler language compared to when interacting unimodaly. Also users tend to use the less error-prone modality at each context (error avoidance) and switch modalities after system errors (synergistic error correction). These behaviors can be reinforced by appropriate user interfaces design, e.g., use of pen input for correcting speech recognition errors.

The emergence of powerful mobile devices such as personal digital assistants (PDAs) and smart-phones, raises new constraints but also design challenges that could be better addressed by a combination of more than one modalities. Efforts to build multimodal interfaces for PDAs are described in [15, 16, 17]. These systems inspired by Bolt's "Put that there" [18] prototype mainly focus on map applications that can use speech and pen (gesture) input in a simultaneous fashion. Although map-based applications exemplify the advantages of multimodal vs. unimodal interaction by maximizing the synergies between modalities, information-seeking applications that involve form-filling are much more common in mobile devices, e.g., travel information and reservation, financial information and transactions, entertainment information. Typical form-filling applications used in MiPad [19, 20] a multimodal PDA prototype use

"Tap and Talk" (a.k.a."Click-to-Talk") sequential multimodality (as opposed to concurrent multimodality [21]), i.e., only one input modality is active at each interaction turn. This work, focuses on information-seeking multimodal systems which combine speech and GUI input, and on the investigation of a variety of multimodal interaction modes in addition to "Click-to-Talk".

1.1 Research questions and goals

A fundamental research question in the design of multimodal interaction is the identification and exploitation of synergies between modalities in order to maximize efficiency and user satisfaction. *Synergy* is a design principle that applies to systems that support more than one input or output modalities. Synergistic multimodal interface design can achieve multimodal interface performance that is better compared to the sum of it's unimodal parts. To achieve this goal it is important not only to use the appropriate modality for each application task, but also to allow for interplay between them, e.g., speech misrecognitions should be resolved via the GUI interface. A synergistic multimodal interface is more than the sum of its parts. Designing multimodal interfaces that effectively combine modalities [22, 23], exploit synergies, are robust and adapt to the users, is not a trivial task.

It is widely supported that voice user interfaces (VUI) and graphical user interfaces (GUI) when combined to create a multimodal system offer high complementarity for most applications [9, 24, 25, 26]. As far as input is concerned, GUI interfaces have low error rates and offer easy error correction. Speech interfaces are inconsistent in terms of input, since they may produce different recognition results for the same user utterances, causing a lack of control feeling to users. Although speech is not error-free, it may be more efficient for relatively high speech recognition accuracy and high verbosity (number of tokens communicated per turn). It is also the most natural type of input compared to other modalities for many applications. As far as output is concerned, GUI output is fast (parallel) compared to much slower (sequential) speech output.

Thus, one of the main goals of this work is to *show how multimodal systems that combine GUI and speech interfaces can potentially become more efficient* by taking advantage of: (i) "input modality choice" synergy, i.e., the user (or system in an adaptive user interface) chooses the most appropriate input modality for each turn (ii) "visual-feedback", i.e., the more efficient cognitive processing of visual compared to auditory information, (iii) "error-correction" synergy, i.e., correcting errors of the VUI via the GUI [27].

An important question one has to consider when building multimodal interfaces is the suitability of various input methods for different tasks and subtasks [23]. For example, in [28] the authors compared data entry of isolated word Automatic Speech Recognition (ASR) with keyboard/mouse interfaces for three different data entry tasks: textual phrase entry,

selection from a list and numerical data entry. Results indicated that speech input is faster for textual phrase entry if typing speed is below 45 words/minute. It is also faster for list selection when the list contains more than 15 items but offers no advantage over keypad or mouse for numerical data entry. Combining multiple modalities efficiently is a complex task and requires both good interface design and experimentation to determine the appropriate modality mix. Few guidelines exist for selecting the appropriate mix of modalities [22, 29, 30]. It is often the case when designing multimodal user interfaces that the developer is biased either toward the speech or the GUI modality. This is especially true, if the developer is speech-enabling an existing graphical user interface (GUI)-based application or building a GUI for an existing speech-only service.

Another question that is not thoroughly researched is the design of multimodal turn-taking and the selection of the most efficient interaction modality at each turn. Should users be allowed to interact as in traditional spoken dialogue systems (SDS) where a voice-activity detector allows the user to barge-in and speak at any moment (commonly referred as an "Open-Mike" interaction mode), should the user be constrained as in the GUI paradigm to press a button to activate the speech recognizer ("Click-to-Talk"), or should either interaction modes be used were appropriate. One of the goals of this work is to *investigate input modality usage from the user point of view and to better understand efficiency considerations and user biases in input modality selection.* Such information would be valuable for user modeling and multimodal dialogue system design in general.

Evaluation [31, 32] of multimodal interfaces is an important task and can help design better interfaces. Although some efforts [33, 34] have emerged that attempt to build a unifying framework for the evaluation of speech and multimodal interfaces, there are various difficulties and issues [35]. Additionally, an important difficulty that arises is that the diversity of possible modalities and the different ways in which they can be combined to result a large number of different types of multimodal applications makes evaluation methodology even harder. Thus in practice, evaluation of multimodal systems is based on traditional metrics used in human-computer interaction. Objective metrics such as speed, number of errors, task completion, are usually computed for the various system configurations along with subjective metrics [36, 37, 38] and are statistically analyzed [39, 40] to determine the best system.

In this work, a travel reservation form-filling multimodal dialogue system is implemented and evaluated for both desktop and PDA environments. The desktop system combines keyboard, mouse and speech input while the PDA system combines pen and speech input. Three multimodal interaction modes were implemented, that differ in multimodal turn-taking and the selection of the default input modality, namely: "Click-to-Talk", "Open-Mike" and "Modality-Selection". For "Click-to-Talk" interaction, GUI is the default input modality while for "Open-Mike" interaction, speech is the default input modality. "Modality-Selection" is a mixture of

the other two multimodal modes. The multimodal systems are evaluated and compared with the unimodal systems ("Speech-Only", "GUI-Only"). To compare the efficiency of the various systems, not only the objective metrics among the different systems are computed, as is typically done in the literature, but also the various factors that could affect the efficiency and modality choice by the user are measured in detail. For this purpose, the break down of turn duration to interaction and inactivity times is proposed in order to better understand the effect of input modality on interface efficiency. In addition, modality usage is measured for different levels of relative efficiency of the input modalities.

A fundamental aim of this study is to investigate how factors such as unimodal efficiency, interface design and user characteristics affect (or bias) input modality selection. For this purpose two new evaluation metrics are proposed, namely "relative modality efficiency" and "multimodal synergy". Relative modality efficiency when compared with modality usage identifies bias and suboptimal use of modalities in the course of the interaction. Multimodal synergy expresses in a single number the percent of interface efficiency improvement compared to the combined unimodal interface efficiency. Multimodal synergy is used to identify problems in effectively combining various modalities. The proposed metrics are shown to be useful tools for identifying usability problems in multimodal systems.

Affective evaluation is also an interesting research direction towards a qualitative assessment of the interaction experience. Employing physiological measurements from skin conductance and EEG allows real time monitoring of interaction which can provide valuable information to the designer. It can help detect problems in interaction and also allow for a direct qualitative comparison between input modalities and various interaction systems.

1.2 Contributions

The main contributions of this work are:

- The identification, exploitation and modeling of synergies between the speech and GUI modalities in the design of a multimodal dialogue system that supports both unimodal and multimodal interaction and can be used across different platforms such as PCs, PDAs and mobiles.

- A detailed evaluation of multimodal interaction modes and the comparison with unimodal modes which includes the break down of the turn duration into interaction and inactivity time to better investigate modality synergies.

- An evaluation methodology that proposes two new metrics for the investigation of the relationship between input modality efficiency and modality usage and the computation

of synergies and the quality of multimodal interaction modes, namely relative modality efficiency and multimodal synergy.

- A methodology for evaluating modalities and interaction systems in terms of affective metrics such as engagement, excitement and frustration.

1.3 Thesis Outline

The thesis is organized as follows. First the related literature review is presented in Chapter 2. The review is given with the aim of providing a guide to several topics related to the design of multimodal mobile interfaces. Initially a brief introduction in Human Computer Interaction (HCI) with emphasis on design and evaluation is given. Then the two interaction modalities of interest, namely speech and GUI are examined in detail. Background in multimodal interfaces with a focus on design, fusion of modalities and interaction patterns is presented next. Examples of multimodal systems design and the advantages of multimodal interfaces such as robustness and adaptation are highlighted. The challenges, issues and guidelines of designing mobile interfaces are discussed next with a focus on the limited input interaction methods found in these devices. Finally, architectures, standards and tools for designing and building multimodal interfaces are also discussed.

The main aim of Chapter 3 is to showcase how to design and build information-filling multimodal dialogue systems combining speech and GUI (e.g. pen or touch) input. From the interaction design standpoint, the main focus is on identifying and exploiting the synergies between the modalities and on the investigation of a variety of multimodal interaction modes in addition to "Click-to-Talk". The system architecture of a system that allows both unimodal and multimodal interaction and can be used across different platforms such as PCs, PDAs and mobile devices is also examined in this chapter and in more detail in Appendix A.

In Chapter 4 the methodology used for evaluating the system is presented with a focus on the evaluation metrics used. Some of these metrics are standard objective metrics used in dialogue systems while the rest were devised specially for the investigation of two important research questions, namely the relation of input modality choice to unimodal efficiency and the measurement of the synergies in multimodal interaction modes. Overall the metrics used aim at: (i) comparing in terms of performance and user satisfaction all the interaction modes (unimodal and multimodal). (ii) identifying input modality selection patterns in the multimodal interaction modes and their relation to unimodal efficiency, e.g. is modality selection proportional to the ratio of unimodal efficiency ratio? (iii) measuring the synergies of the multimodal interfaces.

In Chapter 5 detailed evaluation results using the metrics described in the previous chap-

ter are provided. Objective evaluation results include context statistics, user statistics, input modality overrides and distributions of turn duration times broken down into inactivity/interaction times and input modality type. Relative modality efficiency and multimodal synergy results are shown in detail and subjective evaluation results are also reported. An overall discussion of the results is also provided.

Chapter 6 deals with user behavior patterns and modality prediction. As has already been discussed in Chapter 5, there is significant variability in user behavior and more interestingly in modality selection patterns (as expected) among the users. The reason for this, mainly stems from the differences in unimodal efficiency each user exhibits but is not the only factor. A more detailed investigation of individual user behavior is provided in this chapter. Two important factors that affect modality usage and related to speech modality, namely speech verbosity and speech error correction patterns are discussed in detail. In addition a simple statistical model for predicting input modality selection is described and evaluated. Results, difficulties and possible ideas for improving prediction are also discussed.

In contrast with the previous chapters where evaluation of the interaction systems were based on metrics such as interaction speed, error rates, modality selection and synergy, Chapter 7 employs affective metrics such as excitement, frustration and engagement for the evaluation of the various systems. This, not only provides a more qualitative approach to evaluation, it also provides a better understanding of the interaction process. The methodology proposed is based on the use of two different modalities for the measurement of affect. The first is Galvanic Skin Response (GSR) which relates to the sympathetic nervous system and reveals emotional arousal. The second is Electroencephalography (EEG), a rich source of information which is able to reveal hints of both affective and cognitive state during an interaction task.

This thesis concludes with Chapter 8. It provides a short summary of the work presented in this thesis, discusses the main results and proposes plans for future work.

Chapter 2

Background

[1]

2.1 Introduction

Interface design is interdisciplinary by nature and requires both scientific expertise and creativity. Expertise in interaction modalities, multimedia, software engineering, cognitive psychology, human factors and ergonomics and graphic design is essential in order to create a successful interface. Creative thinking is also required in order to select the appropriate design among the numerous interface implementations possible for a specific task. An important interface design choice is the selection and mixing of input and output *modalities*, i.e., channels of communication, between the user and the system. In addition to traditional human-computer interaction (HCI) modalities, such as keyboard and mouse for input, and text and graphics for output, numerous "novel" modalities are available to today's interface designer, such as *speech*, *gestures* and *haptics*. New devices such as *augmented reality* displays, *force feedback* gloves, *eye-tracking* goggles, *force feedback* gloves and *multi-touch* displays have recently emerged that open the door to new interaction paradigms. The improved device capabilities and available interaction modalities have increased the freedom of choice for the designer, but also the complexity and challenges of interface design.

The purpose of this chapter is to familiarize the reader with fundamental concepts of human computer interaction and review the state-of-the-art in multimodal interface design. First, a short overview of human computer interaction is given in section 2.2, followed by a review of the various interaction modalities in section 2.3. Input and output modalities covered include graphical user interfaces (GUI), speech and gestures. Most interfaces are *multimodal*, i.e., employ more than one input or output interaction modalities. Multimodal interfaces

[1] This chapter is partial adaptation of published book chapter [41].

pose interesting challenges related to the combination or *fusion* of input modalities, and the combination or *fission* of output media streams and are reviewed in detail in section 2.4. As interfaces are becoming increasingly complex, personalization or adaptation of the interface to the user's needs and preferences is becoming a necessity. *Adaptive interfaces* use information from user profiles, user ratings or past user interaction patterns to update their behavior and to better serve the user, as discussed in section 2.5. *Mobile interfaces* are becoming increasingly important as multimedia data is more and more stored and consumed from mobile devices. Mobile interfaces have to cope with small device size, limited processing power, communication bandwidth and most notably limited interaction methods but can also take advantage of sensor input to improve *context awareness*, e.g., global position information, accelerometers. Design of mobile interfaces is reviewed in section 2.6. The chapter concludes with a review of architectures (section 2.7), tools, and standards (section 2.8) available for the design and development of unimodal and multimodal interfaces.

2.2 Human Computer Interaction

Human Computer Interaction (HCI) is the study of interaction between users and computer systems. HCI is a multi-disciplinary subject, combining topics such as: *psychology and cognitive science* that studies user's perceptual, cognitive, and problem solving skills, *ergonomics* (i.e., the study of the physical capabilities of the user), *design*, as well as *computer science*, and *engineering*. HCI is concerned among others with theories of interaction, development of new interfaces and interaction techniques, e.g. for mobile computing, methodologies for designing interfaces, implementation of software toolkits, design of hardware devices, and techniques for evaluating and comparing interfaces.

As the number, diversity, and complexity of interactive applications increases users need to continuously learn, adapt, and cope with new interfaces. As stated in [42]: "a long term goal of HCI is to design systems that minimize the barrier between the human's cognitive model of what they want to accomplish and the computer's understanding of the user's task." The call for interfaces that will be easier to learn and use is popularized by pioneers such as Dertouzos [43], and Shneiderman [44].

2.2.1 Theories of Interaction

The study of human beings in the context of HCI draws mainly from *cognitive psychology* that studies the capabilities and limitations of humans, how they perceive the world around them, how they store or process information and solve problems. Input-output channels (vision, hearing, touch, movement), human memory (sensory, short-term/working, and long-term

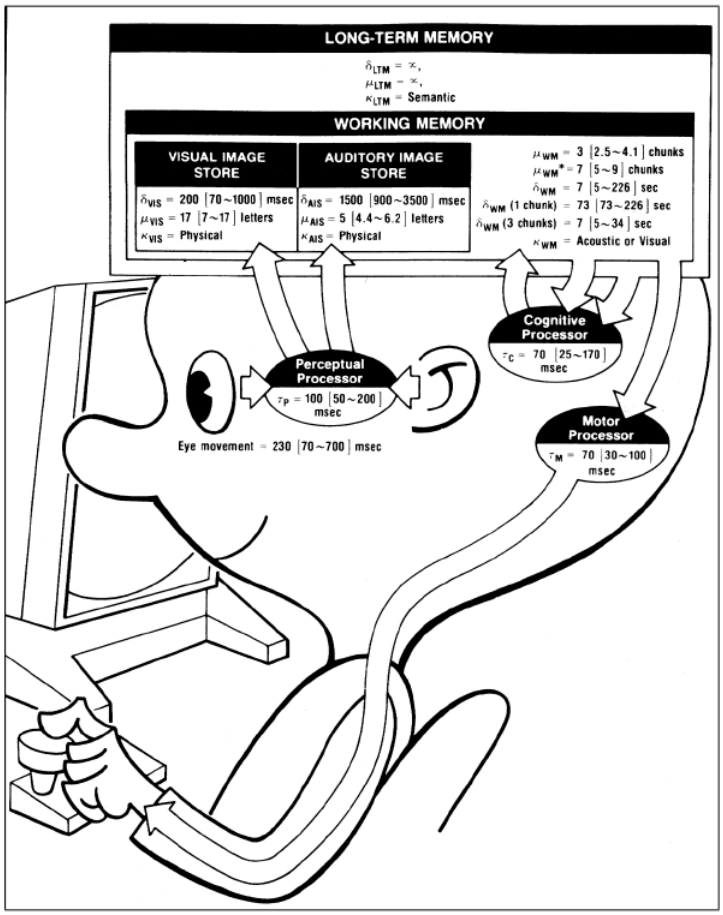
memory), and processing capabilities (reasoning, problem solving, skill acquisition) should all be considered when designing computer systems with *usability* in mind. For more details refer to [45, 46, 47, 48].

Usability concerns the design of a system with the user's psychology and physiology in mind. The end-result should be a system that is easy to learn, efficient to use and promotes user satisfaction (refer also to Section 2.2.2). Based on cognitive psychology, ergonomics, and empirical results, *descriptive* or *predictive* models of human computer interaction have been devised to help designers analyze interaction and build efficient interfaces.

A fundamental empirical result concerns the limited capacity of working memory. Human memory consists of sensory buffers, short-term or working memory, and long-term memory. Short-term memory can be accessed rapidly but it also has a limited capacity. Miller in his classic article "The Magical Number Seven Plus or Minus Two" [49] found that human working memory can hold 7 ± 2 chunks of information. This finding has direct implications in the design of interactive systems; a complex interface may overload the short-term memory, resulting in poor and inefficient user interaction.

Another well-known result concerns information processing in choice reaction tasks. Reaction time increases logarithmically as the number of alternatives increases (Hick-Hyman law), while movement time to a target (ignoring initial reaction time) increases logarithmically with distance to target and inverse logarithmically with target's width (Fitts' law). These rules apply, for example, to the design of menu hierarchies. Another result concerning multimodal interaction is the "visual dominance" effect [50, 30], which states that "if percepts of varying modalities are of the same relative intensity, then information gathered via vision tends to have greater influence on perception, as compared to other modalities". The visual dominance effect applies, for example, to multimodal interface design and audio-visual speech recognition.

An early example of a descriptive/predictive model is the *Human Model Processor* [1] which is a simplified model of human processing when interacting with computer systems (see Fig. 2.1). The model comprises of three subsystems, namely: the *perceptual system* handling sensory stimulus from the outside world, the *motor system* that controls actions and the *cognitive system* that provides the necessary processing to connect the two [45]. "It is a synthesis of the literature of cognitive psychology of that time and sketches the framework around which a cognitive architecture could be implemented" according to [51]. It is also the basis of contemporary cognitive architectures that are used in HCI, such as EPIC (Executive Process Interactive Control), and ACT-R/PM (Adaptive Control of Thought-Rational/Perceptual Motor) [51]. The *Goals, Operators, Methods and Selection* (GOMS) rules model analyzes routine human computer interactions and is used to make quantitative predictions about execution time for a particular task. The interested reader may refer to [1] for more details.



(b)

Figure 2.1: Human model processor as described in [1]

2.2.2 User Interface Design

The design of interactive systems follows the iterative process of the software life cycle, e.g., an iterative waterfall model, consisting of stages such as requirements specification, architectural design, implementation, and testing. For interactive systems, however, requirements specification is much harder to accurately define in advance. In order to achieve a highly usable system, designers continuously enhance the interface based on the feedback that evaluators provide on early prototypes. Various studies have shown that for interactive systems a large part of the development resources (up to 50% of total) are spent on the user interface. The design of interactive systems is not only highly demanding in terms of development efforts; it should also support usability to a high level in order to be successful.

When designing interactive systems, the notion of *usability* is central to the design process. ISO 9241 standard (Ergonomics of Human System Interaction) defines *usability* in terms of

three attributes: the "*effectiveness*, *efficiency* and *satisfaction* with which users achieve specified goals using the system". According to this definition, effectiveness is the accuracy and completeness in achieving the user specified goals using the system. Efficiency relates to the resources expended in relation to the accuracy and completeness of goals achieved. Satisfaction is a measure of the user's comfort and acceptability towards the system. It is common to use objective metrics such as "task completion" and "time to completion" to measure the effectiveness and efficiency of a system, respectively, while satisfaction is measured using subjective metrics, e.g., evaluation questionnaires.

To make the development of interactive systems easier and ensure high levels of usability, a designer should create the interface with usability principles in mind. Since usability principles are essential but rather abstract properties, designers usually try to follow specific design rules such as user interface (UI) guidelines and standards. Applying design methodologies that promote usability such as "usability engineering" [45], using appropriate software toolkits and applying efficient designs such as the Model-View-Controller (MVC) architectural principle, are essential to successful design of interactive systems. The rest of this section focuses on usability principles, design rules, and the MVC paradigm.

Usability Principles

In [45], the authors list attributes that support usability under three different categories:

- Learnability: the ease with which new users can begin effective interaction and achieve best performance.
- Flexibility: the multiplicity of ways the user and the system exchange information.
- Robustness: the level of support provided to the user in the process of achieving his goals.

Learnability encompasses attributes such as *predictability*, i.e., determining the effect of future actions based on past interaction history, *familiarity*, i.e., the extend to which a user's knowledge or experience with other interactive systems can be applied when interacting with a new system, and *consistency*. Consistency, i.e., the likeness in behavior arising from similar situations, is the most important principle in user interface design, because users rely on interface consistency to carry out specific tasks. According to [52], in order to support internal consistency, the same conventions and rules for all aspects of an interface screen (GUI or web pages) should be followed, such as the organization, presentation, usage and location of screen components.

Related to flexibility is *customizability*, which refers to modifiability of the interface by the user (adaptable interfaces) or the system itself (adaptive interfaces); refer to section 2.5 for more

information on adaptive interfaces. Related to robustness are *observability* and *transparency* that allow the user to monitor the internal state of the system, and *recoverability* that allows the user to easily recover from errors, for example through "redo" and "undo" actions.

Design Rules: Guidelines and Standards

Design rules restrict the space of design options and prevent the designer from pursuing options that would likely result in less usable systems [45]. Design rules are often supported by psychological, cognitive or ergonomic theory, areas that the designer (typically a software engineer) might not be familiar with. Following design rules such as guidelines, style guides (e.g., look and feel for GUIs) and standards throughout the design process, is essential for the usability of the interactive system. An extensive list of guidelines for a broad range of topics such as data entry, screen design, graphics/icon design and proper use of GUI components exist in literature, e.g., [52]. An example of standards is ISO 9241, a multi-part standard covering many aspects of interaction such as menu and form-filling dialogues.

The MVC Design Paradigm

Every user interface application consists of three major parts: (i) the *model* or *application semantics*, (ii) the *view* or *interface implementation* and (iii) the *control* or *application logic*. The term model-view-controller has been extensively used in the HCI literature. The separation of these three key components both architecturally and in the system design process is known as the MVC paradigm [53]. The MVC paradigm goes beyond the traditional GUI community and extends also to state-of-the-art multimodal systems, see for example the latest W3C recommendations in [54]. Consider for example a spoken dialogue system: the term "model" could refer to the modules that perform speech understanding, i.e., turning speech into concepts, the term "control" could refer to the application manager that determines the next state of the interaction and the term "view" could refer to the implementation of the communication goals via the spoken dialogue interface.

2.2.3 Evaluation

An important step during the development of an interactive system is the evaluation of the interface design and implementation [45]. Although in practice evaluation takes place as the last step of the development process, ideally it should be integrated as soon as possible in order to provide feedback during the design life cycle, i.e., evaluation should be integrated in the *iterative design process*. Evaluation helps to ensure that the system functionality fulfills the intended requirements of the various tasks supported. It also allows the system designer to measure the effectiveness of the system in supporting the tasks, by measuring user performance.

Finally, evaluation helps ensure that certain usability principles and guidelines have been followed, while common usability problems have been avoided, resulting in high levels of user satisfaction.

A variety of evaluation methods exist to test the design and the implementation of an interactive system. Methods that focus on the design can be used before implementation takes place to identify and eliminate possible interface related problems early in the design cycle. In *heuristic evaluation* [55, 56], usability criteria called *heuristics*, which are based on usability principles and guidelines, are used to identify usability problems, debug and effectively alter the design.

Actual testing of the interfaces with users (user-centered evaluation) includes a number of methods such as *experimental evaluation* and *query methods*, which use objective performance metrics and user satisfaction *subjective metrics*, respectively. With experimental evaluation, performance of different design options can be computed in order to decide the best alternative, e.g., "is interface A better than interface B"? Objective metrics such as speed, number of errors, task completion, are computed for the various system configurations and are statistically analyzed to determine the best system, for examples refer to [39, 40]. Alternatively, query methods can be used to elicit direct user feedback using either interviews or questionnaires. Query methods are simpler to carry out and analyze, and can provide useful information if well designed. Note, however, that the elicited information is subjective and may be less accurate than for objective evaluation methods. Finally there is *participatory design* where users are not only involved in the evaluation phase of the system, but are also included as active participants during the design phase.

2.3 Interaction Modalities

Although GUIs have been the dominant user interface technology for the past two decades, today's computing platforms (ranging from mobile devices to large wall displays) call for new, more natural and efficient ways of interaction. Recently, there has been much interest in investigating alternative input/output interaction modalities that go beyond the traditional keyboard and mouse input, and text and graphics output. Such modalities may include speech, eye-tracking and haptics. In addition, various input/output devices [57, 58], such as glove mounted devices [59] and sensors ranging from accelerometers to GPS (global positioning system), open the door to new interfaces and applications.

In this section, interaction modalities related to this thesis are reviewed, namely GUIs, speech and gestures. First, GUIs are briefly examined and concepts such as the desktop metaphor and direct manipulation are presented. A short history of the development of GUIs is given along with promising future directions such as zooming user interfaces. Speech is

considered the most natural form of communication and although there are several limitations in speech recognition technology much progress in both system architectures and applications have been achieved in recent years. Spoken dialogue systems technology is examined in detail and example applications are provided. The use of speech at the interface level and the comparison with other modalities such as GUIs is also examined. Finally gesture based interfaces that have recently gained much attention due to emergence of touch based devices such as mobile phones, tablet PCs and large wall displays are also discussed.

2.3.1 Graphical User Interfaces

Following the command-line and text-based interfaces, graphical user interfaces emerged and eventually dominated the past two decades. The Xerox Alto and Star (1981) [60] was one of the first personal workstations having significant local processing power and memory, networking capabilities, a high resolution bit-mapped display, a keyboard and a mouse. The user interface incorporated windows, menus, scrollbars, mouse control, and selection mechanisms (*WIMP* interface - windows, icons, menus and pointers) and views of abstract structures all presented in a consistent manner. These systems introduced several innovative concepts found in today's personal computers: the *desktop metaphor*, *direct manipulation* and WYSIWYG (what you see is what you get), where a user sees and manipulates on screen a representation of a document that looks identical to the eventual printed one. By offering a rich set of graphical elements (widgets) upon which users perform actions (direct manipulation), GUIs are easier to learn and operate compared to their command-line counterparts.

GUIs are the dominant interface technology in part due to the high bandwidth they provide on the output side. In general, information can be better organized and presented to the user using graphical output compared to other modalities. Thus GUIs today are used not only in desktop computers but also in a variety of other platforms such as intelligent information kiosks, portable and mobile devices and automated teller machines (ATMs). On the input side, GUIs use for selection, pointer devices such as mouse on desktop computers or pen devices on portable systems with touch-screens. Some touch-screens support touch or *multi-touch sensing* allowing input through one or more fingers which is considered more natural than using a pen. A recent example is the Apple iPhone[2] that supports various gestures, e.g., the user can zoom in/out by spreading the two fingers closer together or farther apart. For text input, desktop computers use keyboard, while portable and mobile devices use methods such as miniaturized physical keyboards, keypads, virtual keyboards, or various handwriting recognition methods such as graffiti [2] input.

Some recent advances in GUI interfaces include 3D interfaces and Zooming User Interfaces

[2] http://www.apple.com/iphone/

(ZUI). With the advent of powerful graphic processing power, 3D desktop environments have emerged as a replacement to their 2D counterparts. Other notable efforts include the Croquet project[3], a free software platform and a network operating system for developing and delivering deeply collaborative multi-user on-line applications. ZUIs extend GUIs by laying out information elements on a infinite virtual surface instead of windows. The user can pan across the surface and zoom into areas of interest. Examples of ZUI applications are mapping applications such as Google earth/maps and desktop-like environments such as the Sugar ZUI found in One Laptop Per Child initiative[4]. ZUIs are especially promising for mobile applications where screen real estate is limited.

2.3.2 Speech Modality

Speech is the most natural form of communication among humans, but it has several limitations when used in HCI. Although speech recognition technology has been studied actively during the past decades and highly sophisticated recognizers have been constructed, machines are far from matching human speech recognition performance, especially in adverse recording conditions.

A second hurdle is the complexity of spontaneous human speech communication because it may contain a lot of ungrammatical elements such as hesitations, false starts and repairs. Finally, another issue is that people are used to talking differently to computers than to other people and often alter their speaking styles when talking to machines.

Spoken Dialogue Component Technology

Spoken Dialogue Systems (SDS) form the majority of speech applications. The main components of an SDS are: speech recognition, natural language understanding (NLU), dialogue manager (DM), response generation and speech synthesis. Next a brief review of these technologies is presented. For more details refer to [61, 62, 63, 64, 65, 66].

Automatic speech recognition (ASR), is the process of transforming a spoken utterance into words. The audio signal is digitized and is transformed into a series of acoustic vectors $Y = y_1, y_2, \ldots, y_t$ (*feature extraction*) at a fixed rate [64]. To determine the most probable word sequence $\widehat{W}$ given the observed signal Y the following Bayesian formulation is used:

$$\widehat{W} = \arg\max_w P(W|Y) = \arg\max_w P(W)P(Y|W) \tag{2.1}$$

where $P(W)$ is the a priori probability of observing W, determined by the *language model*, and $P(Y|W)$ is the probability of observing the sequence Y given a word sequence W, determined

[3] http://en.wikipedia.org/wiki/Croquet_project/
[4] http://wiki.laptop.org/go/HIG/

by the *acoustic model*. For acoustic modeling, each phone (or sequence of phones) is usually modeled by a *Hidden Markov Model* (HMM). An HMM can be though as a random generator of acoustic vectors which consists of a sequence of states connected by probabilistic transitions. The language model provides a mechanism of estimating the probability of a word w_k in a utterance given the preceding words $w_1 \dots w_{k-1}$. This is usually achieved by using N-grams, which assume that w_k depends only on the preceding N-1 words. Due to data sparseness problem, models with N equal to two (bigrams) or three (trigrams) are used in practice.

The output of the speech recognizer is analyzed by the Natural Language Understanding (NLU) component to derive meaning representations that will be used by the Dialogue Manager (DM). This involves syntactic and semantic analysis to elicit attribute-value pairs in a symbolic representation. A grammar that consists of hand crafted rules is sometimes used to produce a complete parsing of grammatically correct sentences. Techniques such as *robust semantic parsing* are often used instead, where only the essential items of meaning are extracted from the text.

The dialogue manager is responsible for the communication flow with the user. At each turn, the DM determines if sufficient information has been elicited in order to complete the user's request, e.g., information seeking. The DM is often implemented as a finite state machine (FSM) with conditions residing on the arcs and system actions residing on the nodes of the FSM. Various techniques are used for resolving errors and ambiguity in user input, such as implicit or explicit verification/confirmation.

Response generation deals with the construction of the message that will be sent to the user. Although complex natural language generation (NLG) methods can be used, usually simpler methods such as template filling (insertion of retrieved data into predefined slots in a template) are the norm. The message is then send to the text-to-speech synthesis (TTS) component, which first analyzes the text message (text to phoneme conversion) and then generates the speech signal (phoneme-to-speech conversion).

Speech as an Input/Output Modality

Speech and GUI interfaces have been extensively studied and compared in the literature, e.g., [8, 67]. With GUIs everything the user wants to do at any given time must be presented at the screen, while speech interfaces lack visual information and require users to memorize all meaningful information. In addition, the sequential nature of speech loads the short-term memory and takes up the linguistic channel, which makes speech interfaces unsuitable for some tasks.

As an output channel, speech is too slow because of its sequential nature, while GUIs convey information in parallel thus making them suitable for presenting a large amount of information.

Speech output may be more appropriate for grabbing attention and offering an alternative feedback mechanism to the user, rather than conveying a large amount of information [67].

Spoken interaction may be faster when users immediately say what they want to achieve without going through menu hierarchies. Spoken messages may also be more expressive and convey richer information compared to GUI actions, such as the selection of similar objects among a large number of them. However the freedom and efficiency that speech gives to user, makes speech harder for the computer to handle. It is also hard for users to know the limitations of what they can say and how to explore the set of possible tasks they can perform [67].

Finally, users interacting with speech interfaces do not have the same feeling of control usually offered by GUI interfaces. This is because speech input may be *inconsistent* due to recognition errors, i.e., the recognition result may be different for the same sentence spoken twice. Handling speech errors efficiently is a key issue for successful speech applications. Well designed spoken dialogue systems or the use of extra modalities in multimodal systems can alleviate these problems and allow for efficient and natural speech interaction.

How Speech Recognizer Features Affect Speech Applications

The capabilities and features of a speech recognition system can affect the design and interaction of a speech application [68]. Vocabulary size and recognition grammars characterize the interaction possibly better than other properties. For example, it is possible to construct a speech-only e-mail application with a dozen of words, but for building an information retrieval system at least a few hundred word vocabulary is needed. The possibility to change or dynamically construct vocabularies and grammars also affects interaction; e.g., allow the system to be context-sensitive and use user profiles with personalized recognition grammars.

Communication style can vary from speaker-dependent, discrete, read speech to speaker-independent, continuous, spontaneous speech. Speaker-dependent or adaptive models are suitable for some applications, e.g., dictation, while speaker-independent models are the norm. Although with current recognizers there is no need to speak in a discrete manner, it usually helps if words are pronounced clearly and properly. Most SDSs have to deal with various degrees of spontaneity in speech input, which is still a challenge for state-of-the-art speech recognition systems. Finally, capabilities like barge-in that can be used to interrupt the system output can influence the design and allow the system to generate longer and more informative responses.

Usage conditions can vary from clean to hostile environments, and low (public mobile phone usage) to high quality channels (close-talking microphones). Even with state-of-the-art recognizers, performance can dramatically suffer if usage conditions do not match recognizer training ones. This is usually compensated by using different acoustic models for each condition.

Speech Applications

Early speech applications included telephone-based interactive voice response (IVR) systems that used speech output and telephone keys for interaction. Such applications were designed to replace human operators. In the past decade, numerous spoken dialogue systems have been designed and deployed that fully automate simple interactive tasks usually performed over the telephone. Example applications that have dominated the field are information services (timetables, weather forecasting, e-banking), e-mail applications, ticketing and voice portals. Today's systems are fairly sophisticated and include state-of-the art recognizers, natural language understanding and response generation components, but still integration and interface design are the important factors for building successful applications [66, 68]. Recently, systems with more advanced natural language and spoken dialogue capabilities have been deployed for customer service applications, e.g., for telephony, cable TV[5], software retailers. Such systems automate complex interactions with complicated call-flows, but often run into miscommunication or other problems. When the system detects such problems a human operator is used as a bail-out.

Desktop applications such as dictation systems and command and control applications have also been deployed. Dictation systems[6] are popular for special user groups. Command and control applications usually control existing graphical applications, without using (or in conjunction with) mouse/keyboard, which can be very useful for mobile devices such as personal digital assistants (PDAs). Other spoken dialogue applications include automotive applications, e.g., navigational aids, gaming, and human-robot interaction.

2.3.3 Gestures

Gesture based interfaces [69] augment traditional graphical user interfaces, which are based on direct manipulation, by incorporating 2D and 3D gestures like manual gestures, head and body movements. Although people may occasionally use gestures as the only means of communication, e.g., to indicate disagreement by a head or hand gesture, in most cases gestures occur along with other modalities such as speech, as demonstrated in Bolt's "Put-That-There" prototype [18]. Apart from *deictic gestures*, *iconic gestures* that refer to objects or actions by describing them visually using familiar representations and *symbolic gestures*, e.g., thumbs-up, are also exploited in typical gesture interfaces. Gestures may be used to specify attributes, e.g., location, size, category of actions, or commands, e.g., creation, confirmation, selection.

Devices to capture 2D gestures include touch sensitive displays, digitizing tablets and light pens. Recognition of 2D gestures is either *template-based*, in which case gesture recognizers

[5] http://www.speechcycle.com/
[6] http://www.nuance.com/naturallyspeaking/

compare input patterns with prototypical templates to choose the best matched one, or *feature-based* where features extracted from the stream of input coordinates are first processed and then classified to a gesture class. 3D gestures such as hand and head or body movements can be incorporated either in active or passive mode. In *active mode*, dedicated devices are used, such as position trackers and sensing data gloves. In *passive mode*, user input is unobtrusively monitored using one or more cameras and computer vision algorithms are used to segment and classify the image data. In passive mode, no intrusive devices are necessary but recognition is much less accurate compared to the active approach. For a review of gesture-based interfaces refer to [69].

2.4 Multimodal Interfaces

Multimodal systems (or multimodal input/multimedia output systems) employ two or more input modalities and presentation media to interact with the user. Examples of input modalities include keyboard, pointing devices (mouse, pen), speech, eye-gaze, gestures, haptics. Examples of presentation media include text, audio, images, video, animation. Multimodal interfaces pose two fundamental challenges namely: the combination of multiple input modalities, known as *the fusion problem*, and the combination of multiple presentation media, known as *the fission problem*. "Optimal" solutions to the fusion and fission problems can significantly improve performance of multimodal systems over their corresponding unimodal constituents, both in terms of *efficiency* and *user satisfaction*. The improvement in performance of a multimodal interface over the "sum" of its unimodal parts is often referred to as *multimodal synergy*.

The most common multimodal interface is that of the personal computer that combines, since the 80's, keyboard entry with a pointing device (usually mouse). Although the two input modalities can typically be used only sequentially, the fundamental concepts of fusion, fission and synergy are still very relevant. Extensive experimentation (as well as cognitive considerations) have determined the rules and guidelines for the design of graphical user interfaces (GUIs). These guidelines are related to the fusion and fission problems. For example, guidelines about when and how to use "text entry" vs "pull down menus" are related to the keyboard and mouse fusion problem, while recommendations on the combination on text and graphics are related to the fission problem.

Recent bibliography on multimodal interfaces and systems focuses on novel interaction modalities, such as speech, gestures, eye-gaze or haptics. New modalities introduce new opportunities and challenges, e.g., speech interfaces are more natural but are prone to recognition errors. According to Oviatt [14], multimodal interfaces should be a paradigm shift away from conventional WIMP interfaces towards more flexible, efficient and powerfully expressive means of human computer interaction. Investigating new interaction modalities and concurrent mul-

timodal interaction are active research directions in the field. Next the basic concepts of multimodal interaction, fusion, fission and design are presented.

2.4.1 Multimodal Interaction

In [70], multimodal interaction is categorized into: (i) *sequential* when at a specific point in the interaction only one input modality in active, e.g., keyboard and mouse on a typical desktop interface, (ii) *simultaneous* or *concurrent* when "simultaneous" input is received from multiple modalities but can be treated separately by the fusion module, e.g., eye-gaze combined with keyboard input, and (iii) *composite* (or synergistic [21]) when "simultaneous" input from multiple modalities has to be processed as a compound entity by the fusion module, e.g., the synchronized speech and gestural input "Put that *[gesture pointing]* there *[gesture pointing]*" from Bolt's famous demo [18].

Sequential multimodality is by far the most common in human-computer interaction. With the advent of "novel" modalities, such as eye-gaze and speech input, it is becoming increasingly common to have simultaneous input from different modalities. Composite multimodal interaction is especially relevant for a range of applications such as map navigation, course plotting etc. Although the basic principles of fusion are the same for all three interaction modes, the fusion module becomes more complex when allowing for simultaneous and (more so for) composite input.

According to [70], multimodal interfaces can alternatively be categorized into *supplementary* or *complementary* depending of whether all input and output tasks can be carried out by every modality or not. Supplementary interface design is the rule, because it results in a consistent user interface and improves usability. However, for modalities with limited interaction scope, e.g., eye-gaze or gestures, or for interaction tasks where one modality is clearly superior (in terms of efficiency) a complementary approach might be taken. Finally, *symmetric multimodality* [71] refers to interface design that has the same modalities available for both input and output.

2.4.2 Fusion Techniques and Data Integration

Multimodal systems require fusion in each of the three layers of the MVC paradigm, namely at the data (semantic fusion), at the view (interface fusion) and at the control level. It is customary in the literature for the term fusion to refer to *data fusion* or *semantic fusion*. However, *interface fusion* or *modality fusion*, i.e., the problem of fusing (or blending) the modalities at the interface level, is an equally important problem for interface design. Fusion at the control level is usually tackled by designing a multimodal application manager that manages all modalities. In fact, if the MVC paradigm is followed the application logic should

be modality-independent and little integration is needed. Next the problems of data fusion and interface fusion are discussed.

Data fusion is usually categorized as *early fusion*, or *late fusion* [72]. The most common example of early fusion, also known as *feature-level fusion*, is the combination of the audio and video feature streams in audio-visual speech recognition. As discussed in [14], multimodal systems based on late fusion integrate common meaning representations derived from different modalities into a combined final interpretation. This requires a common meaning representation framework for all available modalities and a well-defined operation for integrating the partial meanings. Late fusion is more common in multimodal systems.

Depending on the multimodal interaction style (sequential, simultaneous or composite), the internal data representation, and the point of integration in the semantic chain, different fusion algorithms can be implemented. For sequential or simultaneous multimodal interaction the semantic information acquired from each modality can be processed more or less independently and thus late integration is the rule. The semantics extracted from each input stream are combined, often using a probabilistic framework, to resolve ambiguous or conflicting input. For composite multimodal interaction, integration typically occurs earlier in the process because input from various modalities has to be processed jointly. One popular approach is to design multimodal semantic grammars. For example, to handle composite speech and pen input a three-tape finite-state machine was proposed in [73].

According to [68, 21] one can consider fusion earlier or later in the semantic chain, i.e., at the *lexical, syntactic* or *semantic* levels. Lexical fusion is used when primitives, e.g., words, are mapped to application events. Syntactic fusion synchronizes different modalities and forms a complete representation. Semantic fusion represents functional aspects of the interface by defining how interaction tasks are represented using different modalities. Most advanced multimodal systems perform syntactic or semantic fusion.

Fusion also depends on the internal data representation. Application data can be represented in structures such as *frames* [74], *feature structures* [75] or *typed feature structures* [76]. Frames represent objects and relations as consisting of nested sets of attribute/value pairs, while feature structures go further to use shared variables to indicate common substructures. Typed feature structures are pervasive in natural language processing, and their primary operation is unification, which determines the consistency of two representational structures and, if consistent, combines them. As the data structures used become more complex and interdependent, the complexity of the fusion algorithm also increases. Various integration techniques have been devised: *frame-based integration* techniques use a strategy of recursively matching and merging attribute/value data structures (e.g., [77]) while *unification-based integration* techniques use *logic-based* methods for integrating the *partial meaning fragments*. Unification-based architectures have been applied to multimodal system design [78, 79].

Some important unification-based integration techniques include feature-structure and symbolic unification. *Feature-structure unification* is considered well suited to multimodal integration, because unification can combine complementary or redundant input from both modalities, but it rules out contradictory input. *Symbolic unification* when combined with statistical processing techniques results in *hybrid symbolic/statistical* architectures that achieve very robust results.

Recently, with the advent of the semantic web, there has been much interest in using semantic mark-up languages such as DAML+OIL[7] to represent application semantics and perform discourse modeling. Such mark-up languages can be combined with reasoners that can perform automatic inference and consistency checking; refer to [71] for an example of a multimodal dialogue systems that uses these tools.

Example: QuickSet Fusion Mechanism

As an example of how fusion and semantic unification of two recognition based modalities is achieved in multimodal systems, the QuickSet multimodal system is described next [80, 13]. QuickSet supports both speech and pen (gesture) input. For pen input each stroke is time-stamped and an internal data structure holding the x,y coordinates is sent to the gesture recognition component. The recognizer produces a N-best list of possible interpretations, each associated with a probability. These signal-level interpretations are then sent to the natural language agent to create a gestural parse N-best list before being integrated with the parallel speech interpretation. Like gesture processing, the speech recognizer generates an N-best list of interpretations, each associated with a probability estimate. These signal-level interpretations then are filtered by the natural language parser, which forms a spoken language N-best list.

To interpret a whole multimodal command, the time-stamps for speech and gestural input are compared by the integrator. Based on synchronization patterns typical of speech and pen input, an integration rule is applied to these time-stamped signals. The integrator will combine speech and pen signals and attempt to process their multimodal meaning when either a temporal overlap between signals exist or a speech signal begins within four seconds of the end of gesture (sequential signals). If synchronization rules permit joint processing, semantic unification will take place. The common meaning representation for speech and pen input, represented as typed feature structures are combined into a single complete semantic interpretation if compatible. Each item in the N-best list for both speech and pen input is processed by the unification parser to produce the feature structure representations which are combined during multimodal integration to produce full representations. The combined interpretations that do not unify, are left out while the remaining ones are assigned probability estimates (by

[7] http://www.daml.org/

combining the unimodal scores) to build the final multimodal N-best list.

2.4.3 Multimodal Interface Fusion and Fission

Interface designers can force or imply to the user what modality (or combination of modalities) should be used at each point of the interaction. For example, in GUI design, "radio buttons" and "combo boxes" imply mouse (or pen) input, while text fields imply keyboard input. This is also true for "novel" interaction modalities, e.g., for speech and pen interfaces a "click-to-talk" interaction mode biases the user towards the pen modality Designing interfaces that guide the user towards using the "optimal" input modality mix is the problem of *multimodal interface fusion* or fusion at the interface level. Few guidelines exist for selecting the "optimal" mix of modalities [23, 22]; these guidelines are mostly based on efficiency considerations. Overall, multimodal interface designers should respect all available input modalities, offer the user the flexibility to select (or override the default) input modality, and blend modalities having cognitive, efficiency and user satisfaction considerations in mind. The end goal is to create a truly multimodal experience, a user interface that maximizes *synergies* among the input modalities, by improving efficiency and robustness (error-correction capabilities).

The problem of *multimodal fission* is symmetric to that of fusion. Fission is the process of communicating an internal representation of the system to the user, via the co-ordinated action of multiple output modalities and output media. Selecting the appropriate output media, their relative importance for each communication act, and, most importantly, co-ordinating the presentation in time and space are some of the important issues in fission [68, 81]. Fission has not attracted as much research interest as fusion, and often ad hoc solutions are adopted for the fission problem. According to [68], most of the work in this area has been done by the multimedia research community, e.g., in the area of automated multimedia systems [82]. Such systems often focus more on how to render the information for different media and devices, rather than investigating the "optimal" blending of media or the selection of appropriate output modalities.

According to [81], fission algorithms should respect the MVC paradigm and separate communication acts from the interface implementation of these acts. In addition, there should always be output presentation for internal system representations (system states) and vice versa. This later principle is referred to as "*no presentation without representation*" [83]. Co-ordination and synchronization of the various output modalities is also an important problem. For example, for *embodied conversational agents* (also known as talking-heads) [84, 8] system output is presented via both audio and video streams that have to be synchronized to achieve a realistic effect (lip-syncing). Overall, selecting the appropriate mix of media to visualize system information and communicate with the user is an important open research problem

that requires contributions from researchers, technologists and artists.

2.4.4 Multimodal Interaction Patterns and Usage

An important issue when implementing multimodal systems is the choice of interaction style (simultaneous vs. sequential), but also the internal implementation of input and output events in a synchronous or asynchronous manner. According to [70], synchronization of input events can occur instantaneously at the *event* level, at the *field* (concept) level or at the *form* (groups of concept) level. User behavior can serve as a guide for the selection of interaction style and synchronization granularity. In [85], the authors found that users adopt either a simultaneous or a sequential integration pattern during speech and pen multimodal input (70% simultaneous and 30% sequential). Their findings also show that user's dominant integration pattern is predictable early and remains consistent (89-97%) over time.

As discussed also in [86], multimodal interfaces may have many advantages: error prevention, robust user interface, easy error correction or recovery from errors, increased communication bandwidth, flexibility and alternative communication methods. Disambiguation of error-prone modalities is the main motivation for using multiple modalities in many systems. Multimodal interfaces offer improved robustness to errors due to both user behavior and system support [13]. During the evaluation of the QuickSet system, it was found that users tend to use simplified language (briefer utterances, fewer referring expressions) when interacting multi-modally than when interacting using a unimodal spoken dialogue interface. It is also reported that users tend to use the less error-prone modality in a certain context (error avoidance) and switch modes after system errors, thus facilitating error recovery. As far as system support is concerned, temporal, semantic and other constraints can be exploited to rule out candidates. This *mutual disambiguation* and *synergistic error correction* features make multimodal interfaces more robust compared to unimodal ones.

It should be noted, however, that multiple modalities alone do not bring these benefits to the interface: currently there is too much hype in multimodal systems, and the use of multiple modalities may be ineffective or even disadvantageous in some cases [87]. Following good system and interface design principles is essential for building successful multimodal applications.

2.4.5 Multimodal Applications

Numerous multimodal systems have been reported in the literature, a large number of which are cited in [14]. In [69], multimodal applications are categorized according to application domain, input/output modalities and fusion type. From the historical perspective, multimodality offers promising opportunities, as presented in Bolt's "Put-That-There" system [18]. Bolt's

system combined pointing and speech input as a natural way to communicate; gaze direction tracking was added in a later prototype and used for disambiguation. Other early systems used speech input along with keyboard and mouse in an effort to support better complex visual manipulation. Technology advances in late 1980s allowed speech to become an alternative to keyboard, leading to map and tourist information systems such as CUBRICON [88] and Georal [89].

Bimodal systems that combine speech and pen-input, or speech and lip-movements emerged in 1990s leading to work on integration and synchronization issues and the development of new architectures to support them. Speech and pen-input (2D or 3D gestures) involving a large number of different interpretations beyond pointing have advanced rapidly both in research, e.g., Quickset [80], and commercial systems. Speech and lip movement systems exploit the detailed classification of human lip movements (visemes) and offer speech recognition robustness in noisy environments. Lip movement is also used in coordination with text-to-speech output in animated character systems (*talking heads* or *speaking agents*). Examples of such systems include include the Rea system [84], KTH's August, Adapt and Pixie systems [8]. These systems use audiovisual speech synthesis and anthropomorphic figures to convey facial expressions and head or body movements. Systems with animated interactive characters have also been constructed [82, 90]. These systems mainly focus on multimedia presentation techniques and agent technologies. Information kiosks (*intelligent kiosks*), such as SmartKom, use speech and haptics to provide an interface for users in public places, e.g., museums. Animated characters may have a strong motivational impact, since they are considered as being more lively and engaging for many users [91].

As noted in [14], systems combining three or more modalities such as biometric identification and verification systems [92], which use both physiological (retina, fingerprints, face or facial thermograms) and behavioral (voice, handwriting) modalities have also been developed. There is also increased interest in *passive input modes* [14], which refer to naturally occurring user behaviors that are unobtrusively monitored by a computer, e.g., eye gaze or facial expressions. *Ambient intelligence* and blending of active and passive modes is a promising direction to this end.

2.5 Adaptive Interfaces

As computer applications are becoming increasingly complex both in terms of functionality and interface design, it is also becoming increasingly hard to build applications and interfaces that satisfy the needs of all users. For example, users have different capabilities and preferences when multiple modalities are made available to them. New applications and interaction modes make user diversity even more apparent. As a result the need for *adaptation*, i.e., modification of the

data model, application control and/or application interface to the specific user characteristics, needs, capabilities and preferences, is becoming increasingly apparent. Adaptation has been used for a large variety of tasks and applications, often successfully, improving the interaction efficiency and the user experience. However, despite the promise that adaptive interfaces hold, designing interfaces that are adaptive and also appear consistent to the user is a challenging task. In addition, adaptive interfaces are complex and the consequences of adaptivity on the user experience is sometimes unpredictable. As a result, system designers often opt for *adaptable interfaces*, i.e., interfaces that can be modified/adapted explicitly by the user, or limit the functionality of the adaptive algorithms.

2.5.1 A High Level View of User Adaptive Systems

The literature on adaptive interfaces is rich and very diverse, as researchers with different research backgrounds attack the problem. A number of definitions for adaptive systems can be found in the literature [93]. The definition of a user adaptive system given in [94] follows: "An interactive system that adapts its behavior to individual users on the basis of processes of user model acquisition and application that involve some form of learning, inference, or decision making." Thus, in a user adaptive system, the system gathers information about certain aspects of user interaction (*user model acquisition*) and performs learning and/or inference based on that information in order to create or update a *user model*. The system then applies the user model in order to determine how to adapt its behavior to the user (*user model application*). Although much of the adaptation literature focuses on user adaptation, there are also other aspects of adaptation, e.g., adaptation (or updating) of the system model or adaptation of the user interface that are equally important (refer to Section 2.5.2).

User model adaptation algorithms can be categorized based on the ways in which information about users is acquired. As discussed in [94], information about users can be acquired either as *explicit* input to the system or in a *implicit* way. In the first case, the system requests information relevant to the adaptation that may be difficult to elicit otherwise, e.g., location, user's age, topics of interest. In the second case, the system collects relevant naturally occurring actions or past interaction information and exploits it in the adaptation process. Examples include user location information extracted using GPS-capable mobile devices, or emotion detection, such as anger or attention. Often a pattern recognition system is used to extract this information leading to *unsupervised adaptation* algorithms, e.g., emotion recognition.

Another way to categorize adaptation is based on the learning, inference and decision making algorithms used, i.e., model acquisition and application. According to [94], these adaptation algorithms can be categorized into classification algorithms that employ no general knowledge about users and goals, and decision theoretic methods, e.g., Bayesian networks. Classification

methods range from simple ones, such as naive Bayes, to more complex ones, such as advanced probabilistic classifiers, decision trees, and neural networks. For example, the SwiftFile system [95] classifies incoming email messages to user folders and uses text classification methods from the information retrieval field for archiving.

Decision-theoretic systems explicitly define models of interaction, using tools such as Bayesian Belief Networks (BBNs). The models incorporate variables, for which the system has only an uncertain belief to begin with, and are connected in a probabilistic network in which the relationships among them can be interpreted as causal effects. As the system acquires new information, beliefs about network nodes are updated. For example, in the Lumiere project [96], the authors use a BBN to decide whether a user may need assistance based on user's expertise and task complexity. Other approaches include the use of stereotypes and plan recognition. A stereotype is a class of categories that a user may belong to. The system employs rules to assign users to classes and takes actions based on this classification. Plan based approaches consider user actions as steps towards achieving a certain goal; such techniques are employed in dialogue and help/tutoring systems.

User Adaptable Systems

There is a clear distinction between user adaptive and user adaptable systems. User adaptive systems implicitly adapt their user model to user preferences. An *adaptable* system, on the other hand, allows the user to explicitly tailor the interface to his preferences. A number of systems are adaptable but not adaptive. The main advantage of adaptable interfaces is that the user is in control and unwanted side-effects of adaptation can be avoided. The main drawback is that the user might not know how to effectively tune the system to his preferences.

2.5.2 Adaptation Examples in the Context of the MVC Paradigm

An alternative view of adaptivity is through the model-view-controller (MVC) paradigm. Although adaptivity may cut through all the components of the MVC model, usually the adaptation algorithm may concern only one of the three components of the system architecture. As discussed next, adaptivity may focus on the interface (view) level of the application, the data (model) or the application control (controller). Most of the discussion up to this point has been on user model adaptation. Next examples of adaptation at the interface and controller level are presented.

Adaptation at the Interface Level

An example of interface adaptation is the Smart Menus feature introduced in Windows 2000. The idea is to hide infrequently used menu items, so the user can faster access one of the most

frequent used ones. For hidden menu items, the user has to fully extend the menu in order to view and select them. Clearly, there is a trade-off between accessing frequent items faster and "missing" infrequent menu items. The effect might be frustrating or confusing to some users; the list of items in each menu changes over time, which is highly inconsistent. For users that prefer to have the full list of menu items showing at all times this feature can be disabled.

Since many applications have become too complex and feature rich, help systems are needed that can guide users to effectively use the application. Adaptive help systems can potentially detect when the user needs advice, introduce concepts or features relevant to the given situation or even directly propose a solution to a given problem. An example of an adaptive help system is the Office Assistant agent, a derivation of Lumiere research prototype [96]. Lumiere uses decision theoretic methods (Bayesian networks) to decide if help should be given spontaneously. This is done if the computed likelihood that a user needs help, exceeds a given threshold. In the Office Assistant, the decision theoretic methods have been replaced by a relative simple rule-based mechanism.

Adaptation at the Controller Level: Spoken Dialogue Systems

Adaptation has also been applied to spoken dialogue systems at the dialogue manager (controller) level to improve on existing strategies and find optimal application control policies. For example, as noted in [94], the TOOT dialogue system [97] can appropriately adapt its dialogue strategies according to different situations. If the user's speech is poorly understood the system can adopt its strategy by acquiring just one piece of information at a time and by frequently requesting confirmation. Dialogue control for error prevention and correction is a challenging problem that can be formulated as a Markov Decision Process (MDP). Techniques such as reinforcement learning can be applied to find the optimal control policies, as described, for example, in the RavenClaw system [98]. In practice, many multimodal systems implement two application control logics or interfaces, one for novice and one for expert users. Often the choice or novice or expert is left to the user leading to an adaptable (rather than an adaptive) system.

2.5.3 Usability Issues

One of the main concerns of adaptive interfaces is related to usability issues that may arise from adaptation. According to [94]: "some of the typical properties of user adaptive systems can lead to usability problems that may outweigh the benefits of adaptation." Some of these usability problems are outlined next.

"Predictability" refers to the extent to which a user can predict the effects of his actions. SmartMenus (refer above) can be thought as an example of lack of predictability, since the

low usage of a menu item (or high usage of other items) will result in the disappearance of that item. Predictability is closely associated with "transparency" or visibility. When the adaptation mechanism is invisible (not allowing the user to understand how it works), the user will be unable to understand or explain system actions. A way to achieve "controllability", a degree of control over system actions, is to allow the user to confirm any action that may have significant consequences on the interface. Distracting or irritating system behaviors are against the goal of "unobtrusiveness". For example, the distracting ways in which the Office Assistant agent is used to pop up, violates the principles of unobtrusiveness and controllability.

Usually model level adaptation is hidden from the user and does not violate basic usability principles. However, adaptation at the interface and control level are directly observable by the user and often lead to an inconsistent look and feel of the application.

2.6 Mobile Interfaces

As mobile devices are becoming increasingly ubiquitous, mobile interface design is emerging as an important research area of human-computer interaction. Designing and implementing interfaces on mobile devices, such as PDAs and mobile phones, is a challenging task because the designer has to operate under various constraints including device size, network bandwidth and power consumption. In addition, the requirements and usage of mobile devices varies significantly among users and is situation-dependent. As a result, mobile user interface design poses unique usability challenges, but also offers new opportunities, e.g., context-aware services.

Next the main differences between mobile and desktop interfaces are outlined [99]:

- **Input modality**: an important difference between mobile and desktop interfaces is that the "physical" keyboard is no longer the dominant input modality. Although keypads and mini-keyboards are still extensively used on mobile devices, alternative input modalities such as touch-screens, pen, speech, virtual keyboards are becoming increasing popular and competitive in terms of efficiency to physical keyboard input.

- **Screen size**: Mobile devices typically suffer from limited screen real estate, screen resolution and screen brightness (the later is important for achieving increased battery life). As a result, the amount of information that can be displayed using the screen is significantly decreased compared to the desktop. Alternative modalities, e.g., spoken output can be used to improve the system output communication for mobile interfaces.

- **Network bandwidth and device limitations**: Although the cost of network bandwidth for mobile devices is continuously decreasing, bandwidth remains an important factor when designing mobile interfaces. Mobile applications should adapt to bandwidth

considerations, e.g., changing signal strength. Mobile interface design is also affected by device limitations such as processing power and energy consumption. Bandwidth and processing power considerations affect architectural design decisions, e.g., if there is not enough computing power for an application to run locally on the device a client-server architecture might be used.

- **Location**: Location information is available to an increasing number of mobile devices. Location information is obtained either from cell tower triangulation or by using a GPS receiver. This information can be a valuable feature for new services that employ the user's location as a "information filter", in essence adapting the user's list of preferences to match what is locally available. An important subset of location-aware mobile applications are geographical information systems (GIS) applications that typically use GPS-capable mobile devices.

- **Environmental conditions**: A mobile device has to face variable and often extreme environmental conditions, e.g., changing levels and patterns of background noise. Mobile interfaces should adapt to new conditions and allow the user to use appropriate input and output modalities for each condition. For example, speech might be the input modality of choice for a low-noise, hands-busy task.

- **Attention and cognitive load**[8]: In contrast to the desktop, mobile users often show reduced attention (especially visual attention), because the user may be on the move or focusing on other activities. Tactile or audio feedback can be used to draw the user's attention without distracting him from his main task. In general, mobile interfaces should incur limited cognitive load, especially for applications where the user is multi-tasking, e.g., car navigation applications.

These fundamental differences between mobile and desktop interfaces call for updated design principles for mobile interface design and create new opportunities for mobile applications.

2.6.1 Mobile Interface Design: Issues and Guidelines

"Mobile Web Best Practices" [101] is a W3C recommendation that specifies best practices for delivering Web content to mobile devices. It includes a list of 60 recommendations addressing issues such as page layout and content, navigation and links, input and overall behavior. For example, images in a web page should be properly resized and rendered for the mobile device, preferably on the server side.

[8]Cognitive load can be defined as a multidimensional construct representing the load that performing a particular task imposes on the learners cognitive system [100]

Information presentation in the limited screen displays of mobile devices is an important issue. Information should be hierarchically organized in a number of displays containing short lists of options. Displays should be properly designed to minimize clutter and navigation effort. When a large number of items is required in a list, a method to navigate efficiently between the items should be made available. To facilitate scrolling through large menu items a click wheel operated in a rotational manner can be used, e.g. iPod devices.

Another important issue is the high degree of diversity among mobile devices, which makes consistency of applications among platforms and devices a challenging task. For example, PDA devices have a miniaturized desktop-like interface with pen input and various methods of text input such as virtual/physical keyboard or graffiti recognition. Most mobile phones, on the other hand, have a list-based interface that has to be operated with just a numeric keypad for navigation among screens. One solution for the deployment of an application is to use the lower common denominator as far as device capabilities are concerned; another approach is to exploit capability profiles for groups of devices.

2.6.2 Input methods for mobile devices

The diversity of mobile devices keeps growing rapidly, especially towards the high-end spectrum. This is due to the recent technological advances in mobile CPU power, cheaper memory, larger displays often with touch or multitouch support, faster network access and improvements in power consumption electronics. All these factors affect the user interface design but input devices/methods and display size and type (touch support) are the most important ones.

Although recent reports show the smartphone market is rapidly growing, the majority of mobile phones still use a limited size display, a numeric keypad that allows for alphanumeric input and a small number of special function keys along with a simple navigation button (often called 4-way key). Text input is achieved through the use of small physical keyboard called keypad, often utilizing predictive text technology such as the widely adopted T9 system. Text input is mainly used for authoring short messages (SMS) or for selecting items from large lists such as contact lists. Selecting contacts is also the main application of embedded speech recognition; it's use may be quite limited as the user usually needs to audio record the contacts she wish to recall using speech input. Although keyboard input may be very popular among young people, the small size and the learning curve required to effectively input text, proves to be difficult for more senior aged users.

PDA devices proved to be a relatively successful paradigm during the previous decade as they filled the gap between data and application centric personal computers (PCs) and the voice-centric mobile phones. The larger form factor of PDA devices, compared to mobile phones, allowed for larger screen displays that supported touch input through the use of pen

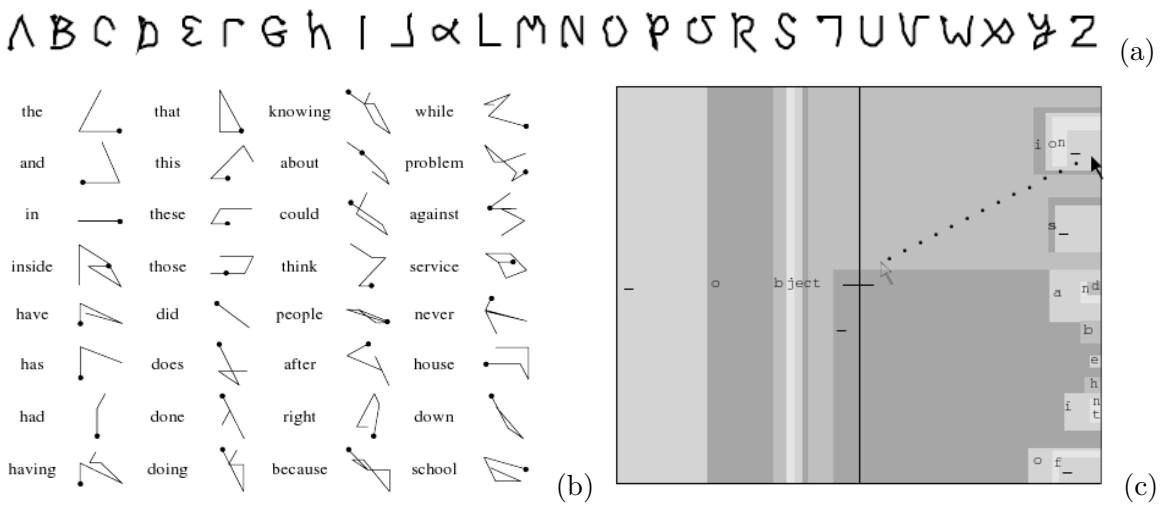


Figure 2.2: Various input methods for mobile computing (a) graffiti single-stroke letters symbols [2] (b) sokgraphs of various common SHARK words [3] (c) dasher input method - user moves cursor towards "ion" suffix to complete input of the word objection [4].

devices. It also allowed for the use of larger physical keyboards for some of these devices and the use of virtual keyboards and/or handwriting recognition technologies for text input. The use of a pointing device such as pen, comparable to mouse input in PCs and the enhanced text input methods, made possible the porting of a desktop-like applications, already familiar to many users, to such devices. From a design perspective, the replication of a desktop-like interface given certain constraints, the use of external pen device needed to operate the interface and the lack of a new interface paradigm more appropriate for mobile use, rapidly shrunk the PDA market in favor of touch enabled smartphone devices such as iPhone.

Although on-line handwriting recognition (see [102] for a review, [103] for the NPen++ system) is a relative mature technology and has been used as an input method for a relatively long time, it is not considered a very successful interaction modality. Despite the fact that it mimics the familiar to everyone "writing on a paper" metaphor, training for new users is needed to effectively use this method; and both speed and accuracy may vary considerably between users. High levels of accuracy is needed before users accept the technology. Because of the variability in written text between users may be remarkably high, such levels may be difficult to reach, requiring sophisticated methods in recognition and adaptation. One of the main problems is that some letters, or symbols, are composed of multiple strokes which are harder to recognize compared to single-strokes (separating multi-stroke to single stroks is called the segmentation problem). Graffiti [2] was a system that successfully addressed the segmentation problem. Popularized by Palm OS devices, the main idea was to use a slightly modified single-stroke alphabet (although some resemblance to the alphabet remains - see Fig. 2.2(a)) that

offered very good performance due to increased discrimination between strokes.

Hardware and virtual keyboards main advantage over online handwriting recognition systems is consistency. Miniature qwerty-like hardware keyboards may offer a solution for some users but they also add design complexity and cost to the device; due to their small size they have also been reported to be hard to use for some users (those with large thumb size). Virtual (or soft) keyboards on the other hand only cost in real estate display size. They can be operated using either pen or finger input in touch displays. They may be configurable, offering different layouts and may be used in both landscape and portrait modes. Since they are programmable they may be adaptable or adaptive or exploit techniques for faster input. ShapeWriter (previously known as Shorthand-Aided Rapid Keyboarding (SHARK)), is a keyboard text input method from IBM that offers improved performance [3]. Instead of tapping the word letters one by one, the user draws a gesture (sokgraphs a form of shorthand defined on a stylus keyboard as a graph -see Fig. 2.2(b)) that connects all the letters in the desired word.

Dasher [4] is another approach of continuous gesture input letters are arranged dynamically in multiple columns, with likely target letters closer to users cursor based on the proceeding context. The user writes letters by making gestures towards the letter's rectangle (which in turn contains more probable subsequent letters following the already written one) as shown in Fig. 2.2(c). The user doesn't need to tap (compared to virtual keyboards) or stroke (compared to handwriting) individual letters but just move the cursor (e.g. moving his finger in a touchscreen) through a path of continuously appearing letters. This method is easy to use and requires almost no training, can be used with any two dimensional pointing device (mouse/pen/eye tracking) and facilitates predictive text input by visualizing probable letter sequences; this is also the method's main drawback since the user's visual attention needs to dynamically react to the changing layout.

In the summer of 2007 Apple released the iPhone device that has revolutionized the smartphone market. A combination of elegant physical design (a large 3.5 inches display with just a home button) and a unique user interface experience utilizing multi-touch technology was the key to success. The user interface is specifically designed for multi-finger input; as finger touch interaction is much more engaging compared to pen interaction and the user interface emphasizes simplicity and consistency, it makes the device very easy to learn and use, even for new users. The movement of the fingers across the screen creates gestures with special meaning that are used for interaction. For example a single finger gesture used for scrolling a list is to drag a finger across the screen; to zoom in and out a picture or a page the user has to use two fingers on the screen and spread them apart (zoom in) or squeeze them together (zoom out). Note that although more complicated gestures could have been defined the decision to keep only simple intuitive (with as much resemblance to the physical world) gestures makes

the interaction easy and pleasant even for novice users.

2.6.3 Example Applications

Despite the limitations in screen size and processing power of mobile devices, the always-on connectivity and the increased bandwidth available in 3G mobile data networks allows for the deployment of sophisticated network based applications and services. Traditionally the majority of applications used in mobile computing was limited to the ones shipped with the device. Installing extra applications for such device was tedious and only used by advanced users. With the emergence and growing popularity of recent powerful devices that provide sophisticated developer libraries and tools, new opportunities for mobile development have emerged. Such applications can be listed in an application market and can be easily accessed, downloaded and installed by the user. Application markets for mobile platforms such as iPhone and Android have been more than successful. According to recent reports[9] there are over 100,000 applications officially available for the iPhone, and 2 billion downloads have been achieved.

An example of a mobile phone browser is the Opera Mini micro-browser[10] that is available for a wide variety of mobile phones. The browser follows a client-server architecture to overcome the limited device capabilities. Opera Mini requests web pages through proxy servers, which retrieve the web page, process it, compress it, and send it back to the user's mobile phone. The architecture and interface design emphasizes simplicity, speed and bandwidth conservation. Most importantly the Web page information is rendered on the server to match phone capabilities with very good results. Although it uses keyboard navigation it also supports a virtual mouse pointer to enhance navigation. More advanced mobile devices include sophisticated browsers based on the WebKit rendering engine which offer almost desktop-like experience. Since most of these devices also have a built-in accelerometer support, the browser can switch to landscape mode when the user rotates the device. Support for zooming in and out and panning web pages makes browsing experience very much desktop-like. Since internet searching is one of the main tasks using a browser, the Android platform has introduced voice searching in the browser; this marks one of the first efforts to use speech as a complementary input method in mobile computing.

An example of a location-aware service is Google Maps for mobile, a web mapping service that can be used both by GPS-enabled devices and by mobile phones (using the "My Location" feature, which exploits cell tower triangulation for approximate positioning). The service offers street maps, route planning (driving directions) and allows the user to find a variety of nearby

[9] http://www.apple.com/pr/library/2009/11/04appstore.html
[10] http://www.operamini.com/features/

businesses, such as theaters, restaurants and hotels. In low-end mobile devices Google Maps uses a keypad or/and pen interface; in more advanced devices such as the iPhone the user can exploit available gestures to easily move and zoom in/out, offering a desktop-like experience.

2.7 Architectures

Most multimodal systems are very complex in terms of architecture and software design, and usually mix and exploit many software architectural styles and models like the pipe-and-filter, finite-state machine, event-based model, client-server, object-oriented and agent-based ones. For example, spoken dialogue systems are usually structured either in a pipeline fashion or use the client-server model with a central component, which facilitates the interaction between other components, like the Galaxy-II architecture [104, 105]. Multimodal systems are based on even more sophisticated architectures like agent architectures [106]. Some of these architectures follow the MVC paradigm and separate the model from the control logic and the interface specification, although, in spoken dialogue systems, it is not uncommon to combine the control logic and speech interface specification into a single module, the dialogue manager. Next the differences in between GUI and multimodal architectures are examined, and some typical architectures employed in multimodal input/multimedia output systems are reviewed.

GUIs vs Multimodal architectures

As noted in [14], the design of multimodal/multimedia systems should address several challenging architectural issues not found in the design of "traditional" GUI applications. First, unlike GUI systems that assume that there is a single event stream that controls the underlying event loop, multimodal interfaces may process continuous and simultaneous inputs and outputs from parallel streams. Also GUIs assume that the basic interface actions, such as selection of an item, are atomic and unambiguous events, while multimodal systems process input modes using recognition-based technologies that are designed to handle uncertainty and entail probabilistic methods of processing. Finally, multimodal interfaces that process two or more recognition-based input streams require time-stamping of input, and the development of temporal constraints on modality fusion operations.

Multimodal Architectures and Frameworks

One popular architecture among the members of the multimodal research community is the *multi-agent architecture*, exemplified by the *Open Agent Architecture* [107] and *Adaptive Agent Architecture* [106]. As described in [68, 14], multi-agent architectures provide essential infrastructure for coordinating the many complex modules needed to implement multimodal

system processing, and permit doing so in a distributed manner. According to the authors, in a multi-agent architecture, the many components needed to support the multimodal system, e.g., speech recognition, gesture recognition, natural language processing, multimodal integration, may be written in different programming languages, on different machines, and with different operating systems. Agent communication languages are being developed that can handle asynchronous delivery, triggered responses, multi-casting and other concepts from distributed systems.

Using a multi-agent architecture, for example, speech and gestures can arrive in parallel or asynchronously via individual modality agents, with the results recognized and passed to a *facilitator*. These results, typically an N-best list of conjectured lexical items and related time-stamp information are then routed to appropriate agents for further language processing. Next, sets of meaning fragments arrive at the multimodal integrator which decides whether and how long to wait for recognition results from other modalities, based on the system's temporal thresholds. The meaning fragments are fused into a semantically-and temporally-compatible whole interpretation before passing the results back to the facilitator. At this point, the system's final multimodal interpretation is confirmed by the interface, delivered as multimedia feedback to the user, and executed by any relevant applications.

Despite the availability of high-accuracy speech recognizers and other mature multimodal technologies such as gaze trackers, touch screens, and gesture trackers, few applications take advantage of these technologies. One reason for this is that the cost of implementing a multimodal interface is prohibitive. The system designer must usually start from scratch, implementing access to external sensors, developing ambiguity resolution algorithms, etc. However, when properly implemented, a large part of the code in a multimodal system can be reused. This aspect has been identified and many multimodal application frameworks have recently appeared such as VTT's *Jaspis* and *Jaspis2* frameworks [68, 108], Rutgers CAIP Center framework [109] and the embassi system [110].

2.8 Standards and Tools

Next tools, standards and recommendations for developing GUIs, spoken dialogue and multimodal interaction systems are briefly outlined.

Graphical User Interfaces

In contrast to web development for which widely used standards exist, e.g., HTML, GUI development is characterized by the lack of a single dominant standard. Instead, a multitude of GUI toolkits, along with their corresponding style guides, exist for different desktop operating

systems, e.g., MacOS, Windows, Linux and various platforms, e.g., mobile or desktop. Nevertheless all these GUI toolkits are very similar in appearance and functionality. This makes the application of common design rules and guidelines easier to follow, in practice, regardless of the toolkit choice. Such guidelines, style guides, e.g., the Apple Human Interface Guidelines for desktop [111] or iPhone [112], standards, e.g., ISO 9241 (Ergonomics of Human System Interaction), and toolkits promote usability principles such as consistency and user satisfaction. However, following these guidelines is not always easy for non-HCI expert developers as reported in [52].

The appearance of cross-platform GUI toolkits and development tools that ease GUI development, e.g. automatic creation of GUI related code, helps developers and designers focus on application functionality and design principles, rather than on low-level details. The diversity of GUI toolkits is not expected to vanish any time soon, especially as new devices and platforms keep emerging. This is especially true in the mobile/embedded space where new devices and interaction paradigms appear, posing new challenges and creating new opportunities for system designers.

Spoken Dialogue Interfaces

The VoiceXML Forum[11] an organization founded by Motorola, IBM, AT&T, and Lucent to promote voice-based development, introduced the *VoiceXML* language based on the legacy of languages already promoted by these four companies. In March 2000, version 1.0 was released and in October 2001, the first working draft of the latest VoiceXML 2.0 was published as a W3C recommendation[12]. The VoiceXML standard has simplified the development of voice-based applications much like HTML did for the development of web-based applications. The main features of VoiceXML are the familiar HTML-like syntax, the logic that an application consists of a series of pages (similar to familiar GUI interface logic) and the ability to provide web content using only voice as an input modality, making web information accessible from fixed or mobile phones.

VoiceXML browsers consist of an interpreter and a set of VoiceXML documents. VoiceXML supports dialogues that include menus and forms, sub-dialogues and embedded grammars. The *voice browser* renders the VoiceXML documents as a sequence of the two-way interaction between the system and the end user. Core VoiceXML interpreter and software components are used for this purpose such as *automatic speech recognition* and *text-to-speech*. Many companies build spoken dialogue development toolkits that include building blocks such as sub-dialogues and grammars. Such toolkits often introduce custom tags of objects in addition

[11] http://www.voicexml.org/
[12] http://www.w3.org/TR/voicexml20/

to the VoiceXML standard ones. Using such complete solutions a system designer can implement and test VoiceXML-based applications and *voice portals*, e.g., the Nuance Voice Platform [13] provides an easy-to-use, complete development environment for voice applications. Other commercial offerings include servers for deploying these applications [113], voice browsers, and VoiceXML editors and grammar development tools. There are also open source VoiceXML tools, such as Carnegie Mellon's OpenVXI interpreter [14].

Multimodal Interaction Standards

The number and diversity of devices that can access the Internet has grown tremendously in the past years. The capabilities and modes of access of these devices varies; consider for example mobile phones, smart phones, personal digital assistants, multimedia players, kiosks, automotive interfaces. The W3C *Device Independence Working Group* main focus is on standards that make the characteristics of the device available to the network and, most importantly, on standards that assist authors in creating sites and applications that can be supported on multiple devices. The group coordinates its work with the *Web Accessibility Initiative*[15] and *MultiModal Interaction Working Group*[16] activities as discussed next.

The main goal of the *Multimodal Interaction Activity* is to extend the Web user interface to multiple modes of interaction (aural, visual and tactile), offering users the means to provide input using their voice or their hands via a key pad, keyboard, mouse, or stylus. For output, users will be able to listen to spoken prompts and audio, and to view information on graphical displays. By allowing multiple modes of interaction on a variety of devices the activity aims for *accessibility to all*. The Working Group was launched in 2002 following a joint workshop between the W3C and the WAP Forum with contributions from SALT[17] (*Speech Application Language Tags*) and XHTML+Voice[18] (X+V). Major contributions of this activity include: the *Multimodal Interaction Use Cases*, the *Multimodal Interaction Use Requirements* and the *W3C Multimodal Interaction Framework* [99]. Work has also been done on: (i) dynamic adaptation to device configurations, user preferences and environmental conditions (*System and Environment Framework*) [114], (ii) integration of composite multimodal input and modality component interfaces such as interfaces for ink and keystrokes, and (iii) context sensitive binding of gestures to semantics (note that speech and DTMF modalities are developed by the *Voice Browser Working Group*[19]).

[13] http://www.nuance.com/voiceplatform/
[14] http://www.speech.cs.cmu.edu/openvxi/
[15] http://www.w3.org/WAI/
[16] http://www.w3.org/2006/12/mmi-charter.html
[17] http://www.saltforum.org/
[18] http://www.voicexml.org/specs/multimodal/x+v/12/
[19] http://www.w3.org/voice/

The group's work has also stimulated the creation of mark-up languages such as EMMA, and InkML. The *Extensible MultiModal Annotation Markup Language* (EMMA) [115], is a markup language intended to represent semantic interpretations of user input (speech, keystrokes, pen input etc.) together with annotations such as confidence scores, timestamps, input medium. The interpretation of the user's input is expected to be generated by signal interpretation processes, such as speech and ink recognition, semantic interpreters, and other types of processors. InkML [116], defines an XML data exchange format for ink entered with an electronic pen or stylus as part of a multimodal system, which will enable the capture and server-side processing of handwriting, gestures, drawings and other specific notations.

Other related efforts for multimodal interaction standardization are the SALT and XHTML + Voice efforts. SALT, is a lightweight set of extensions to existing markup languages, allowing developers to embed speech enhancements in existing HTML, XHTML and XML pages. *XHTML+Voice*, by IBM, Motorola and Opera Software, is another effort exploiting the combined use of XHTML and parts of VoiceXML through *XML events* to support for visual and speech interaction.

2.9 Summary

This chapter presented some of the fundamental concepts behind interface design with a focus on multimodal interfaces. The introduction to HCI focused on the definition and principles of usability, namely learnability, flexibility and robustness. The MVC (model-view-controller) paradigm that serves today as the basis for the architectural design of many unimodal and multimodal systems was introduced.

Next some of the input and output modalities that are involved in modern interface design, namely GUI, speech and gestures were presented. Much of the review focused on speech interfaces, both because of the idiosyncratic nature of the speech modality and the breadth of technologies involved in speech recognition and synthesis. Then the discussion turned to the interesting problem of how to combine different modalities to build multimodal interfaces. The review focused on the problems of multimodal fusion and multimedia fission, as well as the potential rewards and pitfalls of multimodal interface design. The main advantages of multimodality are increased interface robustness and usability, especially in adverse conditions.

Adaptation has been used for a large variety of tasks and applications, often successfully, improving the interaction efficiency and the user experience. However, despite the promise that adaptive interfaces hold, designing interfaces that are adaptive and also appear consistent to the user is a challenging task. In addition, adaptive interfaces are complex and the consequences of adaptivity on the user experience is sometimes unpredictable. As a result, system designers often opt for adaptable interfaces, or limit the functionality of the adaptive algorithms.

As mobile devices are becoming increasingly ubiquitous, mobile interface design is emerging as an important research area of human-computer interaction. Designing and implementing interfaces on mobile devices, such as PDAs and mobile phones, is a challenging task because the designer has to operate under various constraints and most importantly the reduced display size and the limited interaction methods. These limiting factors affect interface design and multimodal interfaces have been proposed as a solution to the design of more robust and efficient mobile interfaces.

Most multimodal systems are very complex in terms of architecture and software design and they usually exploit the MVC paradigm and separate the model from the control logic and the interface specification. The differences between GUI and multimodal architectures are examined, and some typical architectures employed in multimodal systems are reviewed. Finally tools, standards and recommendations for developing GUIs, spoken dialogue and multimodal interaction systems are presented.

These are exciting times for the design of innovative interfaces. The explosion of multimedia content available online, improved device capabilities, novel multimedia signal processing algorithms, new interaction modalities and interaction paradigms have created possibilities that we are only now beginning to understand.

Chapter 3

Multimodal Platform and Interaction Design

The main aim of this chapter is to showcase how to design information-filling multimodal systems combining speech and GUI (e.g. pen or touch) input. From the interaction design standpoint, the main focus is on identifying and exploiting the synergies between the modalities and on the investigation of a variety of multimodal interaction modes. The system architecture of the system which allows both unimodal and multimodal interaction and can be used across different platforms such as PDAs and mobiles is examined here and in more detail in Appendix A. A video demonstration of the multimodal systems designed is available online[1].

3.1 System Overview

The system is built using the Bell Labs Communicator dialogue platform described in [5]. The system architecture diagram shown in Fig. 3.1 shows the Communicator SDS (Spoken Dialogue System) augmented to support GUI in addition to the speech modality. To achieve this the system is also able of handling, parsing and interpreting GUI input (e.g. mouse input for the desktop case) and also produce GUI output. This system is then later augmented in order to support multimodal interaction instead of just unimodal interaction as described in Section 3.4 by incorporating a multimodal controller module shown in Fig. 3.5. The system is designed with the main aim of clearly separating the interface from the task (that is the application).

The main application used with the system is a travel reservation form-filling application which allows for flight, hotel and car reservation. In this chapter the main focus is on the form-filling part (result presentation and navigation part of the application is discussed in [5]) of the application. Fig. 3.2 shows a tree structure (part of) representing the application (prototype

[1] `http://www.youtube.com/user/holystone74`

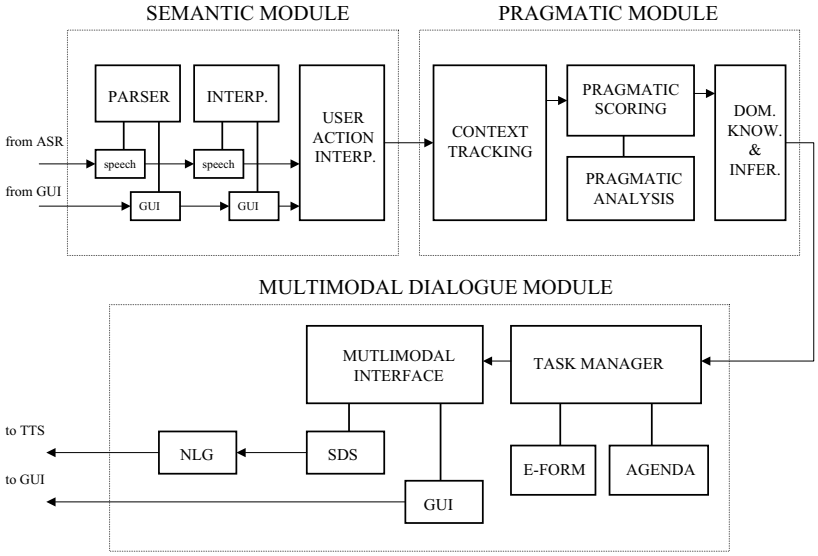


Figure 3.1: Bell Labs Communicator architecture (from [5])

.

tree) and the data collected from user input over multiple dialogue turns (note the flight leg hierarchy and some attribute value pairs, e.g., *departure date*, *June 1* - refer also to Table 3.2 for a list of the attributes used). Switching application, e.g. a movie searching application, is achieved by using just a different prototype tree; Recall that this separation of task and interface is also the main design power of the MVC pattern.

Overall, the system designed, supports five different interaction modes; two unimodal ones, namely, "GUI-Only" (GO) and "Speech-Only" (SO)[2], and three multimodal ones combining the speech and GUI modalities, namely, "Click-to-Talk" (CT), "Open-Mike" (OM) and "Modality-Selection" (MS). In addition, a sixth interaction mode with unimodal speech input and GUI and speech output labeled "Open-Mike Speech-Input" (OMSI) was implemented. In Table 3.1 a summary of the systems described is shown in terms of input and output modalities supported. Note that the three multimodal modes support all available input and output modalities.

The system has been designed to be fully portable across a variety of computing platforms (desktop, PDA and mobile) with minor differences in the GUI design stemming from user interface considerations (thus three different GUI view implementations are provided). The user can communicate with the system using pen and/or speech on the PDA, using keyboard/mouse and/or speech on the desktop and speech and/or touch in the iPhone mobile

[2]which already existed in the original system

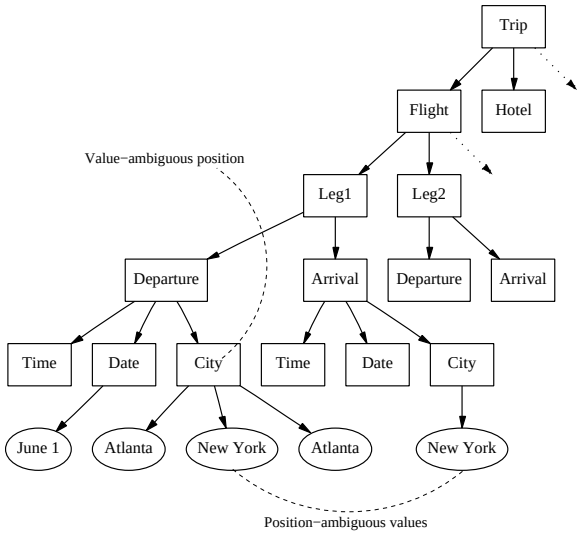


Figure 3.2: Travel reservation application tree (part of) depicting the flight leg hierarchy and the attribute value pairs (from [5]).

device used. Thus in regard to the MVC model, all systems described share the exact same model (the prototype tree) but use different view realizations (speech and/or the three GUI implementations). The controller used exposes the exact same functionality (and behavior) to the various views. Regarding the controller, the only distinction that can be thought of, is between the unimodal and multimodal controller, the later being a superset of the first since it has to coordinate more channels (speech and GUI).

The various systems and interaction modes are described in detail next. First the (original) "Speech-Only" system is described in section 3.2. In section 3.3 the process of designing the "GUI-Only" system in order to support the exact same functionality with the "Speech-Only" is presented. In addition the three different implemented GUI versions for the desktop, PDA and mobile platforms respectively, are also described. The design of multimodal interaction modes that combine the GUI and speech modalities is presented in section 3.4. Two important design issues that are addressed is the exploitation of the combined modalities synergies and the selection of the default input modality in the three multimodal interaction modes designed. The three multimodal interaction modes "Click-to-Talk", "Open-Mike" and "Modality-Selection" are then described in detail. Finally in section 3.5 the "Open-Mike Speech-Input" interaction mode is described. For interesting design and implementation details of the system, please refer to Appendix A.

Table 3.1: Supported input and output modalities in the implemented systems.

	input modalities		**output modalities**	
system	**GUI**	**speech**	**GUI**	**speech**
GO	√	x	√	x
SO	x	√	x	√
OMSI	x	√	√	√
CT/OM/MS	√	√	√	√

3.2 Unimodal speech interaction

The original Communicator uses the BLSTIP [117] telephony platform. This was substituted by a highly flexible audio platform that was designed and implemented (see Appendix Section A.2) in order to be used on both desktop computers and mobile devices (for various operating systems). It is high performance and supports advanced features such as *Voice Activity Detection* (VAD) and *barge-in*, i.e., users speaking over system prompts. The audio platform interfaces with Bell Labs speech recognizer [117] and the FreeTTS [118] speech synthesizer through network sockets.

The "Speech-Only" interface is identical to the one described in [5, 119, 120]. In brief, the spoken dialogue manager promotes mixed-initiative system-user interaction. All types of user requests and user input are allowed at any point in the dialogue, i.e., the full application grammar is active throughout the interaction. The system prompts are focused and try to elicit specific information from the user, e.g., the value of an attribute. Explicit confirmation is used only to confirm the values of the attribute at the form level, e.g., for all flight leg user supplied information. Implicit confirmation is used in all other cases throughout the interaction.

Following is an example interaction scenario:

```
>> Welcome to the Bell Labs travel reservation system.
>> How can I help you?
<< I want to fly from Athens to London

>> I've got you leaving Athens and arriving in London England.
>> Leaving Athens on what date?
<< Tomorrow morning

>> I understand you're leaving on Monday, November ninth in the morning,
>> What is your preferred airline?
<< Olympic
```

```
>> You fly with Olympic airways
>> Is this information correct? [explicit confirmation at the form level]
<< Yes

>> One way trip?
<< No
...
```

Table 3.2: Attribute size (sorted by size) for the travel reservation application. The table is separated in two parts depending on attribute size; we refer to the attributes in the upper part as "long" attributes and the rest as "short" attributes.

attribute name	**attribute size**
hotelname	250
city	135
airline	93
date	22
car type	15
car company	10
time	9

3.3 Unimodal GUI interaction

The application GUI (see Fig. 3.3 and Fig. 3.4) is generated *automatically* from the application tree and the interface specification. It depicts the application tree and state, using a series of forms. Each form contains a list of attribute-value pairs, with each pair employing label and text-field/combo-box/table-view components respectively, depending on GUI view implementation. Three versions of the GUI are implemented[3]: a desktop version which allows for keyboard and mouse input (GUI uses text-field or combo-box components depending on attribute size), a PDA version which only allows for pen input (GUI uses only combo-box components) and an iPhone version which allows for touch input (GUI uses table-views). Flight reservation, hotel reservation and car rental forms are accessible as separate tab panes/tab-bar items in the case of desktop/iPhone and via buttons in the bottom of the form in the PDA GUI[4]. Also note that the "Speech Input" button (used for multimodal interaction) contained in all GUI versions is disabled. Next, the differences between the three GUI designs are discussed.

[3]The desktop, PDA and mobile views use Java Swing, Java AWT and iOS user interface libraries respectively.

[4]Note that the task manager automatically decides when a form is filled and automatically prompts the user to move on to the next form. Thus the form navigation buttons/panes are not used much by the user in our system.

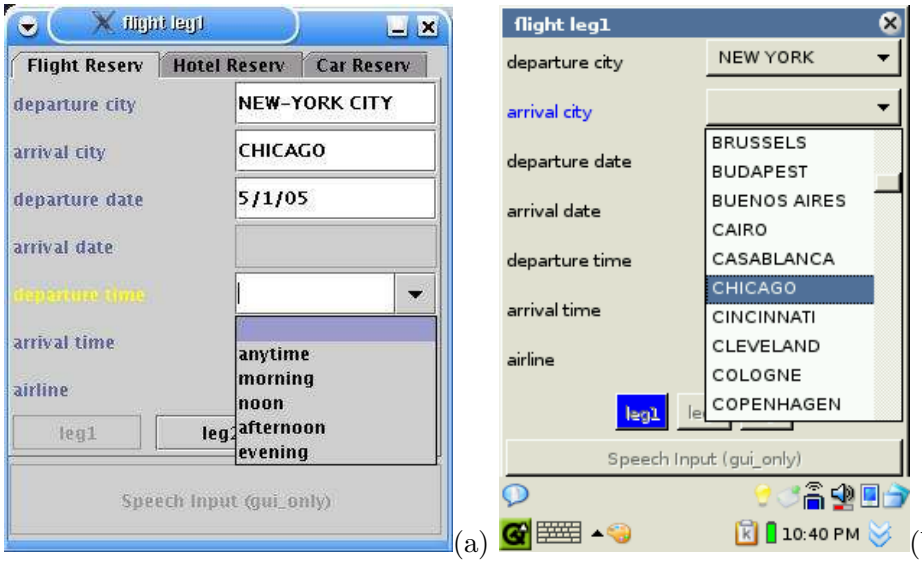


Figure 3.3: GUI-Only interaction examples (a) desktop view (b) PDA view.

Differences between the three GUI designs

Since desktop GUI systems allow for both mouse and keyboard (text) input, the desktop GUI implementation (see Fig. 3.3(a)) exploits both input types to allow for fast GUI interaction. The choice of using text field or combo-box for a certain attribute field, is based on efficiency considerations; that is the number of values of that attribute (attribute size). For small attribute sizes i.e., less than 25 values, a combo-box is used, otherwise a text-field (see Table 3.2). This combination has been found to be the most efficient for our application. For the PDA GUI on the other hand (see Fig. 3.3(b)), all data entry fields are implemented as combo-box components due to the slow text input methods available on such devices. The number of options available to the user in some of these combo-box components is quite large, e.g., 250 choices for the "hotelname" attribute. Note that attribute values in any combo-box appear sorted alphabetically (with the exception of time which is chronologically sorted).

In contrast with the Zaurus PDA which follows a desktop-like GUI interface and is controlled via a stylus, the iPhone uses a touch interface optimized for simple finger gestures operations on the screen (refer to section 2.6.2). Thus instead of the precise pointing of the stylus on PDAs, the larger less precise footprint of finger on the screen has certain implications in the design of the screen components. For example, in contrast with the traditional form views in desktop-like GUIs for which both the field labels and components that contain the fields values (e.g. combo-box) can been fit in a single view, the corresponding form in the iPhone requires a two level view hierarchy.

The main (top) view (a table-view according to iPhone terminology) holds just the field

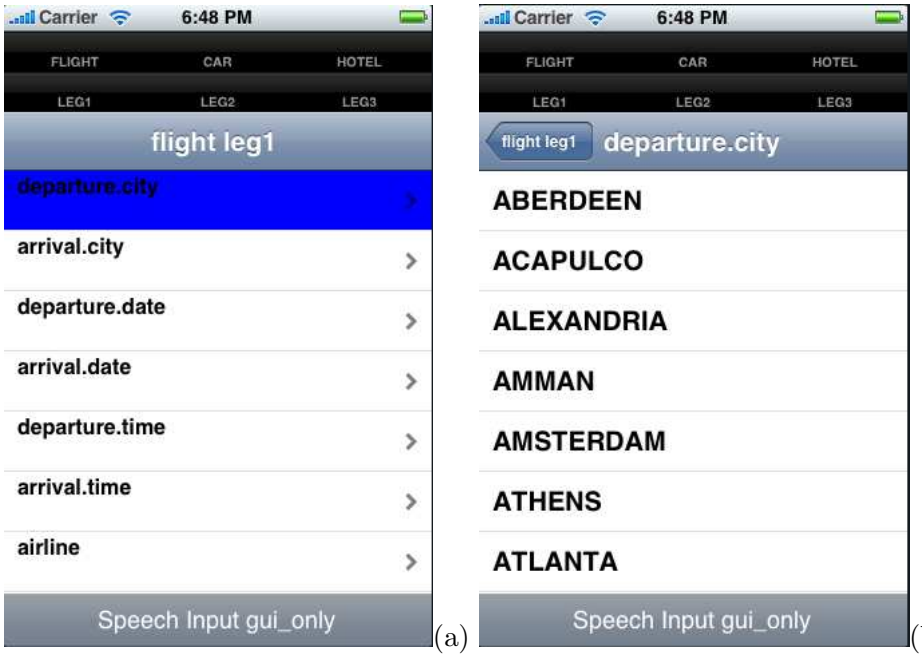


Figure 3.4: GUI-Only interaction in the iPhone device. In contrast with desktop-like interfaces, a form view is represented with a 2-level hierarchy of views (a) top-level view (b) detailed view for deaprture city attribute.

(attribute) labels and the corresponding selected value in each table row (see Fig. 3.4(a)). By touching each row, a new detailed (two-level) view containing all the possible values the user can select from, is shown (Fig. 3.4(b)). A navigation bar indicates the depth level in the hierarchy; after the user scrolls and selects the desired value the detailed view disappears and the main view is shown again with the selected value shown next to attribute label.

Common features

To ensure the exact same functionality and interaction experience with the "Speech-Only" system (albeit with a different representation), the following features are common for all three versions of the GUI : (1) the current context (or focus) of the interaction is highlighted in each turn (2) GUI components that become inaccessible in the course of the interaction are "grayed out" (3) information and error messages are represented in the GUI as pop-up dialogues, and (4) ambiguity is shown as a pull-down box with a list of choices and highlighted in red.

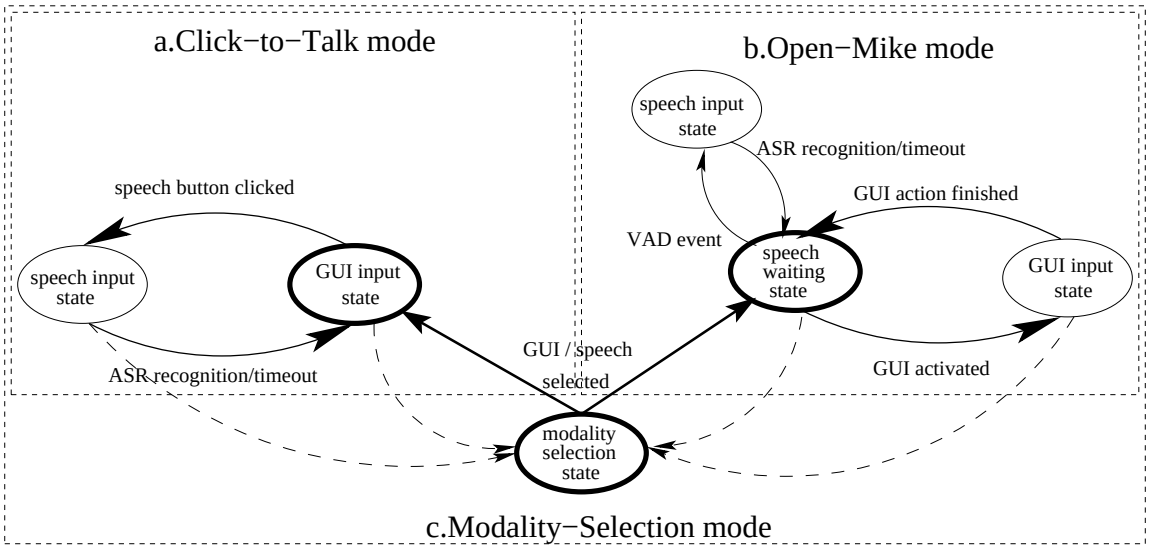


Figure 3.5: State diagrams of the three multimodal interaction modes: (a)"Click-to-Talk", (b)"Open-Mike" and (c)"Modality-Selection".

3.4 Multimodal interaction

In this section the design of multimodal interaction is described. The two important issues that have driven the design of the systems described next is the exploitation of the synergies and the selection of the default input modality; these issues are addressed in sections 3.4.3 and 3.4.4 respectively.

3.4.1 Design issue I: exploitation of modality synergies

It is widely supported that voice user interfaces (VUI) and graphical user interfaces (GUI) when combined to create a multimodal system offer high complementarity for most applications [9, 24, 25, 26]. As far as input is concerned, GUI interfaces have low error rates and offer easy error correction. Although speech is not error-free, it may be more efficient for relatively high speech recognition accuracy and high verbosity (number of tokens communicated). It is also considered the most natural type of input compared to other modalities. As far as output is concerned, GUI output is fast (parallel) compared to much slower (sequential) speech output. Thus, multimodal systems that combine GUI and speech interfaces can potentially become more efficient in terms of time to complete a task by taking advantage of: (i) **"input modality choice"** synergy, i.e., the user (or system in an adaptive user interface) chooses the most appropriate input modality for each turn (ii) **"visual-feedback"**, i.e., the more efficient cognitive processing of visual compared to auditory information, (iii) **"error-correction"** synergy, i.e., correcting errors of the VUI via the GUI [27].

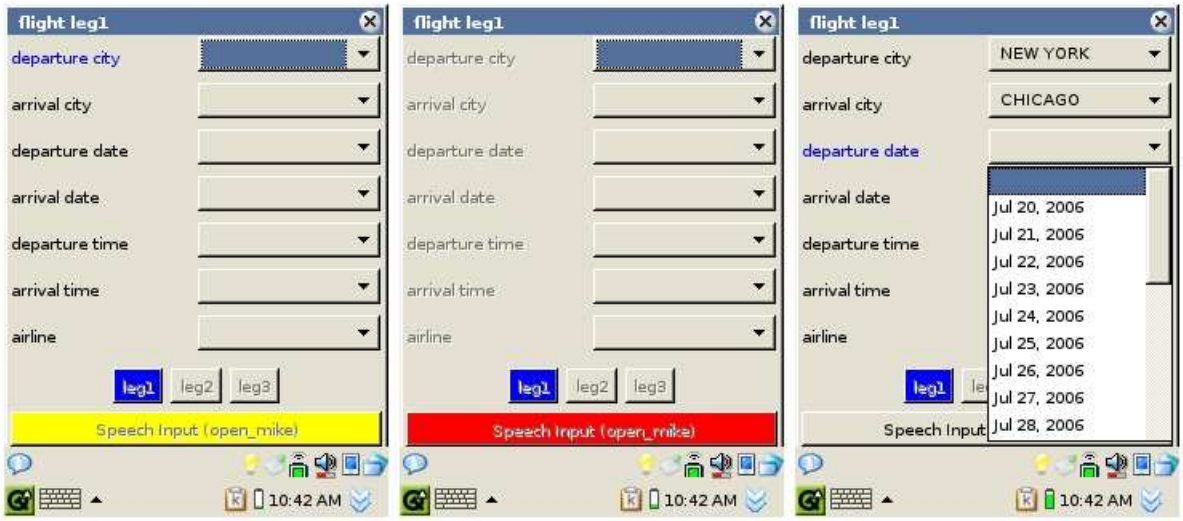


Figure 3.6: "Modality-Selection" interaction mode example on the PDA platform. System is in "Open-Mike" mode in the first frame (speech button is yellow indicating waiting for input), receives user input "From New York to Chicago" during the second frame (speech button is red showing a VAD has taken place) and switches to "Click-To-Talk" mode in the third frame. The speech/pen input default modality is selected by the system in the first/third frame, respectively, due to the large/small number of options in the combo-box.

3.4.2 Design issue II: selection of input interaction modality

A fundamental issue one has to consider when building multimodal interfaces is the suitability of various input interaction methods for different tasks and subtasks [23]. Combining multiple modalities efficiently is a complex task and requires both good interface design and experimentation to determine the appropriate modality mix. Few guidelines exist for selecting the appropriate mix of modalities [22, 29, 30]. It is often the case when designing multimodal user interfaces that the developer is biased either toward the speech or the GUI modality. This is especially true, if the developer is speech-enabling an existing graphical user interface (GUI)-based application or building a GUI for an existing speech-only service.

Another issue that is not thoroughly researched is the design of multimodal turn-taking and the selection of the most appropriate interaction modality in each turn. Should users be allowed to interact as in traditional spoken dialogue systems (SDS) where a voice-activity detector allows the user to barge-in and speak at any moment (commonly referred as an "Open-Mike" interaction mode), should the user be constrained as in the GUI paradigm to press a button to activate the speech recognizer ("Click-to-Talk"), or should either interaction modes be used were appropriate.

3.4.3 Common design of the multimodal interaction modes

The output interface is common for all multimodal interaction modes to allow us to better investigate the effectiveness of the input modality mix. The GUI output is identical to the corresponding "GUI-Only" mode. Audio output prompts were significantly shortened compared with the unimodal "Speech-Only" case. Specifically, implicit confirmation prompts were not used in the multimodal case because confirmation was efficiently done via the GUI modality (mouse/pen/touch). In addition, form creation prompts and explicit confirmation prompts were significantly shortened or not used at all, depending on the interaction context. Finally, information request prompts were shortened down to the name of the attribute requested, e.g., "Arrival city?" (compare this to the longer prompt "I have got you leaving Chicago, where are you flying to?" of the "Speech-Only" mode). In general, speech output was mainly used as a way to grab the attention of the user, emphasizing information already appearing on the screen. The speech interface was identical for all multimodal modes.

Note that in all three multimodal modes only one modality is active at a time, i.e., the system does not allow for concurrent multimodal input[5]. GUI input is not allowed (GUI is "grayed-out") while speech input is active. Also, for all multimodal modes, users are free to override the system's proposed input modality, that is, use a modality other than system's default, e.g., GUI input for "Open-Mike" mode for which speech is the default input modality. The functionality of each multimodal mode is discussed in detail next.

3.4.4 Differences between the multimodal interaction modes

The main difference between the three multimodal interaction modes is the default input modality used at each turn. For "Click-to-Talk" interaction, GUI is the default input modality; the user needs to click the "Speech Input" button to override the default input modality and use speech as an input instead. The "Speech Input" button turns then red to highlight that audio capture and speech recognition are active and the whole GUI view becomes disabled (GUI input not allowed while speech input in progress). Once recognition finishes, the recognized results update the GUI view and the focus advances to the next expected attribute where "Speech Input" button turns gray and becomes clickable again.

For "Open-Mike" interaction, speech is the default input modality; the system is always listening and a VAD event activates the speech recognizer. The "Speech Input" button is not clickable in this mode and has yellow color to indicate that audio recording is active at the beginning of each turn; once a VAD event happens it turns red to indicate that speech recognition is active. Again the user can override the default input modality by using GUI

[5]For information-seeking/form-filling multimodal applications this is not a major limitation.

input (e.g. selecting a combo-box with a pen on the PDA device). Note however, that this can not happen if a VAD event has already taken place (only one modality active at each turn).

"Modality-Selection" is a mix of the "Click-to-Talk" and "Open-Mike" interaction; the system switches between the two multimodal modes depending on efficiency considerations (the size of the attribute that is in focus in the current turn). For short attributes (GUI input faster than speech), the system goes into "Click-to-Talk" mode and GUI input becomes the default input modality, otherwise the system goes into "Open-Mike" mode where speech becomes the default input modality. Thus "Modality-Selection" selects the input modality in a static way (current attribute size); It is a simple version of the adaptive modality tracking algorithm proposed in [120].

The state diagrams of the three multimodal interaction modes are shown in Fig. 3.5. For "Click-to-Talk" (Fig. 3.5(a)) the default system state is the GUI input state (default state shown in bold); the user can transition to the speech input state by pressing on the GUI the "Speech Input" button. Upon being pressed, "Speech Input" button turns red (indicating that the speech recognizer is active), the speech prompt is stopped (barge-in event) and the GUI is disabled ("grayed-out") for the duration of the speech recognition event. At the end of the speech input turn (speech recognition completed or a time-out happened) the system returns to the GUI input state: the GUI is enabled and so is the "Speech Input" button.

For "Open-Mike" (Fig. 3.5(b)) the default system state is the "speech waiting " state. In this state, the color of the "Speech Input" button is yellow to indicate to the user that he/she can speak at anytime. When voice activity is detected, the system goes to the "speech input" state; the "Speech Input" button turns red, GUI is disabled and the audio prompt is stopped. Upon completion of the speech recognition event the system returns to the "speech waiting" state. The system goes to "GUI input" state if the user starts interacting with the GUI input modality; once finished the system returns to the default "speech waiting" state.

For "Modality-Selection" (see Fig. 3.5(c)), the default system state is the "Modality-Selection" state, where the system determines (at the beginning of each interaction turn) the preferred input modality: GUI or speech. Based on the modality selected, the system transitions to the default state of the "Click-to-Talk"/"Open-Mike". Once a user input turn is complete, the system transitions back to the "Modality-Selection" state (following the dotted lines in Fig. 3.5(c) rather than the solid lines and selects the modality for the next turn.

In Fig. 3.6, examples from the "Modality-Selection" mode running on the PDA, are shown. Initially the interaction focus is on "departure city", the speech modality is selected (over 25 options available) and the system goes to "speech waiting" state. User input "from New York to Chicago" activates the speech recognizer (VAD event) and the GUI becomes disabled ("speech input" state). Once recognition of the spoken utterance finishes, the GUI is updated and the modality is selected for the next turn ("modality selection" state). For the next turn,

GUI input is selected (focus is on "departure date" for which a combo-box with less than 25 choices is available) and the system goes to the "GUI input" state.

3.5 Other interaction modes

To better investigate the effect of "visual feedback" synergy in spoken dialogue interaction, a system with limited multimodal capabilities was also implemented, namely "Open-Mike Speech-Input" (OMSI). The user is allowed only speech input while the system output includes both speech and visual feedback. OMSI interaction is thus equivalent to "Open-Mike" interaction with GUI input disabled. Alternatively OMSI can be seen as a "Speech-Only" system with visual feedback and shortened prompts. Note that the OMSI prompts are identical to the rest multimodal systems prompts.

Chapter 4

Evaluation Methodology

4.1 Introduction

In the previous chapter the design of a dialogue system supporting two unimodal and three different multimodal interaction modes was described. In this chapter the methodology used for evaluating the system is presented with a focus on the evaluation metrics used. Some of these metrics are standard objective metrics used in dialogue systems while the rest were devised specifically for the investigation of two important research questions, namely the relation of input modality choice to unimodal efficiency and the measurement of the synergies in multimodal interaction modes. Overall the metrics used aim at: (i) comparing in terms of performance and user satisfaction all the interaction modes (unimodal and multimodal). (ii) identifying input modality selection patterns in the multimodal interaction modes and their relation to unimodal efficiency, e.g. is modality selection proportional to the ratio of unimodal efficiency ratio? (iii) measuring the synergies of the multimodal interfaces.

The objective metrics used are described in section 4.2. Since the system evaluated is a dialogue based system, metrics for the evaluation of SDSs can be applied such as task completion ratio, number of turns and turn duration times. These metrics can be additionally measured per user, task or turn. One important improvement is the break down of turn duration times into inactivity and interaction times which allows to separate system output processing (by user) from user input, in order to better study differences between the various interaction modes. Another metric related to multimodal interaction is the number of overrides which describes how well a multimodal system matches users' modality preferences. The two metrics devised related to items (ii) and (iii) in the previous paragraph, namely relative modality efficiency and multimodal synergy are defined in section 4.3 and their computation is described in section 4.3.3.

4.2 Objective evaluation metrics

Interface evaluation of multimodal dialogue systems is a fairly complex task and different metrics may be used to evaluate various aspects of such systems [121, 34]. Since one of the main interests is in computing the relationship between modality usage and relative efficiency of input modalities, two depended variables become of high importance: modality selection (GUI or speech) and user turn duration (that is the time spent in each turn, for user input to the system using either modality[1]).

In this work, the focus is in the form filling part of the interaction and most specifically on how the user provides attribute-value pairs to the system[2]. Other parts of the interaction such as confirmation questions, verification requests, and navigation among forms were not included in the analysis. The main reason for this is that for the vast majority of these actions, users used GUI input, as it was clearly the faster and easier way to respond, e.g., click "Yes" on a dialog window, containing the question "Is this correct"?. By excluding the navigation, confirmation and verification actions the biasing of the evaluation results is avoided.

Dialogue based form filling systems are turn based. Turn duration (refer to Fig. 4.1) is the sum of user and system processing/response duration. Interaction efficiency focuses on the first component, which in turn consists of user inactivity and interaction times as defined in section 4.2.2.

Based on user turn times, statistics like average turn duration (mean of turn time), overall user times (sum of turn time) and number of turns can then be computed for a certain factor (independent variable) of interest, such as the interaction mode, the user and the attribute (context). The rest of the section discusses the projection of evaluation data to various factors in order to compute statistics for the two depended variables of interest, namely user turn duration and modality selection.

Next a short summary of objective metrics used in this study along with their intended use is outlined:

- Input modality (GUI/speech) usage: Can be computed per user/system/attribute to reveal relation to unimodal efficiency.
- Input modality overrides: How well does the multimodal interaction mode matches user input modality preferences.

[1] This time also includes an overhead in the case of speech input (ASR overhead time), which has been found to be relatively small and is thus neglected by the analysis.

[2] Note that error correction turns are included. Excluded from the analysis are only turns that are responses to YES-NO questions such as "Is this a one way trip?" or "Is this correct?" (that occurs after filling out each form).

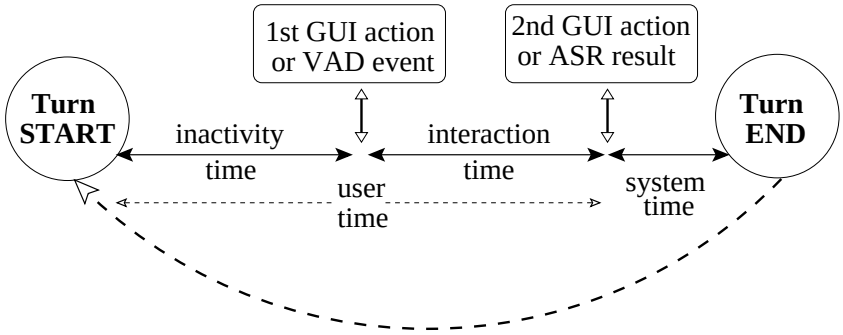


Figure 4.1: Turn time decomposition to user and system time. Note that user time can be further broken down to inactivity and interaction time.

- Inactivity/interaction times: Separate input efficiency (related to interaction times) from output efficiency and modality selection overhead (related to inactivity times).
- Context(attribute) statistics: Relate input modality efficiency for each attribute with modality usage.
- User statistics: Identify individual user patterns.

4.2.1 Modality selection and input modality overrides

The issue of modality usage is a focal point of our research. To answer this question a projection of data on the modality factor is done. Specifically, the usage of each input modality as a function of number of turns is measured and also the duration of turns attributed to each modality. Modality usage is also measured as a function of context, i.e., attribute for which input is expected, as discussed in Section 4.2.3.

Recall that in all three multimodal interaction modes, users have the choice to select among GUI or speech input at each turn, regardless of the default input modality proposed to the user. Thus another related measure is the number of input modality overrides, i.e., the number of turns where users preferred to use a modality other than the one proposed by the multimodal interaction mode. Low number of overrides reveals that the multimodal mode matches user's modality preferences and/or that the modality selection process is system-initiated for this user. A high number of overrides reveals a mismatch to user's modality preferences and/or a power-user that takes the modality selection initiative. The number of overrides is defined as the number of speech input turns for "Click-to-Talk" mode and as the number of GUI input turns for "Open-Mike" mode. For "Modality-Selection", the number of overrides is defined as the number of speech input turns for "short" attributes (where the system selects GUI input

as default) plus the number of GUI input turns for the "long" attributes (where the system selects speech input as default).

4.2.2 Turn duration, inactivity and interaction time

Duration statistics at the turn and task level are important factors, since in this work, efficiency is defined as being inversely proportional to task duration. In addition to measuring turn and dialogue duration in total and for each input modality, turn duration is further decomposed into interaction and inactivity times (refer to Fig. 4.1). Inactivity time[3], refers to the idle time interval starting at the beginning of each turn, until the moment the user actually interacts with the system using GUI or speech input. During this interval, the user has to comprehend the system's response and state and then plan his own response according to the scenario information. The response typically includes entering the system's requested information, using his preferred modality for that turn. Let us refer to this time as interaction time. By breaking the turn duration into interaction and inactivity time it becomes easier to focus better on user input and system output processing by user and investigate them separately.

Inactivity time

For GUI input, the inactivity time is defined as the time interval between the turn start time and the moment the user starts interacting with the GUI. This GUI action event may be the click on the combo-box for PDA case when user starts writing in a text-field for the desktop case or when user touches to select an attribute for the iPhone case. For the case of speech input, inactivity time is defined as the time interval between the turn start time and the moment of a VAD event, that is the moment the audio subsystem has detected speech activity and starts sending speech samples to ASR. Note, that in the case of "Click-to-Talk" mode, one would expect this time to be higher compared, e.g., to "Open-Mike", since the user has to also click the "Speech Input" button to start the audio recording first[4].

Interaction time

For GUI input, interaction time is defined as the time interval between the moment of the first GUI event and the moment the user selects the desired value. For speech input, interaction time is defined as the time interval between the moment of the VAD event and the moment ASR result becomes available (this also includes an ASR overhead time as discussed earlier).

[3]The term "inactivity" refers to the fact that the user *appears* inactive to the system.
[4]"Click-to-Talk" has voice activity detection enabled in this evaluation.

4.2.3 Context statistics

Context statistics refer to the objective metrics regarding the attributes shown in Table 3.2, also referred to as contexts during the course of interaction. Given that the default modality in the "Modality-Selection" mode is chosen based on attribute size, modality usage and duration statistics as a function of context will help us better understand the relation between efficiency and modality choice.

In addition to the traditional computation of the mean (and variance) of turn duration as a function of context, the empirical probability density functions (PDFs) of turn duration for each context is also computed. The empirical distributions are computed as a function of context and modality, for the interaction and inactivity time of each turn.

4.2.4 User statistics

User variability is another important issue that is investigated in this work. The efficiency of each modality is different for each user due to different GUI performance, speech recognition accuracy and prior experience with speech interfaces. Also users may have different modality preferences (bias towards a certain modality) that largely affect modality selection and overall performance. Individual user statistics also give us an idea of the degree of variability that users exhibit in making modality selection decisions and can help us to better understand the generality of the drawn conclusions on the relation between efficiency and modality usage.

4.3 Synergy and Relative Modality Efficiency metrics

Objective metrics are extensively used in HCI in order to evaluate the usability of a system. Common metrics used for the evaluation of both spoken dialogue and multimodal dialogue systems include task completion, time to task completion, number of turns, word and concept error rates. Such metrics can be computed per user, task or subtask. In addition, for multimodal systems, objective metrics such as the usage of each modality (both in number turns and total duration) are computed which can reveal usage patterns and modality efficiency. Although these metrics are very useful for direct comparison between competing interface implementations and systems, the metrics themselves may not provide enough insight from a usability standpoint.

Next two new metrics that can help the system designer (in conjunction with the aforementioned metrics) gain a deeper insight into usability aspects of multimodal interface design are defined. The first metric, relative modality efficiency, computes the amount of information communicated in unit time for each modality, i.e., the information bandwidth. Relative modality efficiency should correlate well with relative modality usage unless there is modality

overuse (bias towards a certain modality). The second metric, multimodal synergy, compares the multimodal interfaces with the "sum" of its unimodal parts and measures how "synergistic" the interface design is.

4.3.1 Definition of Relative Modality Efficiency metric

Modality efficiency is defined here to be proportional to the inverse of the time required by that modality to complete a task. Specifically, lets assume that T_s and T_g is the overall time spent using the speech and GUI modality respectively for a form-filling task using multimodal interface. The number of fields (attributes) that are filled correctly using each modality is N_s and N_g respectively[5]. The relative efficiency of the speech modality (compared to the GUI modality) is defined as:

$$E_s = \frac{\frac{N_s}{T_s}}{\frac{N_s}{T_s} + \frac{N_g}{T_g}} = \frac{N_s T_g}{N_s T_g + T_s N_g} \tag{4.1}$$

for a GUI and speech multimodal interface. *Thus efficiency is proportional to the number of tokens (filled fields) communicated correctly in unit time*, or else the *information bandwidth* of each modality.

Relative modality usage is defined here as the percent of time spent using this modality over the total interaction time. For example, for a speech and GUI system, the relative usage of the speech modality is defined as

$$U_s = \frac{T_s}{T_s + T_g}. \tag{4.2}$$

For a user that selects modalities based solely on efficiency consideration the ratio of modality efficiency to modality usage, E_s/U_s should be approximately one. This is equivalent to using each modality in proportion to its information bandwidth, i.e.,

$$\frac{E_s}{U_s} = 1 \;\Rightarrow\; T_s \sim \frac{N_s}{T_s}. \tag{4.3}$$

Ratios $E_s/U_s > 1$ signify underuse of the speech modality while $E_s/U_s < 1$ signify overuse (speech bias).

Alternatively, one can define relative modality usage in terms of the number of turns rather the time spent using each modality. Let us define Q_s and Q_g the number of speech and GUI

[5]Field refers to any attribute defined in the GUI that has a label and gets filled, thus a single field might contain variable numbers of concepts or words, e.g., "date" field. Also note that there are cases where both modalities are used to correctly fill a field, e.g., correction of speech recognition errors via the GUI, slightly biasing our estimator.

turns, respectively. Then, the percent of speech usage is defined as :

$$QU_s = \frac{Q_s}{Q_s + Q_g}. \tag{4.4}$$

4.3.2 Definition of Multimodal Synergy metric

Multimodal synergy is defined as the percent improvement in terms of time-to-completion achieved by our multimodal system compared to a multimodal system that randomly combines the different modalities. For the case of the designed system, where the GUI and speech modalities are combined, time-to-completion for the "random" system is computed as the weighted linear combination of the time-to-completion of the "Speech-Only" and the "GUI-Only" systems, with weights proportional to the usage of each modality in the actual multimodal system. Specifically, lets assume that D_s, D_g and D_m are the time-to-completion of the "Speech-Only", "GUI-Only" and multimodal systems, and U_s and U_g are the relative usage of the speech and GUI modalities in the multimodal system (normalized in [0,1] and summing to 1 as defined in the previous section). Then the time-to-completion of the multimodal system D_r that randomly selects a modality at each turn (respecting the a-priori probability of modality usage) is $D_r = U_s D_s + U_g D_g$. In general, $D_r = \sum_i U_i D_i$, where i sums over all available modalities. Multimodal synergy S_m for a multimodal system m is defined as:

$$S_m = \frac{D_r - D_m}{D_r} = 1 - \frac{D_m}{\sum_i U_i D_i} \tag{4.5}$$

where i sums over all modalities and corresponding unimodal systems.

Note that multimodal synergy expresses the relative improvement in terms of time-to-completion achieved by multimodal interfaces over the sum-of-its unimodal parts, thus the term *synergy*. Also note that synergy may be negative. For example, a multimodal system that combines modalities inefficiently, does not exploit synergies well or is difficult or complex to use (increased cognitive load) may have negative multimodal synergy.

An alternative definition of synergy is to compare the time to completion of the multimodal system D_m with the *average* time to completion of the corresponding unimodal systems $D_r^R = (1/N)\sum_{i=1}^{N} D_i$, i.e., use a "truly" random combination of the unimodal systems. Thus, the random-combination modality synergy S_m^R for a multimodal system m is defined as:

$$S_m^R = \frac{D_r^R - D_m}{D_r^R} = 1 - \frac{N\ D_m}{\sum_{i=1}^{N} D_i} \tag{4.6}$$

where N is the total number of available modalities. One can argue that this definition of synergy fully captures the efficiency gains due to the "input modality choices" of the user.

Indeed in almost all practical situations the random-combination synergy will be greater than the multimodal synergy defined above.

Finally, note that although the discussion here focuses on input modality synergy, the formulas above capture also output or presentation synergies. If one wants to focus solely on input modality synergies, all unimodal systems used to compute D_r or D_r^R should share the same multimedia output interface. For multimodal dialogue systems this means that the unimodal speech input system should allow for graphical output, i.e., "visual feedback". This speech input/multimedia output system is the OMSI system defined in section 3.5[6].

4.3.3 Use of Synergy & Relative Modality Efficiency metrics

Relative Modality Efficiency & Modality Selection

The percent relative efficiency defined in Eq. 4.1 can be computed as a function of the interaction mode, interaction context or user, by adjusting appropriately the time T and number of tokens N in the definition. Modality efficiency results can additionally be computed for overall time, interaction and inactivity time.

Similarly the relative modality usage is computed based on Eq. 4.2 or Eq. 4.4. The two quantities should be plotted against each other to help us understand inefficiencies in the modality usage. By depicting the relative efficiency and modality usage in a 2D-plot for different modes, contexts and users, it is easy to identify when modality usage is not proportional to modality efficiency; this might be due to poor interface design or user bias towards a certain modality.

Computation of Multimodal Synergy

Likewise, synergy can be computed for each interaction context, interaction mode, user or any combination of the above, by using the appropriate time D measurements. In this evaluation, the random combination synergy defined in Eq. 4.6 is used, since it is easier to compute and interpret. Results are derived for inactivity, interaction and overall times. The breakdown into interaction and inactivity time is especially relevant because interaction roughly corresponds to time spent on user input, while inactivity roughly corresponds to time spent on system output and cognitive processing. As a result, *interaction synergy measures input synergies*, and *inactivity synergy measures output plus cognitive load synergies* [7]. The breakdown can help the designer pinpoint usability problems in the interface design.

[6] It is experimentally verified that a significant portion of multimodal synergy is due to "visual feedback".

[7] Cognitive load synergy is probably a misnomer since this quantity is usually negative. This is due to the fact that the inclusion of additional input and output modalities usually increases cognitive load.

Chapter 5

Evaluation Results

In this chapter detailed evaluation results using the metrics described in the previous chapter are provided. First the interaction modes used for the evaluation are listed (see section 5.1); these include unimodal and multimodal systems running on the desktop and PDA environments. The evaluation scenarios, participants and evaluation procedure is also described. Objective evaluation results are presented in section section 5.2; these include context statistics, input modality overrides and distributions of turn duration times broken down into inactivity/interaction times and input modality type. Subjective results are presented in section 5.3 and a discussion of objective and subjective results is provided in section 5.4. Relative modality efficiency and multimodal synergy results are shown in section 5.5 and discussed in section 5.6. Note that in the analysis that follows, the main focus is on the PDA environment; reference to desktop results is done, only when important differences are found.

5.1 Evaluation setting

5.1.1 Apparatus

Evaluation for both desktop and PDA environments includes the "GUI-Only" (GO) and the three multimodal interaction modes, "Click-to-Talk" (CT), "Open-Mike" (OM) and "Modality-Selection" (MS). In addition, two speech-input modes were evaluated, namely "Speech-only" (SO) and "Open-Mike Speech-Input" (OMSI). Thus a total of 10 systems were evaluated. Evaluation took place in an office environment, with all software (spoken dialogue system, speech platform, GUI interface) running on the same host computer for the desktop and speech-only systems. For the PDA system, evaluation took place with all the back-end software (spoken dialogue system, speech platform) running on the same host desktop computer and the front-end (GUI interface) running on a Zaurus Linux PDA device. Note that OMSI evaluation took place in the desktop environment.

Table 5.1: Evaluation scenarios

Scenario ID	**flight**			**hotel**	**car**
	leg1	**leg2**	**leg3**	**reservation**	**rental**
1	√	-	-	-	-
2	√	√	-	-	-
3	√	√	-	√	-
4	√	√	-	-	√
5	√	√	√	-	-

Table 5.2: Attribute size and attribute usage for the five travel reservation scenarios

attribute name	**attribute size**	**scenario usage**					
		1	2	3	4	5	total
hotelname	250	0	0	1	0	0	1
city	135	2	3	3	3	3	14
airline	93	1	1	1	1	1	5
date	22	1	2	2	2	3	10
car type	15	0	0	0	1	0	1
car rental	10	0	0	0	1	0	1
time	9	1	2	2	2	3	10

5.1.2 Evaluation scenarios and participants

All systems were evaluated using five scenarios of varying complexity: one/two/three-legged flight reservations and round trip flights with hotel/car reservation. Table 5.1 summarizes the required forms in each of the five scenarios. All five scenarios used are shown in Appendix B.1. In Table 5.2, the usage of attributes in each scenario as well as cumulative attribute usage across scenarios is shown. For example, in the first scenario (third column in Table 5.2), the user is required to book a one-way morning flight from Las-Vegas to Miami on July 10th with Northwest airlines; thus *city* attribute is used twice, while $date, time$ and *airline* attributes once. In Table 5.2 attributes are ordered by size. Let us refer to the three attributes listed, namely *hotelname*, *city* and *airline*, that have more than 25 possible values as "long" attributes while the rest are referred to as "short". Note that the cumulative attributes usage across all scenarios is about the same for "long" and "short" attributes (20 vs 22).[1]

Eight non-native English-speaking university students evaluated all systems on all five scenarios. All users had prior limited experience by participating in a previous evaluation of an older version of the system.

[1]This means than on average, if modality selection is solely based on efficiency considerations, one expect that usage of speech and GUI input in multimodal modes will be roughly the same.

5.1.3 Evaluation procedure

The evaluation procedure is described next. First, users are given a short introductory document which explains the system functionality with emphasis on the interaction modes to be evaluated. In order to familiarize users with the system before actual evaluation takes place, users are asked to complete a demo scenario using all different systems, for a maximum of 30 minutes. Finally evaluation takes place, by asking users to complete all five scenarios using all ten systems (a total of 50 sessions per user and 40 sessions per interaction mode). Systems are evaluated in random order and logs for each session are saved for later processing by the analysis software written (objective evaluation). Upon completion of all runs, an exit interview is conducted (user feedback and overall subjective evaluation), using a questionnaire to measure the subjective opinion of each user for each system and modality. The subjective evaluation questionnaire used was similar to the one in [5].

5.2 Objective evaluation

5.2.1 System performance comparison

One of the goals of this study is to compare the different unimodal and multimodal interaction modes in terms of efficiency. User time (time to completion) and total number of turns for all ten systems (four for desktop, four for PDA and the two speech input interaction modes) over all users and evaluation scenarios are shown in Fig. 5.1(a) and Fig. 5.1(b) respectively.[2] Overall, SO is the less efficient mode. OMSI (equivalent to SO interaction mode with visual feedback and shortened prompts) is much faster compared to SO mode; in fact, its efficiency is much closer to the efficiency of the multimodal modes rather than the efficiency of the SO mode. For both desktop and PDA environments, OM is the fastest mode closely followed by MS mode and then by the slower GO and CT modes. Note that minor differences exist between desktop and PDA in GO mode efficiency, number of turns and GUI input usage (slightly higher for the PDA case - see Fig. 5.1(b)).

5.2.2 Turn duration, inactivity and interaction times

Fig. 5.2(a) shows average turn durations broken into interaction and inactivity times for all ten systems. ANOVA analysis was conducted for the four PDA and the two speech only systems (desktop systems are also shown in Fig. 5.2(a)). A within subjects ANOVA shows that the effect of system on inactivity ($F_{5,2014} = 83.78$, $p < 0.001$), interaction ($F_{5,2014} =$

[2]Note that these results are normalized for 38 instead of 40 runs per system, to compensate the failure of 2 runs for OMSI and 1 run for "SO" systems (completion rate 95% and 97.5% respectively as shown in Table 5.3). These outliers were not included in the data to avoid biasing the results.

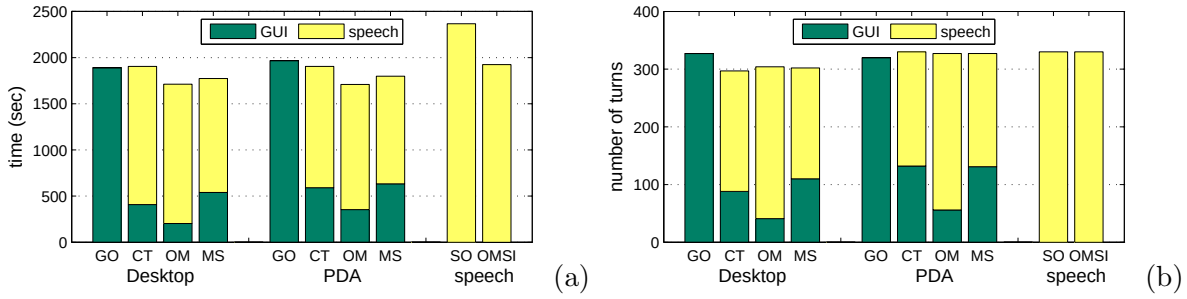


Figure 5.1: Duration and turn cumulative statistics shown for each of the desktop, PDA and speech-only systems summed over all scenarios: (a) total time to completion in seconds, (b) total number of turns. The color-codes for each system bar show the total time and number of turns for GUI and speech input respectively.

33.98, $p < 0.001$) and overall times ($F_{5,2014} = 23.97$, $p < 0.001$) are all highly significant. A post-hoc Tukey HSD test ($p < 0.05$) was performed to find any significant differences. For inactivity times, GO is the faster, followed by OM, then the CT, MS and OMSI systems (whose in-between differences are not statistically significant) and finally, SO which has the highest inactivity times. For interaction times, GO is by far the slower mode; there are no significant differences among the other systems, except for the MS that has the lowest interaction times. Finally, SO has the highest overall times, followed by GO, OMSI and CT, and then the MS and OM systems. Note that the multimodal modes have shorter interaction times compared to GO and shorter inactivity times compared to SO.

Next, refer to Fig. 5.2(b) that shows average turn durations broken into interaction and inactivity times and grouped by input type (GUI/speech) for the PDA system. SO (also shown in Fig. 5.2(b)) and OMSI systems are also included in the ANOVA analysis. A within subjects ANOVA shows that the effect of system/input on inactivity ($F_{8,1990} = 74.25$, $p < 0.001$), interaction ($F_{8,1990} = 32.48$, $p < 0.001$) and overall times ($F_{8,1990} = 23.32$, $p < 0.001$) are all highly significant. A post-hoc Tukey HSD test ($p < 0.05$) was performed to find any significant differences.

For pen (GUI) input (left part of Fig. 5.2(b)), MS inactivity times are higher compared to GO, CT and OM whose in-between differences are not significant. For speech input (right part of Fig. 5.2(b)), SO and CT have the higher inactivity times, followed by MS, then OMSI and finally OM. All differences are significant except for SO and CT. For pen input, all interaction times differ, except for GO and OM (whose estimate is based only on 66 inputs); for speech input however there are no significant differences among the systems.

Furthermore, note the short inactivity times and varying interaction times for GUI input shown at Fig. 5.2(b). Inactivity times are short (compared to speech input inactivity times)

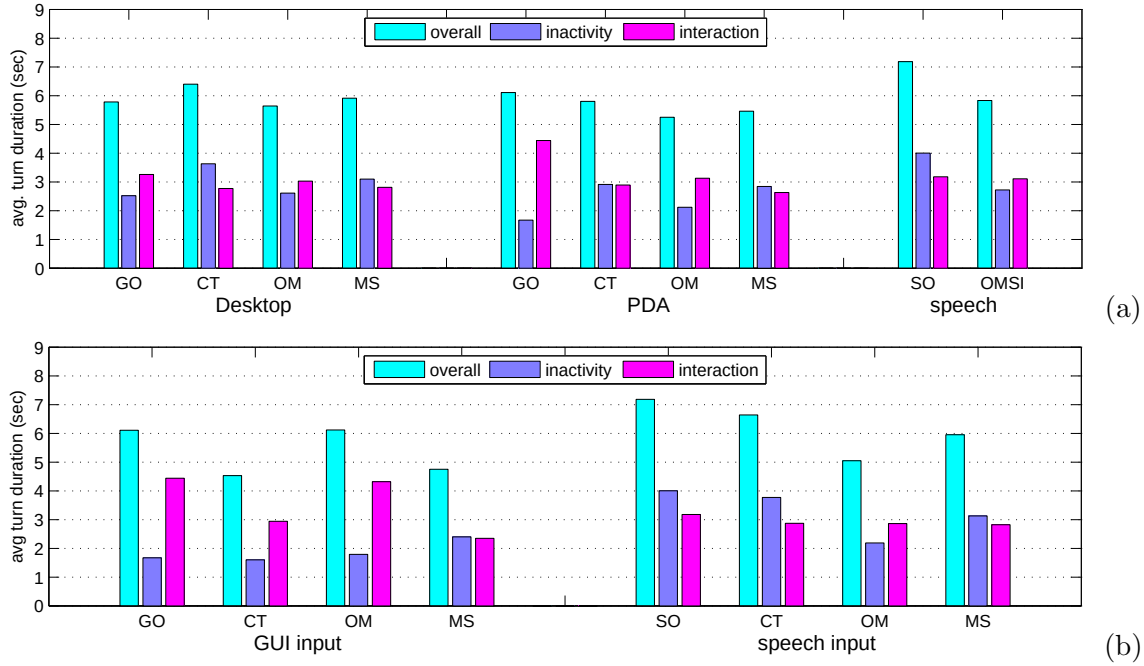


Figure 5.2: (a) Average turn duration (in sec) for all ten systems (four Desktop, four PDA and the two speech-only systems) broken into inactivity and interaction times. (b) PDA inactivity and interaction times grouped by input type (GUI and speech) respectively. Note the "Speech-Only" (SO) system is also included as a reference.

and also roughly the same for all modes, except for the MS mode. Interaction times vary considerably; they are very high for GO (no input modality choice) compared to multimodal modes. Note that for GUI input MS has the highest inactivity but lowest interaction times.

As far as speech input is concerned, one can note almost identical interaction times for the three multimodal modes but highly varying inactivity times. OM has shorter speech inactivity times compared to CT, while MS inactivity times are approximately the average of the other two modes. Comparing GUI and speech input, it is evident that inactivity times are much shorter for GUI input compared to speech input (GUI click vs VAD event).

Table 5.3 shows a summary of objective statistics for the current evaluation. The statistics are reported for all ten systems evaluated (two for speech only systems, four for desktop and PDA environments). Metrics include task completion rate(%), percent of speech turns, task related statistics such as average number of turns per task and average task duration. Turn related statistics such as average turn duration are also reported along with the break down into inactivity and interaction parts.

Table 5.3: Summary of main objective statistics. The second column labeled CR denotes the task completion rate and the third column labeled SU denotes the percentage of speech turns, thus speech usage.

	Overall		Task Average		Turn Average duration(sec)		
System	**CR(%)**	**SU(%)**	**# turns**	**duration(sec)**	**inactivity**	**interaction**	**overall**
Speech system evaluation							
SO	97.5	100	8.69	62.43	4.00	3.18	7.18
OMSI	95	100	8.50	59.09	2.72	3.10	5.83
Desktop evaluation							
GO	100	0	8.68	50.10	2.52	3.26	5.78
CT	100	69	8.08	51.65	3.63	2.77	6.40
OM	100	68	8.10	45.70	2.61	3.03	5.64
MS	100	64	8.35	49.38	3.10	2.81	5.91
PDA evaluation							
GO	100	0	8.50	51.95	1.67	4.44	6.11
CT	100	61	8.75	50.78	2.91	2.89	5.80
OM	100	82	8.93	46.82	2.12	3.13	5.25
MS	100	59	8.65	47.26	2.84	2.63	5.46

5.2.3 Context statistics

Fig. 5.3 shows PDA inactivity and interaction time distributions grouped by input type (GUI or speech) for the four most frequently used attributes (termed as contexts in the course of interaction). Note that inactivity times for speech input are higher compared to GUI ones (Fig. 5.3(a) and Fig. 5.3(c)). As shown in Fig. 5.3(b) GUI interaction times clearly depend on attribute (combo-box) size, as expected, while speech interaction times (Fig. 5.3(d)) are similar for all attributes. Finally, interaction times for speech input are considerably shorter compared to GUI ones for the case of city and airline attributes but slightly longer for date and time attributes.

In contrast with GUI interaction for which only one concept is provided per turn using speech, users can input more than one concepts per turn, e.g. "From New York to Chicago". Table 5.4 shows the average number of concepts provided per turn (speech verbosity) for the various contexts (expected attribute input) grouped by system. In addition, the concept accuracy is shown, defined as the percent of concepts recognized correctly by the system over the total number of concepts uttered by the user. Note that for the city and date attributes verbosity is high, e.g., for the city case, users usually provide both departure and arrival city e.g., "From Las Vegas to Miami". The same holds for date, e.g., "July 25th in the morning". Also note that concept accuracy is high (about 90%) for all attributes except for date[3].

[3]The "date field" consists of two distinct words namely: month and day, e.g., "July 6". A recognition error

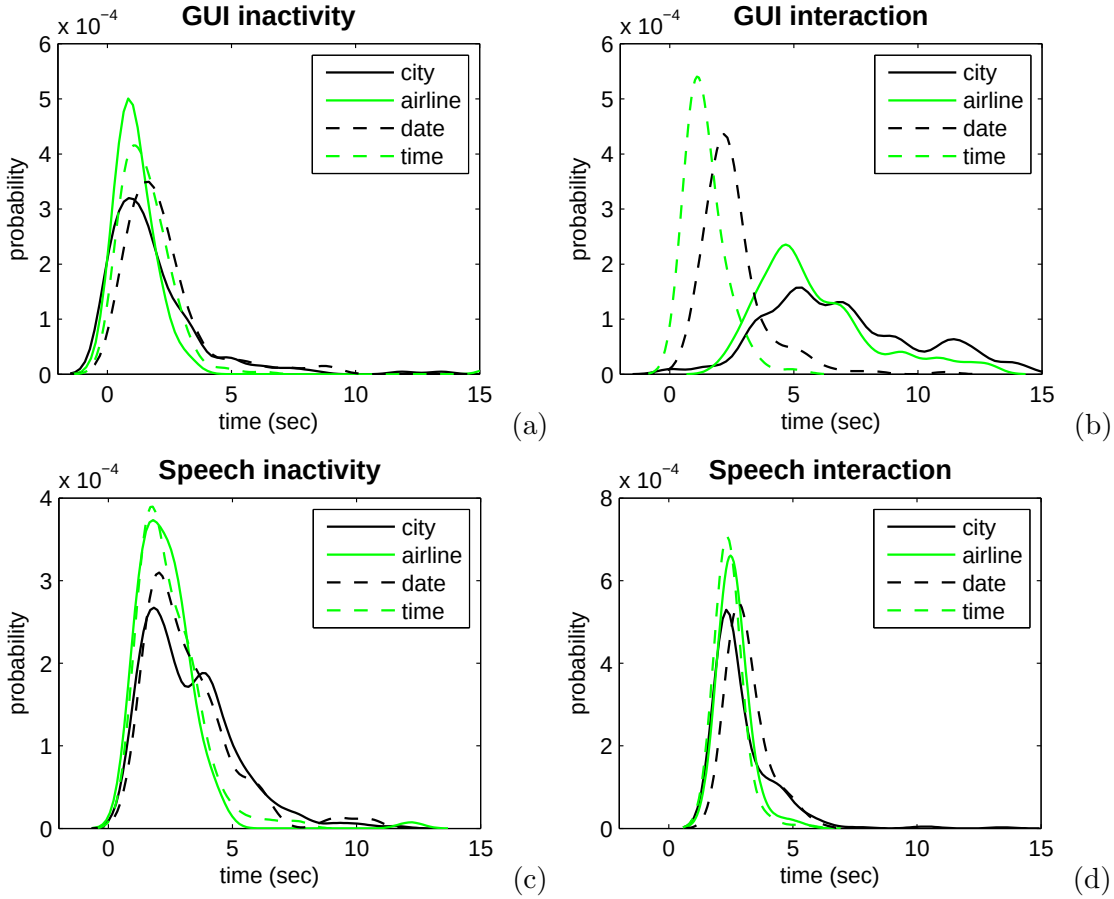


Figure 5.3: Distributions of average turn duration in seconds broken down into inactivity/interaction times and input type (pen/speech) for the four most frequently-used contexts (city, airline, date, time). Results are cumulative for the four PDA systems (GO, CT, OM, MS). Distributions approximated using kernel density functions. (a) Avg. inactivity time distribution for pen input. (b) Avg. interaction time distribution for pen input. (c) Avg. inactivity time distribution for speech input. (d) Avg. interaction time distribution for speech input.

Fig. 5.4(a) shows input modality selection (% number of turns) for the four most frequently used attributes, sorted by size (e.g., 135 for city attribute). Speech usage is fairly high for "long" attributes (between 80% and 90%) and mode-independent. For "short" attributes on the other hand, speech usage is clearly mode-dependent i.e., for the time attribute it is 80% for OM, 35% and 25% for CT and MS respectively.

for either word would result in an error for the "date" concept. Note that although the number of "legal" date values shown at the GUI are only 22, the speech recognition grammar is unconstrained allowing effectively 365 values.

Table 5.4: Speech input context statistics: concepts per turn (verbosity) for the three PDA multimodal systems and averaged % concept accuracy for four contexts.

	CT	OM	MS	
context	**verbosity**			**% concept accuracy**
city	1.71	1.52	1.54	92
date	1.35	1.31	1.34	65
time	1.05	1.00	1.00	92
airline	1.00	1.00	1.00	88

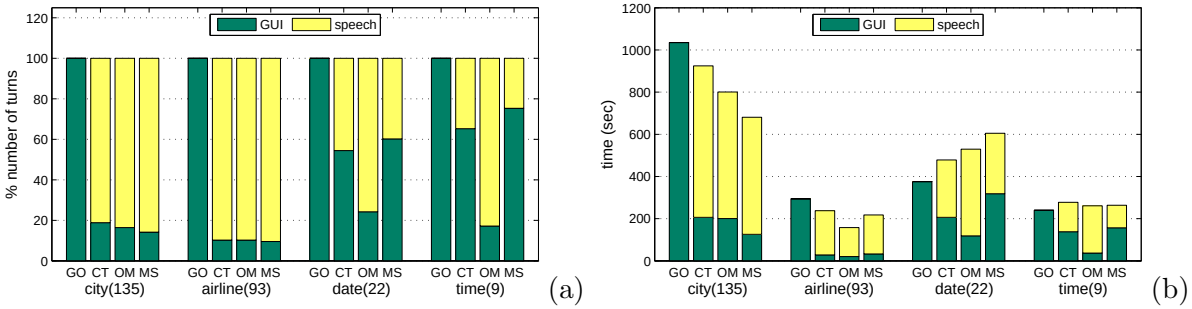


Figure 5.4: PDA context statistics for the four most important attributes (a) percent number turns and (b) overall user time.

As shown Fig. 5.4(b) (user times for the same four attributes) multimodal interaction for all three modes is much faster compared to GO mode regarding "long" attributes. For "short" attributes (date and time), however, multimodal modes perform worse compared to GO mode.

5.2.4 Input modality overrides

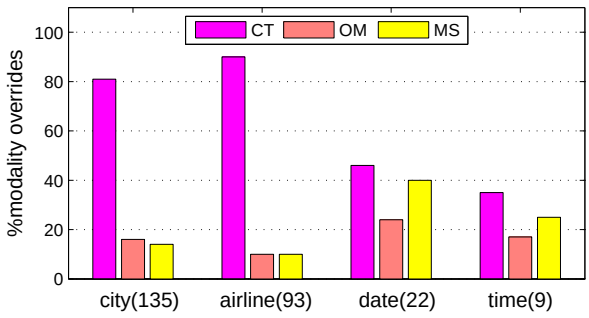


Figure 5.5: Input modality overrides (%) for the three PDA multimodal modes (CT, OM and MS) grouped by attribute type (attribute size included in parentheses).

Fig. 5.5 shows the (%) number of default input modality overrides for the three PDA multimodal modes; the four most important attributes are shown, sorted by size. For CT the

number of overrides (use of speech instead of GUI input) is very high for "long" attributes where users preferred to override default GUI modality in favor of speech. For OM the number of overrides is the lowest. Very few overrides occur for "long" attributes and slightly more for "short" ones. Finally, although MS has fairly low percent of overrides (use of GUI instead of speech input) for "long" attributes, percent of overrides for "short" attributes (use of speech instead of GUI input) is higher (between those of CT and OM). As a result MS has slightly more overrides compared to OM. Overall, OM has the least number of overrides, closely followed by MS and then CT where a very high number of overrides occurs.[4]

5.2.5 User statistics

Fig. 5.6(a) shows total task duration for the three multimodal and the two unimodal GO and SO systems per user (PDA evaluation scenarios). Task duration per user is further broken down into duration of GUI-input and speech-input turns. Note how unimodal performance for both GO and SO interaction modes highly varies between users. Also note that for almost all users, multimodal interaction modes are more efficient compared to at least one of the unimodal interaction modes, and for some of the users such as usr2 and usr7, they are much faster than both unimodal interaction modes.

Fig. 5.6(b) shows number of turns and Fig. 5.6(c) shows average turn duration by averaging all three multimodal modes for the PDA case. For speech input turns, average turn duration is between 5 and 6 secs, while for GUI input turns, average turn duration is between 3 and 8 secs. Thus variability in duration (or variability in efficiency) among users is much higher for GUI input compared to speech input.

There is also high variability in the number of turns; the total number of turns depends on both the percentage of GUI and speech turns and the combination of speech verbosity and concept accuracy (thus the number of correct concepts per turn). It was found that concept accuracy varies between users from 75% to 94% while verbosity (number of concepts supplied per user turn) varies from 1.05 to 1.52. Note that some users (usr1, usr5 and usr6) completed all scenarios with less than 126 turns (more than one *correct* concept per turn for speech input), while others needed considerably more turns.

5.3 Subjective evaluation

In Table 5.5, the overall subjective evaluation scores are shown for all ten systems. In the last two rows the mean and standard deviation are also reported. The overall scores were supplied

[4]Note that override results were presented as a % of the input turns; one has to also consider the relative usage of the four attributes in Table 5.2. Also, the cost of overrides may not be the same for all cases.

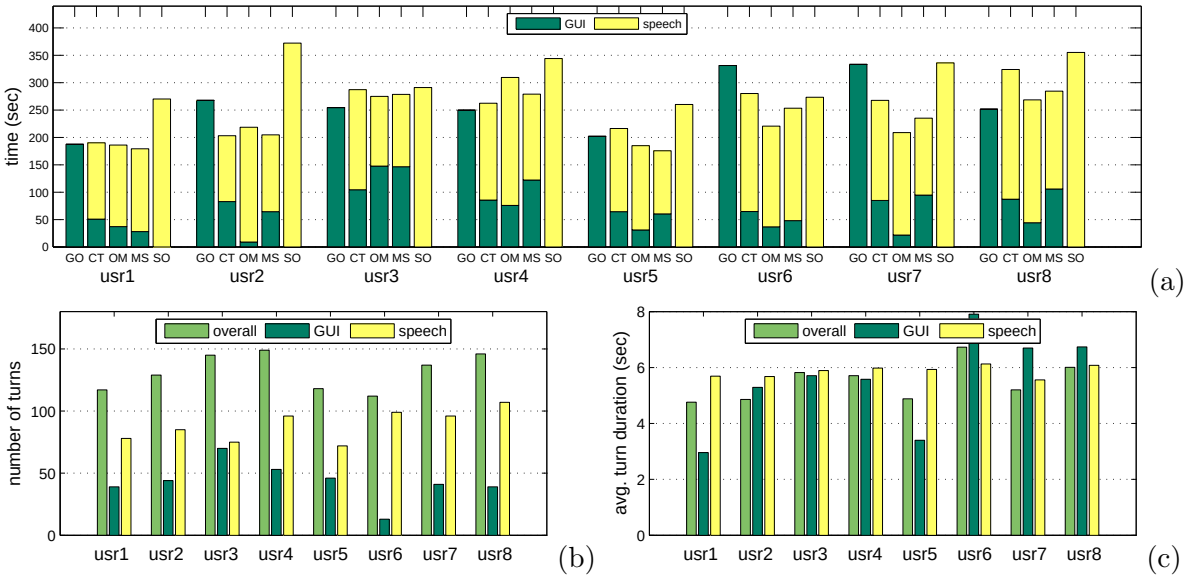


Figure 5.6: PDA user statistics. (a) total time to completion for the multimodal and GO systems (b) sum of number of turns for the three multimodal modes (c) average turn duration for all three multimodal modes.

by the users after the exit interview. A within subjects ANOVA shows that the evaluated systems differ ($F_{9,63} = 6.08$, $p < 0.001$). A post-hoc Tukey HSD test ($p < 0.05$) was performed to find any significant differences.

Results show that while SO significantly differs with all other systems, OMSI only differs with the OM and MS for both desktop and PDA systems, which got the highest ranking overall. The only significant differences for the desktop environment are among GO with OM and MS systems and for the PDA environment among the GO and OM system. Further analysis shows that the correlation between time-to-completion and overall subjective evaluation scores for the ten systems is relatively high (-0.43).

5.4 Discussion of objective and subjective results

The results in Fig. 5.2(a) clearly show the importance of having "visual feedback" in a spoken dialogue system. By incorporating visual output to OMSI the efficiency increases dramatically (inactivity time decreases by 1.3 secs) compared to the SO system. "Input modality choice" also plays an important role; note the decrease in interaction time between GO and multimodal modes, e.g., MS for the PDA case. By offering the users the freedom to select the most efficient input modality in any given context, interaction time can be shortened considerably. This is especially true for the "long" attributes (city and airline) in the PDA case for which speech

Table 5.5: Subjective evaluation results

Platform	**desktop**				**PDA**				**speech**	
System	**GO**	**CT**	**OM**	**MS**	**GO**	**CT**	**OM**	**MS**	**SO**	**OMSI**
Usr1	9	10	10	9	9	9	10	9	7	10
Usr2	8	7	10	10	10	10	10	10	6	10
Usr3	10	7	8	7	7	9	10	8	4	5
Usr4	6	7	7	8	7	8	9	8	6	5
Usr5	8	8	10	9	8	9	10	10	7	8
Usr6	6	10	10	10	9	10	10	9	7	8
Usr7	8	9	9	10	8	9	9	10	8	9
Usr8	8	9	9	10	7	8	8	9	8	9
Mean	7.88	8.38	9.13	9.13	8.13	9	9.5	9.13	6.63	8
StDev	1.28	1.13	1.07	1.12	1.07	0.76	0.73	0.83	1.29	1.83

input is much faster compared to GUI input.

5.4.1 Multimodal interaction modes

Among the three multimodal systems the CT system is clearly the least efficient. This is due to inefficiencies of this mode for speech input; observe the high inactivity times in Fig. 5.2(b) combined with the relatively high percent of speech usage (see Fig. 5.5).

From Fig. 5.2(b), one can observe that GUI input has on average lower inactivity times, while speech input has lower interaction times. Although speech is the most efficient in terms of input (interaction times), recognition errors and context switching incurs higher cognitive load to the user resulting in higher inactivity times for speech input.

The "adaptive" MS system, which at each turn suggests to the user the most efficient input mode, has the shorter interaction times, however it typically has high inactivity times. This is due to the increased cognitive load that adaptivity incurs on the user; automatically switching between interaction modes (CT/OM) and thus default input modality is sometimes inconsistent and confusing. This is a common problem of adaptive interfaces.

Given that speech input usage was much higher in our current evaluation compared to GUI input, it is no surprise that OM is faster than CT; MS being a mixture of the other two multimodal modes, has efficiency that lies somewhere between the efficiency of the other two modes.

5.4.2 Modality usage patterns

The interaction mode statistics results in Fig. 5.1(b) clearly show that the multimodal system biases the input modality usage (CT vs. OM). Users tend to use GUI input more often when it is the default input mode (in CT), compared to the OM system where speech in the default

input mode.

In Fig. 5.3(b), one can see that the mean interaction times for GUI input are shorter for attributes with fewer options in the combo box, as expected. For speech input, the PDFs shown in Fig. 5.3(d) are very similar for all attributes. Comparing the interaction times per attribute, it is clear that GUI input is more efficient for "time" and "date", while speech input is more efficient for "city" and "airline".

Based on the observation above and given the almost 50-50% balancing between "GUI-efficient" and "speech-efficient" attributes in the scenarios, one would expect a 50-50% input modality usage split between GUI and speech. However, the results show that for all multimodal systems speech input is used for over 60% of the turns. A possible explanation for this, is that we have an asymmetrically balanced situation; that is, although our scenarios are almost balanced in terms of number of turns (number of "long" vs "short" attributes), the difference in unimodal efficiency (GUI vs speech) for the "long" and "short" attributes is not symmetrical. Difference in efficiency between GUI and speech is much higher for "long" attributes (in favor of speech) compared to "short" ones (in favor of GUI) as shown in Fig. 5.3(b) and Fig. 5.3(d). Additionally users are aware of the relation of GUI efficiency with attribute (combo-box) size; however such a relation is not clear for speech input. Speech errors also affect input modality selection. This can be clearly seen in Fig. 5.4(a) where users use GUI input for "long" attributes, mainly to correct speech recognition errors.

Subjective results show that users prefer multimodal modes compared to unimodal ones. Users seem to value both the visual feedback (OMSI vs SO) and the input modality choice offered by the multimodal modes (multimodal vs unimodal). Although the correlation between user times and subjective scores is high, other factors also affect users mode preferences.

5.4.3 User variability

From Fig. 5.6 and Table 5.5 it is clear that the user patterns vary significantly as far as unimodal efficiency, modality selection and subjective scores are concerned. The users display significant differences in efficiency for GUI input and (expected) differences in efficiency for speech input (due to different speech recognition error rates). The users also differ significantly in input modality usage and preferred interaction mode. The high variability in user patterns shows that a "stereotypical" modality selection model (such as the one implied by the MS interaction mode) might not model adequately user modality preferences.

Overall, combining multiple modalities efficiently is a complex task that requires both good interface design and experimentation to determine the appropriate modality mix. From the analysis of the relative efficiency of the input modalities and from the modality usage results, it is clear that a relationship between input modality selection and interaction mode efficiency

exists but is not perfectly linear.

5.5 Relative Modality Efficiency and Synergy evaluation

In this section the two proposed metrics are put into test. Again the presented results are for the PDA evaluation.

5.5.1 Relative Modality Efficiency and Modality Selection

In Fig. 5.7, relative speech modality efficiently is plotted against relative speech usage (in terms of number of turns, see Eq. 4.4). There are three free variables in these plots, namely, interaction mode (CT, OM, MS), interaction context (city, airline, date, time) and user (u1 to u8). In all plots, a dashed line (y=x) is used to help identify efficient behavior, i.e., modality usage that is proportional to the modality efficiency. Correlation between modality efficiency and modality usage is indicated with a solid line in each plot. Note that in almost all cases, the linear regression line is located higher than the dashed line, indicating an "overuse" of the speech modality by the users, i.e., a "speech bias".

As shown in Fig. 5.7(a) there are quite large differences in relative speech efficiency between short (time, date) and long attributes (airline, city). This is expected due to the large number of options available for long attributes in the GUI combo-box and vice-versa for short attributes. For both short and long attributes there is a clear bias towards the speech modality. As a result, users choose speech more frequently over pen input, e.g., even for the date field, despite the fact that pen input is more efficient in this case. In Fig. 5.7(b), results are shown for the three multimodal interaction modes. All three modes display speech bias, especially MS and CT modes, which have relative speech efficiency less than 50% and speech usage around 60%. In Fig. 5.7(c), the combined data points for interaction modes and contexts over all users are shown. Note that for the two long attributes (city and airline) speech usage is very high (ranging from about 80% to 90%) as expected, regardless of interaction mode. On the other hand, for short attributes (date and time) *interaction mode clearly affects input patterns.* For CT and MS modes the data points are near the dashed line as expected, however, for OM mode speech usage is very high (much above 70%). Thus, the default input modality (speech in this case) biases users away from efficient modality selection.

User behavior is shown in Fig. 5.7(d). Note that with the exception of users u3 and u6, the rest have relative speech efficiency ranging between 33% and 50% and a corresponding speech usage between 60% and 75%. All users with the exception of u3 display a speech bias. Users u1, u2 and u5 display the least speech efficient behavior. Fig. 5.7(e) shows the combined data points for interaction contexts and users over all modes. For long attributes, with the exception

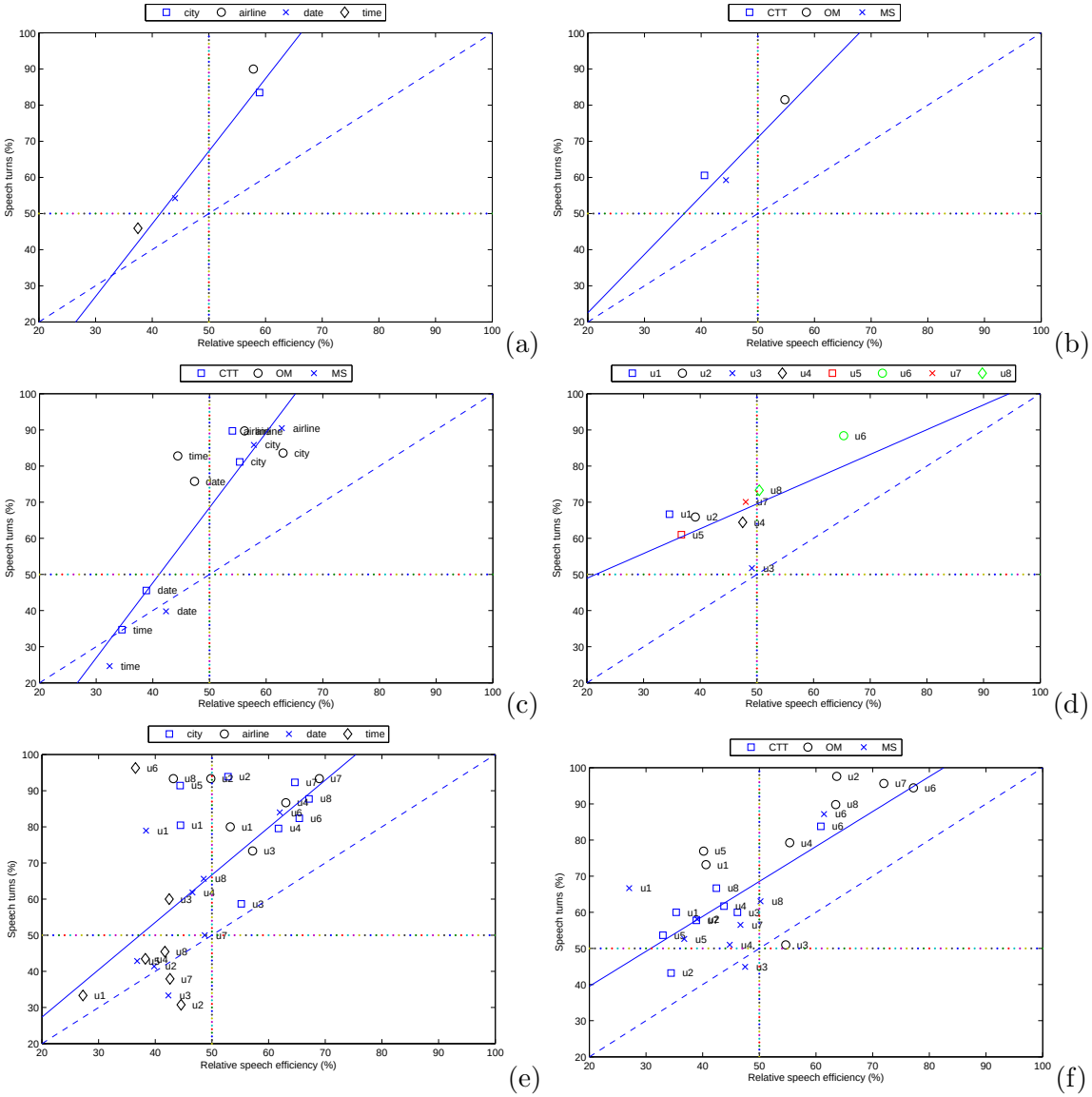


Figure 5.7: Speech modality usage (QU_s) as a function of relative speech modality efficiency - overall times are shown. (a) context averaged over users and interaction modes (4 points). (b) interaction mode averaged over users and contexts (3 points). (c) combined data points for interaction modes and contexts over users (12 points). (d) user averaged over contexts and interaction modes (8 points). (e) combined data points for users and context over interaction modes (32 points). (f) combined data points for modes and users over contexts (24 points).

of point (city, u3) speech usage ranges between 74% and 95%. For the time attribute, with the exceptions of u3 and most notably u6, speech usage is below 50% as one would expect. For the two short attributes, only three users are GUI biased. The data points demonstrate *a "non-linear" user behavior; users abruptly switch from GUI to speech when speech becomes*

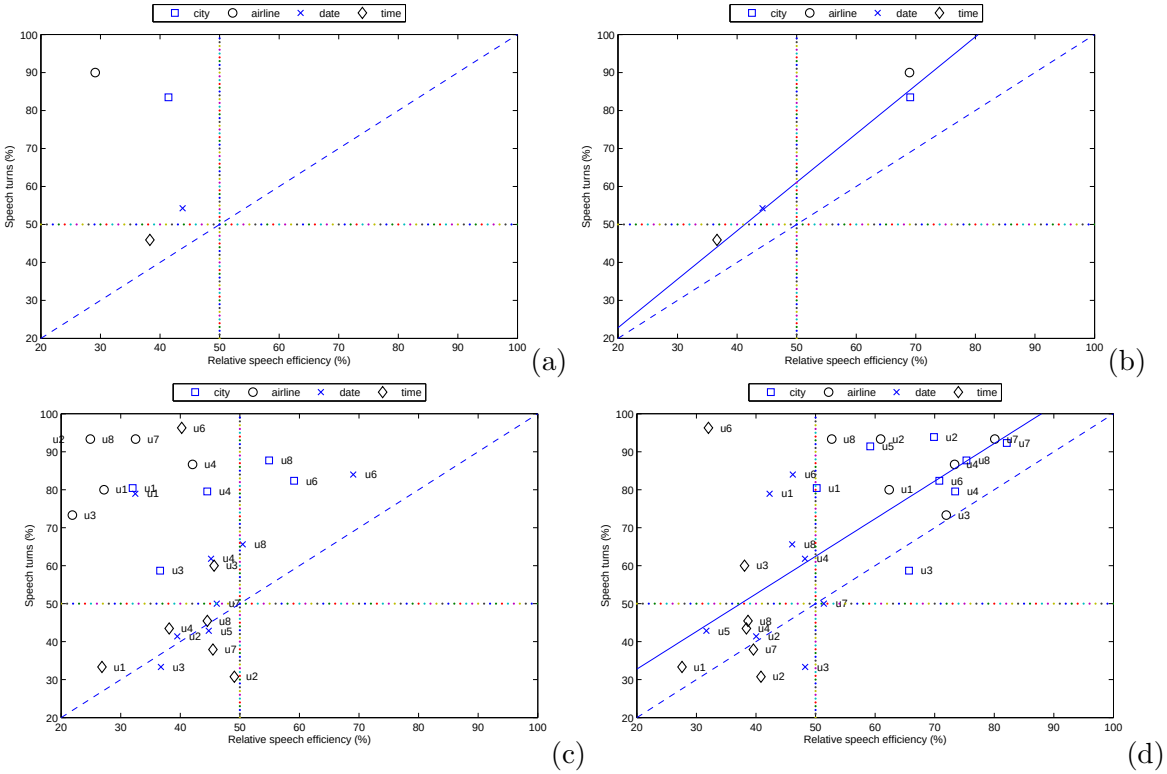


Figure 5.8: Speech modality usage as a function of relative speech modality efficiency. Context (a) inactivity times and (b) interaction times. Combined data points for users and context over interaction modes (c) inactivity times (d) interaction times.

more efficient. Two important observations are that: (i) the switching point is around 45% speech efficiency rather that 50% demonstrating a speech bias, and (ii) in the area of equal modality efficiencies there is high variability in modality usage demonstrating the uncertainty of the user over which modality is more efficient. Finally in Fig. 5.7(f), the combined data points for interaction modes and users are shown over all contexts. For OM mode, half of the users have a speech usage between 88% to 98%; they use speech almost exclusively. User u3 is a notable exception, having speech usage close to 50% and being somewhat GUI biased while the rest have speech usage between 70% and 80%. This high diversity in OM mode which clearly favors speech input, can be attributed to individual speech recognition accuracies and speech verbosity (which varies considerably among users).

Up to this point the main focus has been on overall times. The same analysis conducted for inactivity and interaction times is shown in Fig. 5.8. Plots (a) and (b) correspond to Fig. 5.7(a) and (e) respectively; plots (c) and (d) correspond to Fig. 5.7(e). Fig. 5.8(b) concerning interaction times is quite similar to Fig. 5.7(a) except for a non-linear scaling effect on the x-axis. This effect is due to the incorporation of inactivity times in Fig. 5.8(a). In general: (i) for

Mode	click-to-talk	open-mike	modality-selection
inactivity	-2.6	25.5	0.0
interaction	24.0	17.8	31.0
overall	12.7	21.1	17.8

Table 5.6: Multimodal synergy(%) for the three multimodal interaction modes.

context	city (135)	airline (93)	date (22)	time (9)
inactivity	-8.1	21.6	4.9	24.9
interaction	33.1	31.5	6.6	10.3
overall	18.7	27.6	5.8	18.4

Table 5.7: Multimodal synergy(%) for the four contexts

interaction times, speech bias is less compared to overall times, and (ii) for inactivity times, behavior is less efficient compared to interaction and overall times. For more detailed results concerning inactivity and interaction times, refer to Section B.2.

5.5.2 Multimodal Synergy

As shown next, the achieved synergy is both context, interface dependent and user-dependent. For computing the synergy Eq. 4.6 is used.

In Table 5.6, the synergy between the speech and GUI modalities is computed for the three multimodal interaction modes. For interaction times, MS mode has the higher synergy (31%) followed by CT and then OM modes of interaction. This means that for the MS mode, users selected input modality, based on unimodal efficiency consideration most of the time compared to, e.g., OM mode[5]. As far as inactivity times are concerned, OM which by design favors speech modality choice has low inactivity times. In contrast, high use of speech in the other two modes, results high inactivity times and thus very low synergy (-2.6 for CT, 0 for MS). The *low inactivity synergy for CT and MS modes demonstrate increased cognitive load and time lost to modality switching.* For overall times, synergy is higher for OM mode, followed by MS and then by CT modes. Overall synergy, can be generally thought as a weighted average of the synergies of inactivity and interaction times.

In Table 5.7, the synergy between the speech and GUI modality is computed for the four attributes. As far as interaction times are concerned, there is a clear separation of long and short attributes. Users exploit modality selection to use speech input in favor of pen input for the two long attributes, since as shown in Fig 5.7(a) the relative speech efficiency is close

[5]Recall that for OM users used speech much more often (see discussion on speech overuse regarding Fig. 5.7(d)).

User	u1	u2	u3	u4	u5	u6	u7	u8	mean	std
inactivity	16.4	21.4	8.4	-21.1	-2.7	9.6	24.8	2.5	7.4	14.7
interaction	26.5	33.2	15.5	30.5	17.2	14.4	39.0	13.4	23.7	9.85
overall	22.8	28.2	12.5	11.0	10.0	12.0	32.5	8.2	17.2	9.33

Table 5.8: Multimodal synergy(%) for the eight users

Time	Mode	u1	u2	u3	u4	u5	u6	u7	u8	mean	std
inactivity	CT	22.6	22.5	-13.1	-19.8	-29.6	-0.2	3.5	-8.2	-2.8	18.8
	OM	29.3	25.0	29.1	-16.0	23.5	30.2	48.6	27.5	24.7	18.2
	MS	-5.2	16.8	6.5	-27.8	-0.8	-0.0	21.7	-12.1	-0.1	15.8
interaction	CT	22.8	38.5	16.1	32.9	21.3	2.3	38.8	13.1	23.2	12.9
	OM	24.5	21.7	10.8	24.1	6.5	9.5	34.6	5.9	17.2	10.5
	MS	32.9	38.9	19.9	35.1	23.8	30.4	43.5	21.7	30.8	8.5
overall	CT	22.7	31.8	3.6	12.9	2.8	1.1	22.7	2.9	12.6	11.8
	OM	26.2	23.1	18.6	9.0	12.7	19.9	41.0	16.3	20.2	9.8
	MS	19.1	29.6	14.2	11.3	14.9	15.1	33.6	5.5	17.9	9.4

Table 5.9: Multimodal synergy(%) for the three multimodal interaction modes and eight users

to 60%. In contrast to long attributes, for which synergy is above 30%, synergy for short attributes is much lower since users overuse speech input despite being less efficient, compared to pen input. For inactivity times, there is high synergy for airline and time attributes but low and negative synergy for date and city attributes respectively.

In Table 5.8, the synergy between the speech and GUI modality is compared across the eight users. The mean and standard deviation for synergy across users is shown in the last two columns. For interaction times all synergies are positive and for some users quite high, e.g., 39% for u7. The variability of interaction synergy is high among the users. For inactivity times, one can also note high variability among users. Some users even show negative synergy, such as u4 and u5, demonstrating high cognitive load. Overall time synergy results, show that users helped by system design, can improve considerably their performance compared to unimodal systems.

In Table 5.9, the synergy between the speech and GUI modality is compared across the eight users and the three multimodal modes. The mean and standard deviation for synergy across users is shown in the last two columns. Note that in contrast to OM mode, CT and MS have almost zero mean synergy as far as inactivity times are concerned. As far as interaction times are concerned, OM has lower synergy compared to CT and MS modes. Again note the disparities among users.

5.6 Discussion of results for the new metrics

5.6.1 Context

As far as interaction times are concerned, the "input modality choice" synergy is more clearly pronounced in the case of context results shown in Table 5.7 and Fig.5.8(b), for which differences in unimodal efficiency are quite large, especially for the long attributes. This causes a clear decision on behalf of the users regarding modality choice; users almost always use speech input, except in the case of speech recognition errors for which they use GUI input. In contrast, for short attributes, relative speech efficiency is closer to the 50% decision line, thus making more blurry the modality choice decision. As a result, one can note speech overuse for short attributes (Table 5.7 and Fig 5.8(b)).

As far as inactivity times are concerned (Table 5.7), synergy is negative for "city" attribute and only about 5% for the "date" attribute. The first is due to the fact that "city" is the first attribute users have to fill in a series of forms, which often requires the pre-reading of all form attribute values (cognitive load). For the "date" attribute, it can be attributed to the fact that the default modality changes from speech to GUI in MS mode, causing increased cognitive load to the users.

5.6.2 User Variability

The variability of interaction synergy (Table 5.8) is high among the users, indicating that multimodal modes may not serve equally well all users. Note that user synergy expresses the percent improvement of combined modality usage over unimodal user efficiency. This doesn't mean that for example, u7 (interaction synergy 39%) is the faster user (see Fig. 5.6(a)); it means that during multimodal interaction, that user exploited input modality and other synergies in a higher degree, that helped him improve his performance with the system more, compared to other users. The differences in synergy are due to user dependent input modality usage, variable speech recognition rates, variable number of concepts per utterance for speech input, variable ability/experience using pen input on the PDA and most-importantly, to what degree users used efficiency considerations when selecting the input modality at each part of the interaction[6].

In any case, the fact that synergy is highly user-dependent shows that there is potentially high-reward in designing multimodal interfaces that *adapt to the user*. Creating multimodal interfaces that are "optimal" for a stereotypical user does not grep all the reward (in terms of synergy) over unimodal interfaces. Multimodal dialogue interfaces will not work equally for all users. Just as is the case for unimodal spoken dialogue systems, there might be some

[6]This last factor is directly related to synergy, random input modality selection achieves zero synergy.

users for which one or more modalities might not work well, or the ability of the user to maximize modality synergy might be limited (these users are referred to as "goats" in the speech recognition slang). Some of these shortcomings might be cured over time with training, but clearly multimodal interfaces will not work for everyone, right from the start.

Finally, although multimodal synergy computes the improvement in multimodal interaction due to combined synergies, it would be interesting if one could measure the improvement due to each synergy (modality choice, visual output, error correction) separately. This is not a simple task, since for example error correction and modality choice are closely related. Also there are negative synergies such as the modality selection process overhead. As has been shown in the results visual output can be estimated indirectly by comparing the SO and OMSI interaction modes.

Chapter 6

Usage Patterns and Input Modality Prediction

As has already been discussed in the previous chapter, there is significant variability in user behavior and more interestingly in modality selection patterns (as expected) among the users. The reason for this, mainly stems from differences in unimodal modality efficiency between user exhibits but is not the only factor. A more detailed investigation of individual user behavior is provided here, in section 6.1. Two important factors that affect modality usage and related to speech modality, namely speech verbosity and speech error correction patterns are discussed in section 6.2. Next statistical models for predicting input modality selection are described and evaluated in sections 6.3-6.5. Discussion of results are provided in section 6.6.

6.1 Modality usage patterns

Fig. 6.1 shows input modality selection (as % number of input turns) for the three multimodal interaction modes (CT/OM/MS), evaluated for the PDA device; the four most frequently used attributes are shown, sorted by size (e.g., 135 for city attribute). Speech usage is fairly high for "long" attributes (between 80% and 90%) and mode-independent. Due to large difference in unimodal efficiency between speech and GUI, users prefer speech input unless they need to correct speech recognition errors, in which case they might use GUI input. For "short" attributes on the other hand, speech usage is clearly mode-dependent i.e., for the time attribute it is 80% for "Open-Mike", 35% and 25% for "Click-to-Talk" and "Modality-Selection" respectively. Thus for short attributes there is a clear bias towards speech modality usage as far as "Open-Mike" is concerned. For the other two multimodal interaction modes GUI usage is above the 50% line, which can be attributed to input efficiency considerations.

As has already been discussed, there is significant variability in modality selection patterns

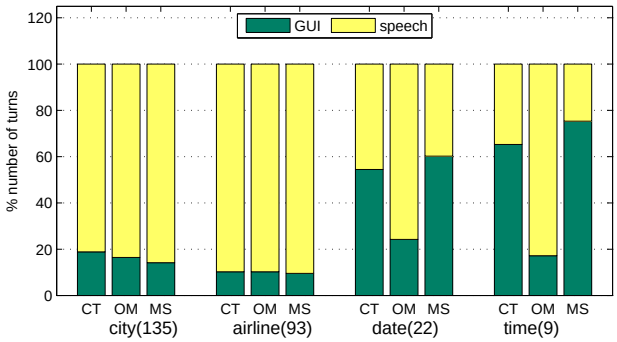


Figure 6.1: Modality selection usage (context) statistics for the three multimodal PDA systems (CT, OM, MS); the four most important attributes are shown as % number turns.

(as expected) among the users. The reason for this, stems mainly from the unimodal modality efficiency each user exhibits but is not the only factor. To better investigate and justify individual modality usage, a more detailed analysis follows based on Fig. 6.2 which shows modality selection usage (context) statistics for each of the eight users.

For the two long attributes (city and airline) GUI usage is very low (mainly used to correct speech recognition errors as explained above) for all users; the main exception is users u3 who exhibits high speech recognition error rates. For short attributes (Fig. 6.2):

- GUI usage in OM is very close to zero for both date/time attributes for most users with the exception of users u1, u3 and u5 (Fig. 6.2(a),(c),(e)). In contrast with the modality usage for long attributes, high GUI usage in the case of short ones is a combination of both speech recognition errors (case of u3 for date attribute) and speech bias/modality efficiency; for example user u5 prefers GUI input because it is faster, despite the fact he has very high speech recognition accuracy.

- Generally GUI usage for time attribute is higher compared to date attribute (except for user u2).

- User u6 (Fig. 6.2(f)) is clearly an outlier; GUI usage is very close to zero even for CT/MS modes. All in all, this user seems to have used GUI only to correct errors (having ASR WER/CER of 90%/93% respectively and speech verbosity of 1.35)

- User u3 is also an outlier (having ASR WER/CER of 72%/75% respectively and speech verbosity of 1.05) he is one of the few users to have higher GUI usage for date compared to time.

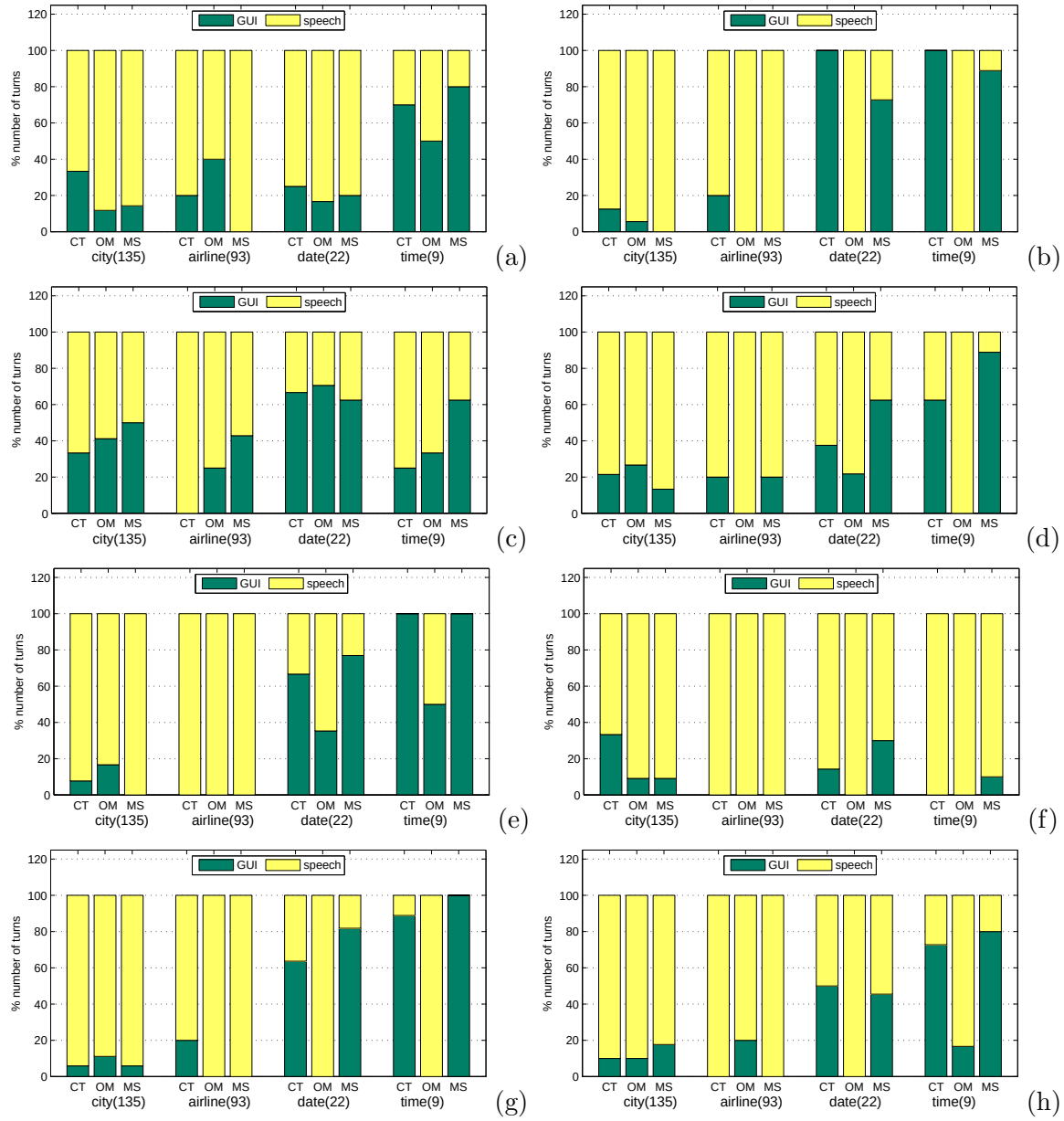


Figure 6.2: Modality selection usage (context) statistics for each of the eight users, for the three multimodal PDA systems (CT, OM, MS); the four most important attributes are shown as % number turns. (a)-(h) corresponds to users u1 ... u8 respectively.

6.2 Speech verbosity and error correction patterns

Two important factors that affect modality usage are related to speech modality. In contrast to GUI input that has almost zero error rate and allows for only one concept input at a time, speech input is more complicated. First it allows the user to input several concepts at a single

turn with the system. Also, since speech recognition accuracy is not perfect, errors may occur that change the user behavior. As a result, speech input efficiency (number of correct concepts per time unit) is a combined result of both speech verbosity and speech errors. As shown in a previous discussion at section 5.2.3 and Table 5.4 both verbosity and concept accuracy depend on context (but also on interaction mode and user). These two factors and their relation to modality selection is discussed next.

6.2.1 Speech verbosity patterns

As has been shown in various literature studies, speech expressiveness differs between human-human communication, human-machine speech-only communication and in multimodal systems with speech support [13, 14]. For example, it was found that in "Speech-Only" system users respond to system often with more rich and expressive language ("I would like to flight from New York to Chicago please") compared to the multimodal interaction modes ("From New York to Chicago"). This may be attributed mainly to prompt design (more verbose for speech-only systems compared to multimodal interaction modes).

The analysis conducted has shown that some users tend to use the multimodal system as a speech-only system with GUI correction support (that is they still speak with relatively high verbosity), while others tend to use them as GUI systems with speech input support (that is, verbosity near 1). Some of the verbosity patterns noticed during the PDA evaluation are described next:

- *departure.city* → *arrival.city* → *date* → *time*: e.g. "From New York to Chicago July 13th in the morning". This verbosity pattern indicates a user trying to fill the whole form at once. Although not used often in multimodal interaction it does however shows users willing to use the system as a speech-only system; such users have already high confidence of successfully using speech and/or correcting speech errors if they happen. Such user is user u6, who has a concept accuracy of 93%.

- *departure.city* → *arrival.city* → *date*: Used more frequently than the previous pattern, shows users eager to input several concepts in one turn such as users u1 and u6, who both have concept accuracy above 90%

- *departure.city* → *arrival.city*: The most frequent pattern used by almost all users due to close semantic relation between departure and arrival cities.

- *date* → *time*: A less frequent pattern, mainly because the date attribute already requires two words and has high CER.

The most common action is to input only one concept per turn; this is the main pattern of use for attributes that are not related with previous input, e.g. when the dialogue state goes to hotel form and asks for the "hotelname" attribute. It is also a common pattern for input to other attributes with high verbosity (such as *city*), by users with high WER/CER who hesitate to input more than one concepts and have an overall low verbosity. Such users are u1 and u7, who have verbosity of exactly one; these users have the lowest concept accuracies (75% and 78% respectively).

6.2.2 Error correction patterns

Since having speech recognition errors in any speech interaction is unavoidable, it is interesting to investigate how users cope with such errors when they happen. While in unimodal speech interaction systems it may be hard to correct errors, causing interaction sessions to even fail and leading to user frustration, in multimodal systems error correction may be as simple as using an alternative less error prone modality such as GUI input (such as in the case of the system described in this thesis). Robustness is arguably one of the main benefits of multimodal systems when they combine a more natural recognition-based inconsistent interface such as speech with a more constrained but consistent interface such as GUI.

Before examining the error correction patterns, it is important to underline the types of errors that may arise when using a spoken or multimodal dialogue system.

- Rejections, e.g. (<city> →<>) for the case of a single attribute; i.e. although the user tried to input a concept, (e.g. Boston), the recognizer understood nothing and asks the user to input the concept again.

- Single attribute (but same concept) error (<city1>→<city2>): e.g. user speaks Boston but the system understands Austin and moves to the next expected input in the dialogue flow. To correct the error the user has to request a transition of the dialogue state to the previous context and input the concept again. This type of error is called *value ambiguity* [5] and is the most common error type encountered.

- Single attribute but different concepts error (<airline>→<city>): e.g. user speaks an airline concept while the system understands a city value; this *semantic or position ambiguity* [5] that might be harder to solve compared to previous kind of errors.

Shown in Fig.6.3 is an example of ambiguity caused by a speech recognition error. The example is taken from the evaluation of an "Open-mike" two way trip scenario. Note that at state number three where user is expected to provide a time concept, the speech recognizer understands *may ninth* instead of *midnight* thus causing ambiguity for the departure date. The ambiguity introduced is reflected in the application tree as follows:

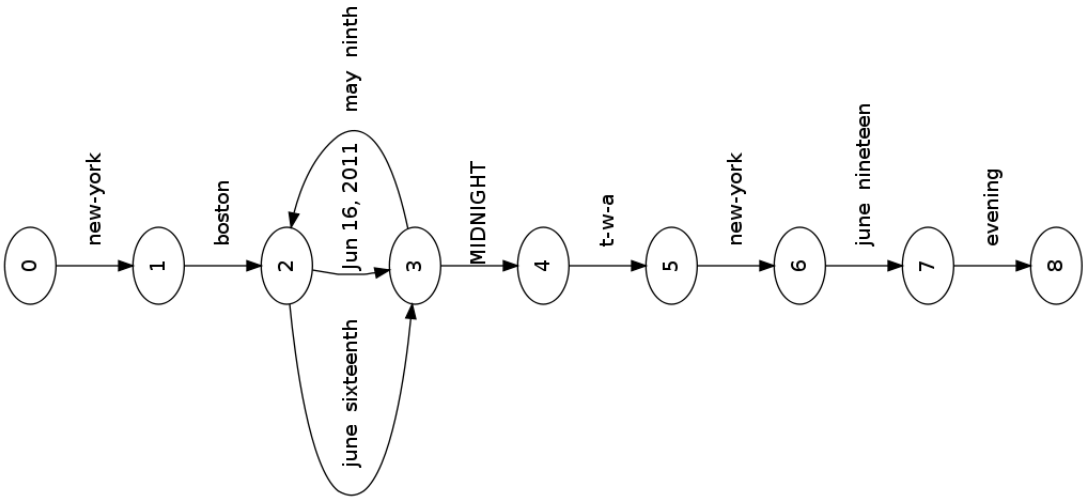


Figure 6.3: Example dialogue flow with speech recognition errors, ambiguity and error correction.

```
S APPLICATION TREE:
  root
  .  trip
  .  .  flight
  .  .  .  leg1
  .  .  .  .  departure
  .  .  .  .  .  city    ( 0.500,  0.755 )      NYC/JFK_LGA_EWR
  .  .  .  .  departure
  .  .  .  .  .  date    ( 0.500,  0.444 )      Jun 16, 2011
  .  .  .  .  .  date    ( 0.500,  0.444 )      May 9, 2011
  .  .  .  .  arrival
  .  .  .  .  .  city    ( 0.500,  0.650 )      BOS
  .  .  .  .  .
```

The user then resolves the ambiguity by using GUI input (*June 16, 2011*) and proceeds to next input field (time) for which he also used GUI input (Fig.6.3). It is also important to note that multiple recognition errors may arise when users input more than one concepts per turn (verbosity > 1) which may even further complicate the error correction process.

Next the error correction patterns found during system evaluation are described. Since error correction is also a concept-input action, it is interesting to investigate if the same concept input patterns apply, as with the case of error free input. For example, do users always use GUI input to correct errors directly or do they also try again with speech input, and for how

many turns until the error is corrected? Also, does the WER/CER or input modality efficiency affect error correction patterns?

Since in terms of the total number of error turns, errors mainly occur for the *date* (65% concept accuracy - see Table 5.4) and *city* (92% concept accuracy but large number of turns) attributes, the error correction patterns are examined separately for these two attributes.

For the *date* attribute, the dominant error correction patterns are :
(a) $Speech \rightarrow Speech \rightarrow GUI$ (13%)
(b) $Speech \rightarrow Speech$ (25%)
(c) $Speech \rightarrow GUI$ (62%)
In the case of the *date* attribute, users try to correct the error using directly GUI input two out of three times (c).

For the *city* attribute, the dominant error correction patterns are :
(a) $Speech \rightarrow Speech \rightarrow Speech \rightarrow GUI$ (6%)
(b) $Speech \rightarrow Speech \rightarrow Speech$ (7%)
(c) $Speech \rightarrow Speech \rightarrow GUI$ (7%)
(d) $Speech \rightarrow Speech$ (40%)
(e) $Speech \rightarrow GUI$ (40%)
It interesting to note that only a 40% of errors are corrected directly through GUI (e). Users take a chance to correct the error using speech input again 60% of the time, the majority of which is a success (d). Note that 14% of the time, users try to correct the previous unsuccessful correction effort using again speech input ((a) and (b) cases).

Overall, results show that both input (GUI) modality efficiency and CER levels affect the error correction strategy. For the case of the short *date* attribute, not only is probability of misrecognition using speech again high, but also GUI is much faster to use for correcting the error; that is why users use GUI input two out of three times to correct errors. For the case of the long *city* attribute on the other hand, the above pattern is used only 40% of the time, because the chance of successfully correcting the error using speech is relatively high and also GUI input is slow compared to speech input.

6.3 Modality prediction based on context and interaction mode

Generally as shown in Fig.6.1 modality selection depends on both the current context and interaction mode for short attributes. For the case of OM, users clearly prefer speech input while for the CT and MS users use slightly more GUI compared to speech turns (selection patterns are not that clear). This makes the modality choice prediction difficult, since it is very close to the 50% decision line for these cases. For the long attributes, the modality patterns are much more clear however; GUI usage is between 10% - 20% (users use speech

mainly to correct speech recognition errors).

6.3.1 Statistical model

Thus a first model that can be used for modality prediction is described by $P(m|c,s)$, that is input estimates based on context and interaction mode, where m denotes the input modality type (GUI or Speech), c the context and s the system used (CT/OM/MS). The model probabilities are estimated using Maximum Likelihood; thus they are estimated by counting the number of times that input m was used when context c occurred during evaluation of interaction mode s. Since as shown in Table 5.2 the majority of turns in the evaluation scenarios are for two long (city, airline) and two short (date, time) attributes, let us use these 4 attributes to compute the model probabilities (the few turns for the other attributes may not provide robust statistics).

GUI selection probabilities for the four attributes over all users are shown in Table 6.1. The values denoted with bold face, are probabilities close to the 50% decision line which are expected to cause the most prediction errors. The application of the model for the modality prediction evaluation is described in the following section.

Table 6.1: GUI selection probability for the $P(m|c,s)$ model

Context/mode	CT	OM	MS
city	0.1885	0.1641	0.1416
airline	0.1026	0.1026	0.0952
date	**0.5444**	0.2424	**0.6022**
time	**0.6528**	0.1719	0.7534

Table 6.2: GUI selection probability for the recomputed $P(m|c,s)$ model with users u3 and u6 removed

Context/mode	CT	OM	MS
city	0.1474	0.1300	0.0909
airline	0.1333	0.1000	0.0333
date	0.5588	0.1622	0.6418
time	**0.8182**	0.1702	**0.8909**

6.3.2 Model evaluation process

Based on the probabilities of the computed model, the process of predicting the output is the following: for each turn conducted in the user evaluation sessions (across all users, scenarios, systems and the four attributes) the classifier chooses GUI input with probability $P(m_i|c_i,s_i)$ according to Table 6.1. Then, the predicted inputs are compared to the real one, performed by

the users, in order to derive the prediction accuracy of the classifier. This process is repeated a large number of times, and the overall mean and standard deviation estimations are computed.

6.3.3 Results

Using as both train and test-set all the evaluation sessions, a large number of times, as described above the prediction accuracy of the model is 76.73%The accuracy of the model is not very high, considering that the same data are used for both training and testing (for results using the leave-one out method see following sections). Apparently, values near 0.5 in Table 6.1 cause a lot of misclassifications; this is the case with *date* and *time* attributes for CT and MS interaction modes.

Clearly one way to increase prediction accuracy would be to decrease the variability between users selection patterns in the train-set. In other words, in the process of building a statistical model such the one that is described above, what would be a good train-set to use? The answer of course is one that maximizes the balance of representing all possible input patterns while keeping enough discrimination power. It is clear from the analysis that users u3 and u6 can be considered as outlier users (section 6.1) since their usage patterns vary considerably compared to the rest users (see also how far these users are from the cluster of the rest six users in Fig.5.7(d)).

In order to test how much the prediction accuracy increases when removing these two users from the train-set, the statistical model $P(m|c, s)$ is recomputed by removing each user u3 and u6 data out of the training set, one at a time and then both. For each of the three recomputed models, the model evaluation process described above is repeated again. The results for the original and three recomputed models are shown next:

```
Using as test-set, train-set all 8 users: accuracy 77%
Using as test-set, train-set all users except u6: accuracy 78%
Using as test-set, train-set all users except u3: accuracy 79%
Using as test-set, train-set all users except u3,u6: accuracy 82%
```

Thus removing users u3 and u6, classification performance increases from 77% to 82%. The recomputed model for the last case is shown in Table 6.2. Comparing the values with that of Table 6.1, one can note that with the exception of *airline* and CT value, all other values have been moved further away of the 50% decision line, thus providing increased discrimination power. Also note that the values changed more in Table 6.2 compared to Table 6.1 are the ones for the time attribute ($0.65 \rightarrow 0.82$ for CT and $0.75 \rightarrow 0.89$ for MS - shown in bold face). Recall from Fig. 6.2(f) the strange modality selection behavior of user u6 who uses almost 100% speech input for the time attribute - removing this behavior improves overall prediction accuracy.

Another question related to the variability of the training data and thus the prediction accuracy is how *consistent* are users' modality choice patterns across turns. That is, do users follow the same consistent modality patterns again and again or not? Users who do, can be classified as highly *predictable* with respect to their modality choices. An important factor (already examined in some respect in the previous paragraphs) is how similar to "common behavior" is the modality choice patterns of a certain user (such user u6 in previous paragraph). Obviously users with consistent behavior and close to "common behavior" patterns of other "mainstream" users will help produce a prediction model of high accuracy.

Table 6.3: Modality prediction classification rate(%) results per user

	self-test	leave-1-out	difference
u1	77	68	-9
u2	**93**	85	-8
u3	**67**	**57**	-10
u4	77	73	-4
u5	86	84	-2
u6	90	**63**	**-27**
u7	91	90	-1
u8	80	80	0

One way to test a user's input *consistency* is to build a prediction model and evaluate it on the user's own collected data (self-test column in Table 6.3). Note in Table 6.3 that with the exception of u3 who has classification rate of only 67%, most users have classification rate $> 77\%$ (some of them above 90% such as user u2).

The second column shows results of the "leave-one-out" method. That is the prediction model has been estimated by using data from all rest users and then tested on that user's evaluation sessions. Note that users u3 and u6 are the ones located far apart from the rest users (also in Fig. 5.7(d)) and having prediction accuracy of 57% and 63% (shown in bold face) respectively. The third column of Table 6.3 shows the difference in prediction accuracy between the two previous methods. Note how large the difference is for u6; for some other users though difference is close to zero, indicating that their modality patterns conform to "common behavior".

6.4 Modality prediction based on context and previous input

6.4.1 Statistical model

Another approach for input modality selection is to assume that the input type in each turn depends on both the current context and the input type used in the previous turn, described by

$P(m_i|c_i, m_{i-1})$. Using Maximum Likelihood(ML) estimation, the probability can be computed by the following equation:

$$P(m_i|c_i, m_{i-1}) \stackrel{ML}{=} \frac{\#c_i, m_{i-1} \rightarrow m_i}{\#c_i, m_{i-1} \rightarrow *} \tag{6.1}$$

That is, the number of times users used input type m_i while in context c_i while in the previous turn used input type m_{i-1} divided by the number of times users used any input type m_i while in context c_i.

Another way to compute the model probabilities is to assume that events c_i and m_{i-1} are independent and that c_i and m_{i-1} are also independent given m_i. Then

$$P(m_{i-1}|c_i) = P(m_{i-1}) \cdot P(c_i), m_{i-1} \perp c_i \tag{6.2}$$

$$P(c_i, m_{i-1}|m_i) = P(m_{i-1}|m_i) \cdot P(c_i|m_{i-1}), m_{i-1} \perp c_i \text{ given } m_i \tag{6.3}$$

and using the following derivations (Bayes rule):

$$P(m_i|c_i, m_{i-1}) = \frac{P(c_i, m_{i-1}|m_i) \cdot P(m_i)}{P(c_i, m_{i-1})} \tag{6.4}$$

$$P(c_i|m_{i-1}) = \frac{P(m_{i-1}|c_i) \cdot P(c_i)}{P(m_{i-1})} \tag{6.5}$$

$$P(m_{i-1}|m_i) = \frac{P(m_i|m_{i-1}) \cdot P(m_{i-1})}{P(m_i)} \tag{6.6}$$

By applying the previous equations, we get

$$\begin{aligned}
P(m_i|c_i, m_{i-1}) &\stackrel{(6.5)}{=} \frac{P(c_i, m_{i-1}|m_i) \cdot P(m_i)}{P(c_i, m_{i-1})} \\
&\stackrel{(6.4)}{=} \frac{P(m_{i-1}|m_i) \cdot P(c_i|m_i) \cdot P(m_i)}{P(c_i, m_{i-1})} \\
&\stackrel{(6.6),(6.7)}{=} \frac{P(m_i|m_{i-1}) \cdot P(m_{i-1}) \cdot P(m_i|c_i) \cdot P(c_i)}{P(m_i) \cdot P(m_i) \cdot P(m_{i-1}, c_i)} \\
&= \frac{P(m_i|m_{i-1}) \cdot P(m_{i-1}) \cdot P(m_i|c_i) \cdot P(c_i)}{P(m_i) \cdot P(m_{i-1}, c_i)} \\
&= \frac{P(m_i|m_{i-1}) \cdot P(m_{i-1}) \cdot P(m_i|c_i) \cdot P(c_i)}{P(m_i) \cdot P(m_{i-1}) \cdot P(c_i)} \\
&\stackrel{(6.3)}{=} \frac{P(m_i|m_{i-1}) \cdot P(m_i|c_i)}{P(m_i)}
\end{aligned} \tag{6.7}$$

Assuming also that $P(m_i) = \frac{1}{2}$ and maximizing the above probability:

$$\begin{aligned} \arg\max_{m_i} \frac{P(m_i|m_{i-1}) \cdot P(m_i|c_i)}{P(m_i)} &= \arg\max_{m_i} P(m_i|m_{i-1}) \cdot P(m_i|c_i) \\ &\overset{ML}{=} \arg\max_{m_i} \frac{\#m_{i-1} \rightarrow m_i}{\#m_{i-1} \rightarrow *} \cdot \frac{\#c_i \rightarrow m_i}{\#c_i \rightarrow *} \end{aligned} \tag{6.8}$$

6.4.2 Results

Table 6.4: $P(m_i|m_{i-1})$ probability

previous input type	current input type	probability
GUI	GUI	0.5
GUI	AUDIO	0.5
AUDIO	AUDIO	0.74
AUDIO	GUI	0.26

Table 6.5: Classification results (%) for $P(m_i|c_i, m_{i-1})$ model

	ML computation	Bayes, independence and ML
u1	67	67
u2	83	83
u3	50	58
u4	71	71
u5	74	77
u6	84	64
u7	82	82
u8	76	76
overall	73	72

Table 6.5 shows the computed $P(m_i|m_{i-1})$ probabilities for each possible combination. Note that the most frequent pattern is using speech input if speech was used in the previous turn too which reflects the high speech usage in data. As shown in Table 6.5 the two models perform similarly except for the case of users u3 and u6, that is the two users having $P(m_i)$ away from 0.5 (see Fig. 6.2). The results indicate that modality prediction using these kind of particular statistical model is not enough to achieve high performance.

6.5 Modality prediction using modality tracking

Another model that could be used is modality tracking. Let's denote m the modality type (GUI/speech) the user selects at each turn, i the modality type proposed by the system and u

the user. Given that $i \perp u$ we get :

$$\begin{aligned} P(m|i,u) &= \frac{P(i,u|m) \cdot P(m)}{P(i,u)} \\ &= \frac{P(i|m) \cdot P(u|m) \cdot P(m)}{P(i,u)} \\ &= \frac{P(i|m) \cdot P(u|m) \cdot P(m)}{P(i) * P(u)} \\ &= \frac{P(u|m)}{P(u)} \cdot \frac{P(i|m)}{P(i)} \cdot Pm \\ &= \frac{P(m|u) \cdot P(m|i) \cdot P(m)}{P(m) \cdot P(m)} \\ &= \frac{P(m|u) \cdot P(m|i)}{P(m)} \end{aligned} \tag{6.9}$$

$$\widehat{m} = \arg\max_m P(m|i,u) = \arg\max_m \frac{P(m|i)P(m|u)}{P(m)} \tag{6.10}$$

This model was not evaluated due to lack of discrimination power for the context statistics.

6.6 Discussion

There are two main issues in building more complex and potentially successful user behavior and prediction models. The first one is to identify all possible factors that may affect user behavior. The second one, is to quantify these factors and estimate their effect on user's behavior and decision making, e.g. modality selection, speech verbosity and error correction patterns. For example, there is strong evidence that such a factor is speech bias, but how to quantify it and what weight should be given to this factor?

Another issue is that modality selection shouldn't be considered in isolation but rather in relation to overriding the default input modality. For example using speech input for *date* attribute in OM mode (where speech is the default input modality) and speech input for CT mode for the same attribute (where GUI is the default input modality and thus user should override the proposed input modality), should not be considered the same. Since the cost of overriding the default input in this case is high in CT mode, perhaps a higher weight should be given to this modality selection action. By taking into account modality overrides information compared to just modality selection alone, more valuable information can be incorporated in a prediction model.

The fact that one of the modalities is speech further complicates modality prediction. This is because speech usage entails both recognition errors and the verbosity feature (both not found in GUI usage) . As speech recognition is inconsistent, there is the additional difficulty

of knowing whether a recognition error has taken place. A possible approach for detecting speech recognition errors would be to use speech recognition's engine confidence scores or other method such as emotion recognition (although these methods only provide evidence for presence of errors). Incorporation of this information to the prediction model would allow to use the error correction probabilities derived in section 6.2.2.

In practice, even if we devised a good model, it doesn't mean users would adopt it (not overriding the system's proposed modality choice), during application of the algorithm Also the generalization power of the model should be validated across different conditions, e.g. levels of relative modality efficiency. For example it would be interesting to investigate it's performance in situation where user's modality choice decision is always hard, e.g. all attributes in the travel reservation application have almost equal size resulting in same modality efficiency for speech and GUI modalities.

Chapter 7

Affective Evaluation

7.1 Introduction

In human communication affect and emotion play an important role, as they enrich the communication channel between the interacting parties. The lack of this source of information in human computer interaction has recently inspired many research efforts which aim at incorporating affective and emotional cues in the human computer interaction loop. These efforts are known collectively as affective computing, a term coined by MIT Media Lab professor Rosalind Picard [122].

In contrast to the previous chapters where evaluation of the interaction systems were based on evaluation metrics such as interaction speed, error rates, modality selection and synergy, this chapter uses affective metrics such as excitement, frustration and engagement for the evaluation of the various systems. This, not only provides a more qualitative approach to evaluation, it also provides a better understanding of the interaction process in general. The methodology proposed is based on the use of two different modalities for the measurement of affect. The first is Galvanic Skin Response (GSR) which relates to the sympathetic nervous system and reveals emotional arousal. The second is Electroencephalography (EEG) a rich source of information which is able to reveal hints of both affective and cognitive state during an interaction task. Use of such physiological channels and their elaborated interpretation is a challenging but also a potentially rewarding direction towards emotional and cognitive assessment of multimodal interaction design.

In section 7.2 a brief introduction of affective computing is presented. Section 7.3 describes in detail the EEG and GSR apparatus used along with the software developed (affective recording studio) to record and analyze the evaluation sessions. It also outlines the techniques used for eliminating noise (artifact removal) of the EEG signals. Section 7.4 presents and comments the affective evaluation results. Further discussion regarding various issues such as the appli-

cability of affective evaluation is provided in section 7.5. Additional ideas for future work are discussed in section 7.6. The chapter concludes with section 7.7.

7.2 Affective computing

Affective computing studies the communication of affect between humans and computational systems. It is an interdisciplinary area of research, spanning disciplines such as psychology (study of affect and emotion) cognitive science (emotion and memory, emotion and attention) computer and electrical engineering and design (signal processing, affective detection and interpretation, affective design). In the realm of HCI and context awareness, it aims at supporting enhanced interaction experience by utilizing the affective dimension of communication.

The main efforts until now have been concentrated in the fields of affective detection and emotion recognition[1]. Affective computing relies on detection of emotional cues in channels such as speech (emotional speech), face (facial affect detection) and body gestures. It also utilizes a number of physiological channels such as GSR, facial electromyography (EMG), blood pressure, heart rate monitoring (EKG) and pupil dilation. All these channels have been shown to correlate with certain emotional states such as fear, joy, surprise, etc. Thus they can potentially provide valuable information in the course of interface evaluation.

Choosing which modalities (channels) to use in an affective setting depends on many factors, such as availability of modality (e.g. emotional speech cannot be used in the evaluation of a GUI system), affect resolution (facial expression more appropriate than GSR) and affect recognition rates. Thus modality appropriateness play an important role. As a result, as emotion modulates all these channels, the idea of using more than one channels (e.g. facial expression and EKG) is common in the research community and is referred to as multimodal affect recognition.

Advancements in cognitive and brain sciences has recently made it possible to add the brain as another source of rich information. In the case of the study of the brain, the term is emotion instead of affect recognition to reflect the fact that affect is the expression of emotions. Detection of emotions using brain signals (EEG) compared to affective detection is way more difficult but also conveys much more information. Also, using brain signals, apart from emotions, we can study other cognitive functions that are also of high significance in the context of interaction design such as attention, memory and cognitive load.

Incorporation of brain study is thus a challenging but also a potentially rewarding direction towards emotional and cognitive computing.

[1]Note that affect is generally considered to be the expression of emotions and as such is used in the context of this thesis.

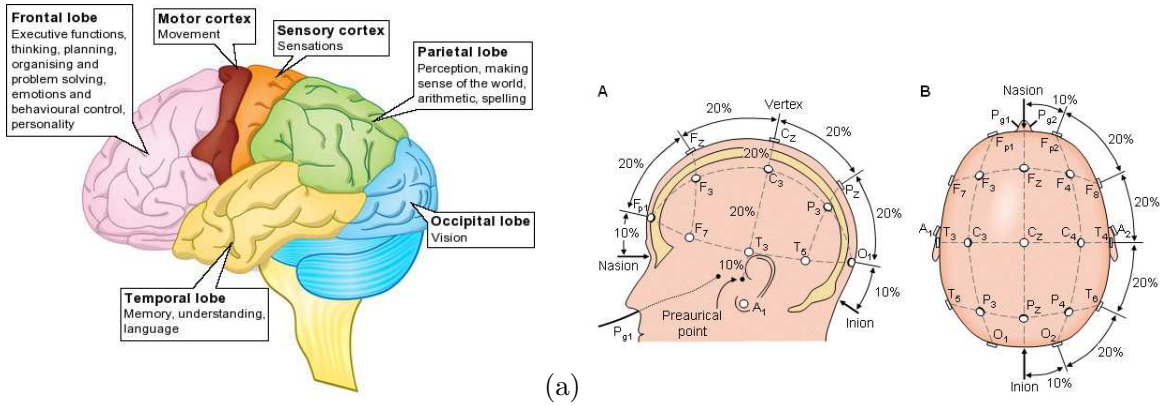


Figure 7.1: (a) Human brain areas (b) EEG sensor locations according to 10-20 system [6].

7.2.1 The human brain

The human brain is the more complex human organ. There exist around 100 billion neurons in the human brain, about the same number as of stars in our galaxy. The study and understanding of the brain has been benefited by the recent advances in sophisticated equipment such as functional magnetic resonance imaging (fMRI), magnetoencephalography (MEG) and near-infrared spectroscopy (NIRS). Compared to EEG which is the older method used in brain studies most of these methods are very expensive or have low data transmission rates and are not ambulatory. EEG is an established and mature technology which can be used outside the lab, has high temporal resolution (which makes it ideal for interaction evaluation) and is relatively cheap. The main drawback of EEG compared to the other methods is the relative poor spatial resolution and the high noise from non cognitive sources called artifacts.

Fig.7.1(a) shows the main areas of the human brain called lobes in different colors along with their main functionality. Some of these areas are almost dedicated to a single function such as sensorimotor cortex (sensation and movements) or occipital lobe (vision). Other areas have been found to be the source of a large variety of cognitive processes such as the frontal lobe which is the center of planning, problem solving and also emotions to name a few.

EEG measures the electric potential of the scalp by detecting the summation of the synchronous activity of thousands or millions of neurons. Using surface electrodes at various scalp locations it can reliably detect even small such changes in the cerebral cortex. Because the brain activity is attenuated by the tissue and bone between the cerebral cortex and the electrodes, the recorded potentials are only in the range of a few microvolts; midline or deep structures of the brain on the other hand have minimal impact in the EEG recordings.

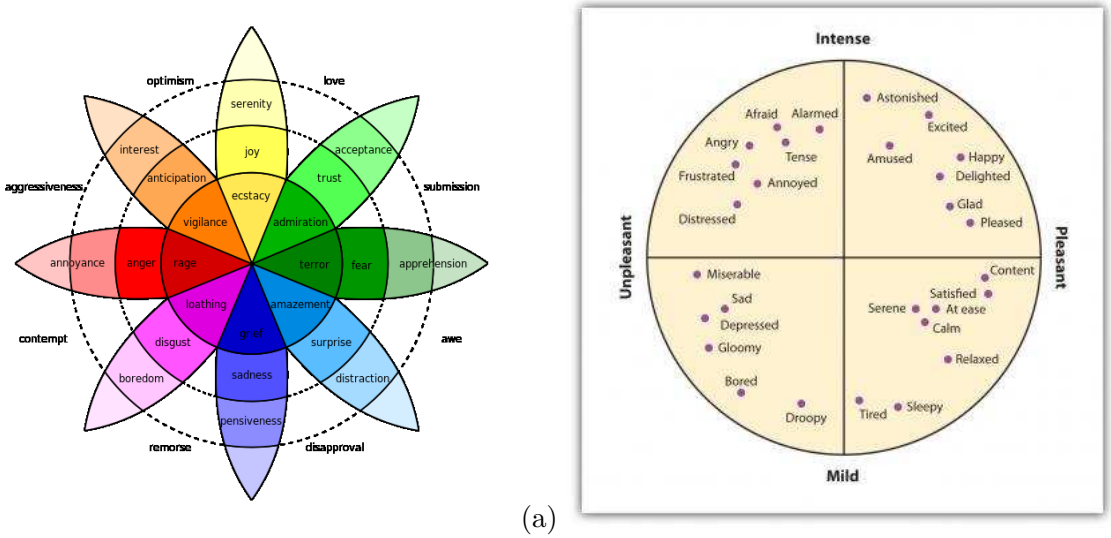
A large number of electrodes are usually used in clinical settings to allow for a relative adequate spatial resolution. The placement of the electrodes follows some standard to allow

for reproducibility across a subject's measurements or between subjects. A common standard used in the 10-20 system shown in Fig.7.1(b). The two numbers refer to the distances between adjacent electrodes which can be at 10% or 20% of the total front-back or right-left distance of the skull. Locations are identified by a letter corresponding to the lobe (e.g. F for frontal, P for parietal, etc) and a number that identifies the hemisphere location (odd or even numbers for left/right hemispheres respectively).

The brain activity produces a rhythmic signal that is constantly present. These rhythms (so called brain waves in popular literature) are divided in several bands according to their frequency which ranges from 1-100 Hz, have characteristic amplitudes ranging from 10 to 100 microvolts and are associated with certain states. These bands are delta (up to 4Hz), theta (4-8 Hz), alpha (8-13 Hz), beta (13-30 Hz), gamma (>30Hz) and mu rhythm(8-13 Hz). Lower bands have higher amplitude compared to high frequency ones. Alpha and beta waves are the most relevant to this work. Alpha waves are typical of an alert but relaxed mental state and are evident in the parietal and occipital lobes. Beta waves are indicative of active thinking and concentration, found mainly in frontal and other areas of the brain.

Apart from the regular brain rhythms, electric activity is altered during external events (e.g. sensory stimuli) or internal processes taking place. These changes occurring during such external or internal events are generally called event potentials (EPs). Study of these EPs can be accomplished using various approaches such as ERPs and ERD/ERS. Event-related potentials (ERPs) are transient changes in brain activity (typically a series of voltage polarity changes with characteristic peaks and troughs) time locked to the onset of an event. As different kind of EPRs have been found to be related with distinct events or processes in the brain, they are used to distinguish and identify the different neural processes involved in perceptual tasks. For example P300 is an ERP elicited using the *oddball paradigm* in which infrequent target items are mixed with frequent non-target ones. It is used in the P300 brain computer interface keyboard (P300 speller) to allow people with disabilities to enter text using only brain input. Note that because of the relative small fluctuations of ERPs compared to background brain activity the study of a certain phenomenon requires averaging of a large number of time locked trials called epochs; recent developments towards single trial identification of EPRs is an active research effort in this field.

Another modulation of information flow in the brain is manifested by event related synchronization and desynchronization (ERS/ERD) of EEG rhythms [123]. These phenomena correspond to relative decrease/increase in the power of a certain frequency band during stimulus processing. For example according to [124] the encoding of acoustic information elicits topographically widespread alpha-ERS responses whereas auditory retrieval or recognition elicits topographically widespread alpha-ERD responses. Similarly it is well known that execution of movement e.g. finger movement, is accompanied by a desynchronization of mu and central



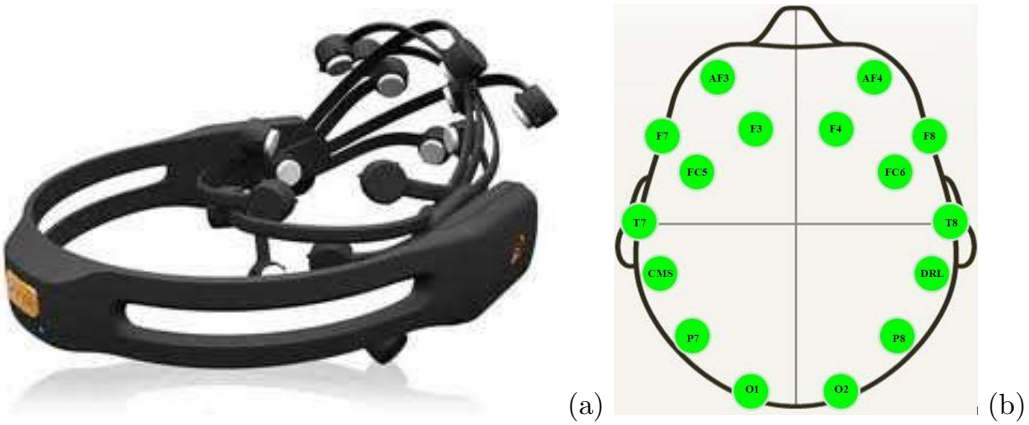
(a) (b)

Figure 7.2: (a) Plutchiks model of emotions. (b) Emotions mapped to arousal - valence space (y - x axis respectively).

beta rhythms over the corresponding area of the sensorimotor cortex. This again is used for the detection of imaginary movements in BCI scenarios where people with disabilities can move prosthetic limbs using just brain input.

EEG studies more relevant to the domain of HCI and affective computing are those studying emotions and fundamental cognitive processes related to attention and memory. EEG emotion recognition has been an active topic in the last years. Emotion recognition using EEG has advantage over affect recognition using e.g. video facial expression recognition since it potentially allows for a more fine grained classification and might be present even if not expressed. There are various representations of emotions such as the wheel of emotions by Plutchik [125] shown in Fig.7.2(a). Fig.7.2(b) shows the mapping of various emotions in the arousal-valence space, one of the most used frameworks in the study of emotions. Arousal is the degree of awakeness and reactivity to stimuli and valence is the positiveness degree of a feeling. The mapping of emotions in the arousal-valence space allows a quantitative approach to emotion recognition since theoretically, if one could estimate these two values he could easily determine the exact emotion. According to previous studies, indicative metrics of arousal is the beta/alpha band power ratio in the frontal lobe area of Fp1, Fp2 and FPz. For valence the alpha ratio of frontal electrodes F3/F4 has been used, as according to [126] there is hemisphere asymmetry in emotions regarding valence e.g. positive emotions are experienced in the left frontal area while negative emotions on the right frontal area.

User state estimation based on cognitive attributes related to attention and memory (in addition to emotions) is also of great importance in the context of HCI research. Memory load



(a) (b)

Figure 7.3: a) The Emotiv Epoc neuroheadset, a 14 channel consumer EEG device. b) locations of the 14 EEG channels according to 10-20 system [6]; CMS/DLR are the two reference electrodes.

(an index of cognitive load[2]) is an important index of mental effort while carrying out a task. Memory load classification has thus drawn attention from the HCI research community since it can reveal qualitative parameters of an interface. In [127] authors report a classification accuracy of 99% for two and 88% for four different levels of memory load. They argue that previous research findings that high memory loads correlate with increase in theta and low-beta(12-15 Hz) bands power in the frontal lobe may not always hold true for their experiments. Other memory load metrics such as the ratio of beta/(alpha+theta) powers are also put in question. They built their classifier by exploiting data from the *n-back* experiment [128]. N-back is a well known experiment in which at each trial, participants are presented with a specific stimuli (e.g. letters) and have to recall the last trial they encountered the same stimuli; thus they have to hold a sequence of n items in memory. Interestingly, they also report that using only three specific electrodes (Cz, Pz, Fz) they get quite the same performance as with using all 32 electrodes.

7.2.2 Galvanic Skin Response

Galvanic skin response (GSR) or skin conductance measures the electrical conductance of the skin, which varies with its moisture level. GSR is used as an indication of psychological or physiological arousal since sweat glands are controlled by the sympathetic nervous system. Previous studies indicate that GSR may correlate not only to emotional (e.g. arousal [129]) but also to cognitive (e.g. cognitive load [130]) activity.

[2]Cognitive load can be defined as a multidimensional construct representing the load that performing a particular task imposes on the learners cognitive system [100]

7.3 Affective evaluation

7.3.1 EEG device

The EEG device used is the Emotiv[3] Epoc, a 14 electrode neuroheadset device (see Fig 7.3) targeting mainly the gaming and HCI market. The main advantage of the device is the very low price (around 300$ for the consumer and 700$ for research edition, as of 2010) compared to clinical grade EEG devices prices (tenths of thousands of $). In addition to it's price, it is very easy to use and the preparation time is very short (only few minutes to apply saline solution) compared again to clinical EEG systems which require enough time and expertise in order to use. It is also wireless allowing the users to freely move while interacting. Finally, the provided SDK provides a suite of detections (affective, expressive and cognitive) which allow people without EEG expertise to integrate them to a large number of applications. The main critique is the low number of electrodes (compared to e.g. 128 electrodes of clinical grade EEG) and the lower sensitivity/higher noise of EEG measurements compared to clinical grade EEG devices. Nevertheless, using advanced noise filtering techniques one can solve most of these issues. As a result, the device has been actively used recently by game developers, individual researchers and HCI labs around the world.

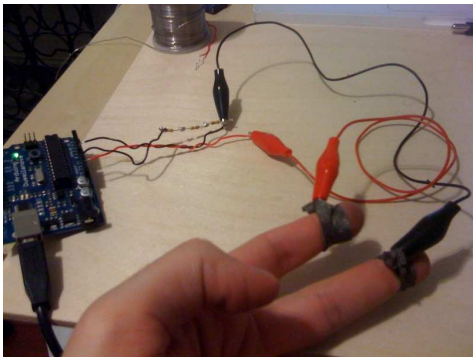
As the device is mainly targeted towards computer interaction, it is accompanied by a standalone tool which offers the following family of capabilities (suites):

- Expressive suite, which detects user's face expression and depicts them using a talking head agent (avatar)
- Affective suite, which measures several affective metrics such as frustration, engagement, excitement and meditation
- Cognitive suite, which allows the mapping of different cognitive patterns to different actions, e.g. pull/push a virtual object in the screen

These capabilities allow a user to associate various expressive events or cognitive states to computer actions and thus use the Epoc device as a BCI (brain computer interface) [131] device, substituting devices like mice and joysticks. This capability is also offered for programmers through a programming library, which allows to integrate the device into any kind of application. Finally a research edition offers additional features such as access to raw EEG signals.

The detailed device specifications, according to the company, are listed below.

[3] www.emotiv.com


(a)

(b)

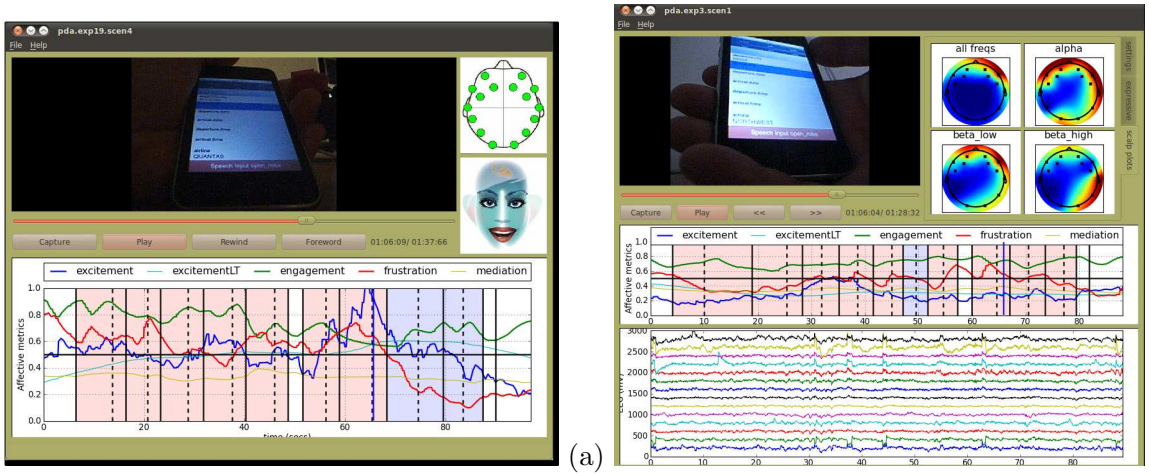
Figure 7.4: (a) Early Galvanic Skin Response (GSR) apparatus. Breadboard circuit and velcro strips were added later. (b) Evaluation setting. Depicted counter clockwise is the iphone device, the GSR apparatus (arduino and breadboard), the Emotiv device, the audio headset and the PlayStation Eye camera.

Number of channels	14 (plus CMS/DRL references)
Channel locations (10-20 system)	AF3 AF4 F3 F4 F7 F8 FC5 FC6 P7 P8 T7 T8 O1 O2
Sampling method	Sequential sampling, single ADC
Sampling rate	128Hz (2048Hz internal)
Resolution	16 bits (14 bits effective) 1 LSB = 1.95 μV
Bandwidth	0.2 - 45Hz, digital notch filters at 50Hz and 60Hz
Dynamic range (input referred)	256 mVpp
Coupling mode	AC coupled
Connectivity	Proprietary wireless, 2.4GHz band
Battery type	Li-poly, 12 hrs
Impedence measurement	Contact quality using patented system

7.3.2 GSR apparatus

The GSR apparatus designed exploits the arduino physical computing platform [4]. As shown in Fig. 7.4 the hardware part basically consists of an arduino board connected to a simple circuit. On the software side a program was developed and uploaded to the arduino board which takes care of reading the GSR values and make them accessible to the computer in digital format in a rate of 50 readings per second. GSR was measured using electrodes placed inside Velcro straps on the distal phalanx of both the forefinger and middle finger of the left hand.

[4] `http://arduino.cc/en/`



(a) (b)

Figure 7.5: Screenshots of the affective evaluation studio replaying previously recorded sessions. (a) Standard edition. The two main components depicted are the video and affective plot (see Fig 7.6) widgets. The vertical blue line indicates the playing position in the affective data corresponding to video frame displayed. The user can click on any position of the plot to move in that particular moment in the video stream or vice versa using the video slider. The two widgets in the right of the video widget display the 14 electrode contact quality and the user face expression widget. (b) Research edition. Offers additional EEG processing capabilities such as EEG plot (found below affective plot) and single channel analysis plot and spectrogram (shown when selecting specific channel). It also provides real time scalp plots (next to video widget) which show EEG power distribution for selected spectrum bands animated through time.

7.3.3 Affective Evaluation Studio

A dedicated tool was developed to collect in real time, data from the Emotiv device (affective and EEG), the GSR system and a video camera. Screenshots of the affective studio are shown in Fig 7.5 for the standard and research versions of the Emotiv SDK respectively[5]. The tool was used to capture, record, replay and analyze evaluation sessions of the multimodal interaction system[6]. A short list of its capabilities include:

- In capture mode, it captures data from Emotiv device, GSR system and video camera. This is useful for the examiner to check and resolve any problems such as the correct contcact quality of Emotiv or GSR before starting the recording of a new session.

- In recording mode, affective, EEG, GSR and video data are all concurrently saved while also been displayed for the duration of the interaction session. Again the real time

[5] A video demonstration of the tool functionality is available on line at `http://www.youtube.com/user/holystone74`

[6] Actually it was developed as a general purpose tool that can be used for the evaluation of any interaction system and beyond e.g. music listening, video trailer evaluation, etc.

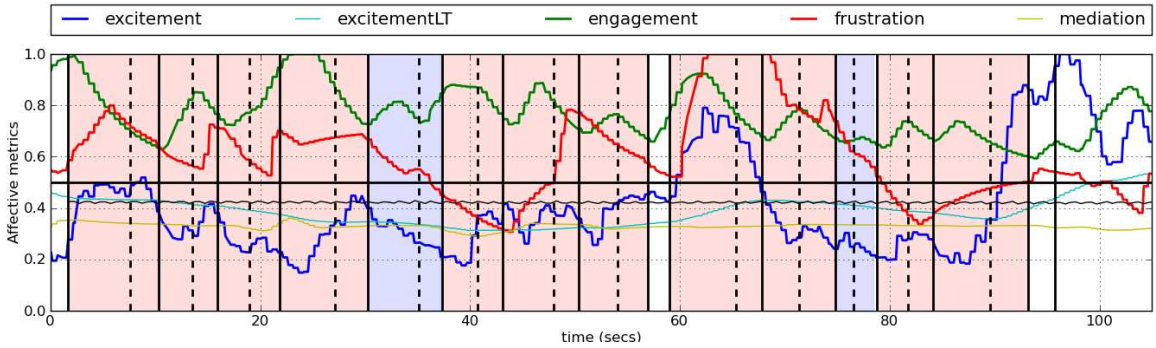


Figure 7.6: Example session (OM scenario) annotated in the affective plot. Annotation projects the multimodal's system's log file information (turn duration, input type, etc) onto the affected data of a recorded session. The five affective metrics (excitement, long term excitement, engagement, frustration and meditation) provided by EPOC are depicted, along with the GSR values (black horizontal line oscillating around 0.4) in the [0-1] space. The software automatically annotates the plot showing all interaction turns. A turn is the time period between two thick vertical lines; each dotted vertical line separates a turn into the inactivity and interaction periods. Only fill turns have background color. That color is red for speech turns and blue for GUI turns. The whole interaction period is defined between first and last vertical line.

information is used by the examiner to ensure for the correctness of the each recording session.

- In play mode, data are displayed, annotated and analyzed (e.g. affective annotations and EEG spectrograms and scalpmaps - see Fig 7.5(b)) offering valuable insights for the course of an interaction session.

The tool serves as a valuable tool for inspecting in detail how users interact with the system.

7.3.4 Participants and Procedure

For this evaluation study, eight healthy right handed graduate university students participated. They were all briefly introduced to the nature of the experiment. After wearing the Emotiv headset and the GSR apparatus, they were asked to take a comfortable position and instructed to avoid excess movements. All five multimodal interaction modes were used during the evaluation (SO, GO and three MM ones). Participants tried all different systems at least once in order to get familiar with the systems before starting the evaluation scenarios. For the evaluation scenarios, four different two way trip scenarios were used, that is a total of 20 (5 systems $\times$ 4 scenarios) sessions per user. Some of the participants opted for evaluating all 20 sessions while some for only 15. Note that in contrast with previous evaluations, users were

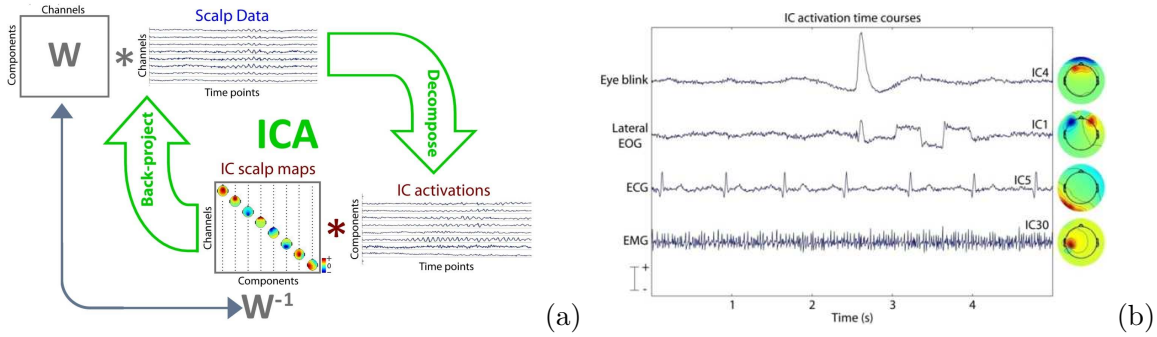


Figure 7.7: (a) Schematic flowchart for Independent Component Analysis (ICA) data decomposition and back-projection [7]. (b) ICA components accounting for eye blinks, lateral eye movements (EOG), ECG and EMG [7].

advised to speak one concept per turn (in case of speech usage) in order to collect as many turns as possible.

7.3.5 Artifact removal using ICA

One important problem in EEG analysis is that the recorded EEG signal might be heavily affected by various always present and unavoidable artifacts such as these caused by eye movements, eye blinks and muscle activity. Such signals are *mixed* with the real brain activity causing a noisy EEG signal at most locations. The Emotiv SDK preprocesses and appropriately filters data depending on the type of detections. Although the exact procedure is disclosed it apparently uses minimal filtering for the expressive detections (allowing to detect face expressions). Affective and cognitive detections on the other hand are designed to filter out most noise artifacts and different preprocessing techniques are applied for each one. Nevertheless, because Emotiv is targeted at real time interaction it makes it very difficult to use more advanced artifact removal techniques that can be effectively used only in batch mode (after all EEG samples have been acquired).

From initial analysis contacted on the collected data it was found that excitement and engagement metrics may be affected by high noise artifacts such as that caused by tense jaw movements during speech interaction. GUI interaction sessions usually lack such noisy affective metrics patterns. Thus, in order to be able to compare GUI and speech parts of interaction, it was decided to spend some effort to clean the EEG signal of such artifacts.

Various techniques have been developed in identifying and correcting such artifacts. A common approach used in the past was for experts to identify such artifacts in the time domain and then just reject portions of the EEG signal affected. This methodology may still apply in cases when EEG recordings and nature of experiments are based on large number

of repetitive trials (commonly with fixed time period called epochs). However deleting whole epochs contaminated by artifacts may not be applicable to other types of experiments such as for the evaluation sessions used in this thesis.

The solution chosen for EEG artifact removal is based on ICA (Independent Comparing Analysis) [132]. Fig 7.7(a) shows the outline of the procedure. Scalp data are unmixed to independent components called activations. These activations can be examined either in time domain or as scalp maps (Fig 7.7(b)), a representation that also reveals topological distribution of spectral properties of EEG sensor locations. The transformation matrix W mapping from scalp to components data can then be used in reverse to obtain the original scalp data. What this procedure offers is that independent components representation unmixes the signals allowing for easier artifact detection but also rejection procedure, since components identified as artifacts can be removed (just rejected) and by inverse transforming to scalp domain, artifact clean data can be obtained.

Thus, the main task of the examiner is to identify certain components that account for known types of artifacts as shown in Fig 7.7(b). Eye based artifacts are always present, since both eye movements and blinks are unavoidable. As shown in the Fig.7.7(b) (IC4 component), eye blinks cause abrupt voltage change in the prefrontal area locations. Lateral eye movements are also possible to identify, since they have characteristic properties (component IC1). The source of noise can be attributed to the potential difference between the cornea and the retina, which is called corneo-retinal dipole. Eye movements yield an increased potential in electrodes towards which the eyes are rotated and decreased potentials in the opposing electrodes. Heart activity is also a common source of interference and is more evident in the left back area the scalp area (component IC5).

Muscle artifacts are also very common and difficult to identify because of the variety of possible sources and spectrum properties; this is the reason why users during experiments are advised to reduce excess movements to a minimum. Sources of muscle artifacts may be caused by various parts of the body such as limb or even finger movements, head movements, face expressions, jaw and tongue movements (glossokinetic artifacts). For example, finger and hand movements are evident in sensorimotor cortex (refer to Fig 7.1(a)) and are exploited in BCI (brain computer interface) systems to detect real or even imaginary movements. Other sources of artifacts include EEG sensor displacements known as electrode pops which cause abrupt impedance changes and thus cause sharp voltage changes in the EEG signal. Mains interference at 50 or 60 Hz may also heavily contaminate the EEG signals, so notch filters are usually applied at these frequencies.

Once these artifacts are identified they can be rejected to yield artifact clean scalp data. For this purpose the EEGLAB [7] software toolbox was used. For each evaluation session recorded the raw EEG data were analyzed using the ICA technique. By examining the ICA

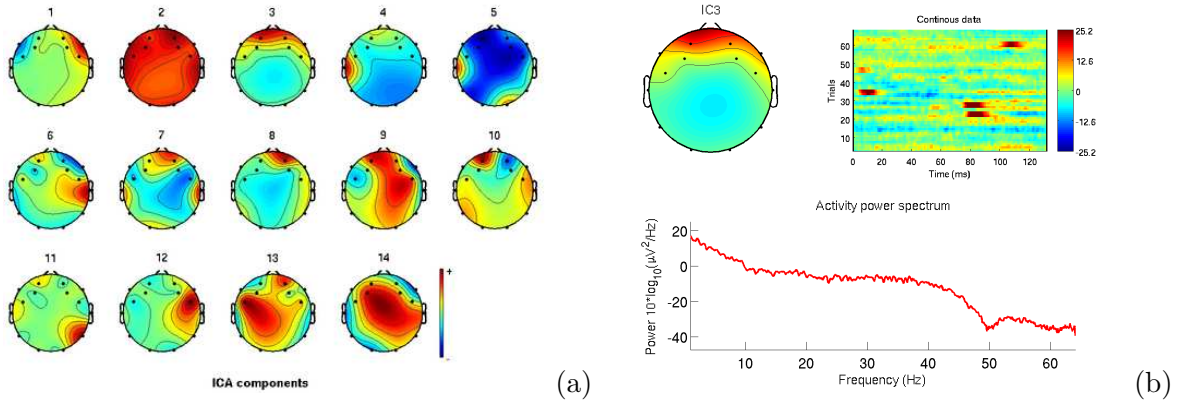


Figure 7.8: (a) The 14 ICA components of a sample session shown as scalp maps. (b) IC3 is an eye blink artifact component and will be rejected.

activations, artifacts were identified and rejected from the data resulting EEG data with much less noise. These artifact clean data were used to produce new affective metrics [7].

Results of an artifact rejection procedure for a GUI-only evaluation session are shown in Fig.7.8 and Fig.7.9. Fig.7.8(a) shows the 14 ICA activations produced, represented as scalp maps. This representation reveals the relative EEG signal power spectrum of various EEG sensors which can help the examiner to find out known patterns of artifacts or other activity by visual inspection. Because the order of components corresponds to the amount each components contributes to variance of the EEG signal, candidates components for artifact rejection are usually at the leading positions of the scalp array [8]. For example, component IC3 is similar to a typical eye blink artifact, since component power is located in the two electrodes above the eyes, as shown in Fig.7.8(b). In that figure in addition to the scalp plot, there is a spectrogram and a power spectrum plot. Although the data are continuous they are artificially epoched for a time period for 130 msecs and analyzed. The plot clearly reveals four eye blinks (the four red stripes) in the course of the evaluation, thus the component can safely be removed from the data.

Shown in Fig.7.9(a) is the time course of the 14 ICA components for the same evaluation session. The three first components apparently account for artifacts since they contain high levels of noise compared to the rest 11 components. The IC1 accounts for a typical eye-movement artifact containing 6 events; IC3 accounts for eye blink component with 4 events. Fig.7.9(b) shows the original noisy scalp data. Notice how the six eye movements and blinks confound the scalp data. These artifacts are evident in the first and last signals of Fig.7.9(b)

[7] Although this functionality was not provided by the Emotiv SDK.

[8] According to EEGLAB manual, the scale in the scalp plot uses arbitrary units. The scale of the component's activity time course (shown in Fig.7.9[a]) also uses arbitrary units. However, the component's scalpmap values multiplied by the component activity time course is in the same unit as the data, that is μV.

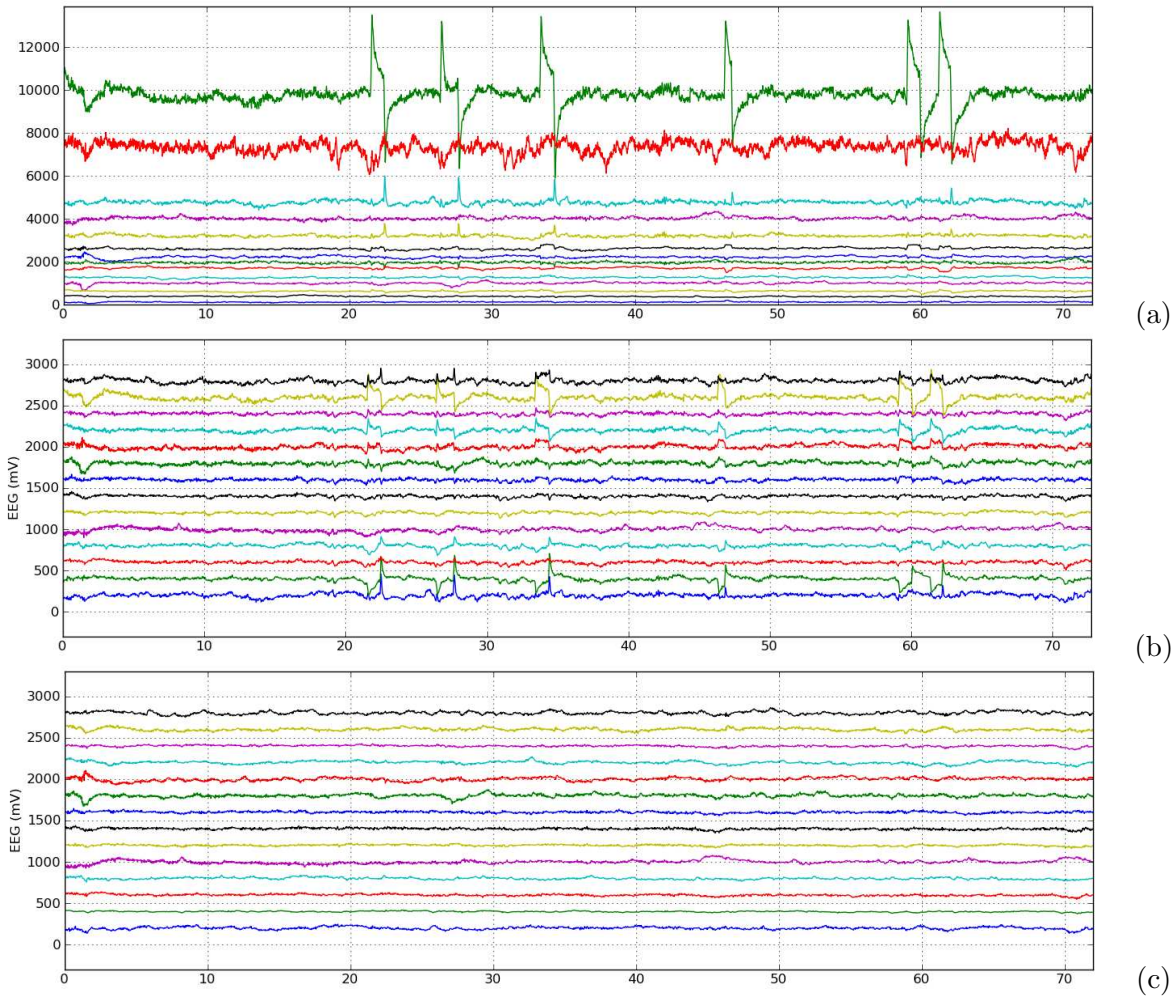


Figure 7.9: (a) The 14 ICA (IC1 - IC14) components of the same session plotted against time. (b) Original scalp data containing artifacts. (c) Scalp data after rejection of three first components (artifacts). Note that all signals are displaced in the $y - axis$ to allow better viewing.

which correspond to frontal and prefrontal locations (AF3, F7, F3, FC5 on the left side FC6, F4, F8, AF4 on the right side); eye movements show opposite polarity for left and right sides for the eye movement artifact. The clean data (after removing the first three ICA components) are shown in Fig.7.9(c). As shown clearly, removing artifactual ICA components results more reliable data to work with.

7.4 Results

To illustrate the results, some example evaluation sessions are reported first as shown in Fig. 7.10. The figure shows the first five evaluation sessions of a single participant (usr4) including all five different systems (evaluated in the order plotted). Explanation of an annotated affective plot was detailed earlier in Fig. 7.6. Note that GSR values are reported in microSiemens and are rescaled to 0-1 space in order to be fitted to the same plot along affective metrics[9]. Also note that although EEG data have a constant sampling rate of 128Hz, affective metrics have a rate of around 12Hz; the detections are event-driven and their sample rates depend on the number of expressive and cognitive events. This means that affective metrics have low temporal resolution compared to EEG. The three affective metrics more relevant to this study are frustration, excitement and engagement. Since all scenarios are two way and users spoke one concept per speech turns, a total of eight fill turns are needed; thus evaluations with more than eight fill turns in Fig. 7.10 denote one or more erroneous inputs.

Fig. 7.10(a) shows the GO session. Note that affective metrics have a generally smooth plot, except for turns 7 and 8. As shown in turn 7, frustration raises high after user realizes he entered the wrong value in the previous turn. Note how it immediately decreases after the dashed line (interaction starts) of the same turn. In turn 8, frustration raises after the dashed line. The user seems to be confused about which value to select; when selected, frustration and later excitement start decreasing again. Fig. 7.10(b) shows a CT session. Since no errors occur, the plot has again smooth lines. Note how both frustration and excitement are less than 0.5 for the duration of the session.

Fig. 7.10(c) shows an OM session. Due to a couple of speech recognition errors and user confusion there is an evident variability for frustration at these points. Note however, how using GUI input to fix the error in turn 5 results in a rapid decrement of frustration; this is a pattern found frequently in the whole evaluation set. Similar to previous session, the MS session shown in Fig. 7.10(d) shows raising of frustration when speech errors occur or user is confused about system's response. Again frustration rapidly decreases when user corrects errors as shown in the two GUI input turns. Fig. 7.10(e) is the SO session. Due to many speech errors or user confusion, these variations in both frustration and excitement happen a lot of times; again, this is found frequently across the whole evaluation set.

Examining the GSR values across the five sessions, one can find differences during each session but also between them. GSR is 0.53 microSiemens for GO session and slightly increases during the end of the CT session. It is mostly constant for the OM session and increases to 0.59 by the end of the MS session for which more errors happened. The increase of GSR values continues even after the end of MS and the start of the SO session. As a result the SO session

[9] Actual values are a multiplication of 10 compared to ones plotted.

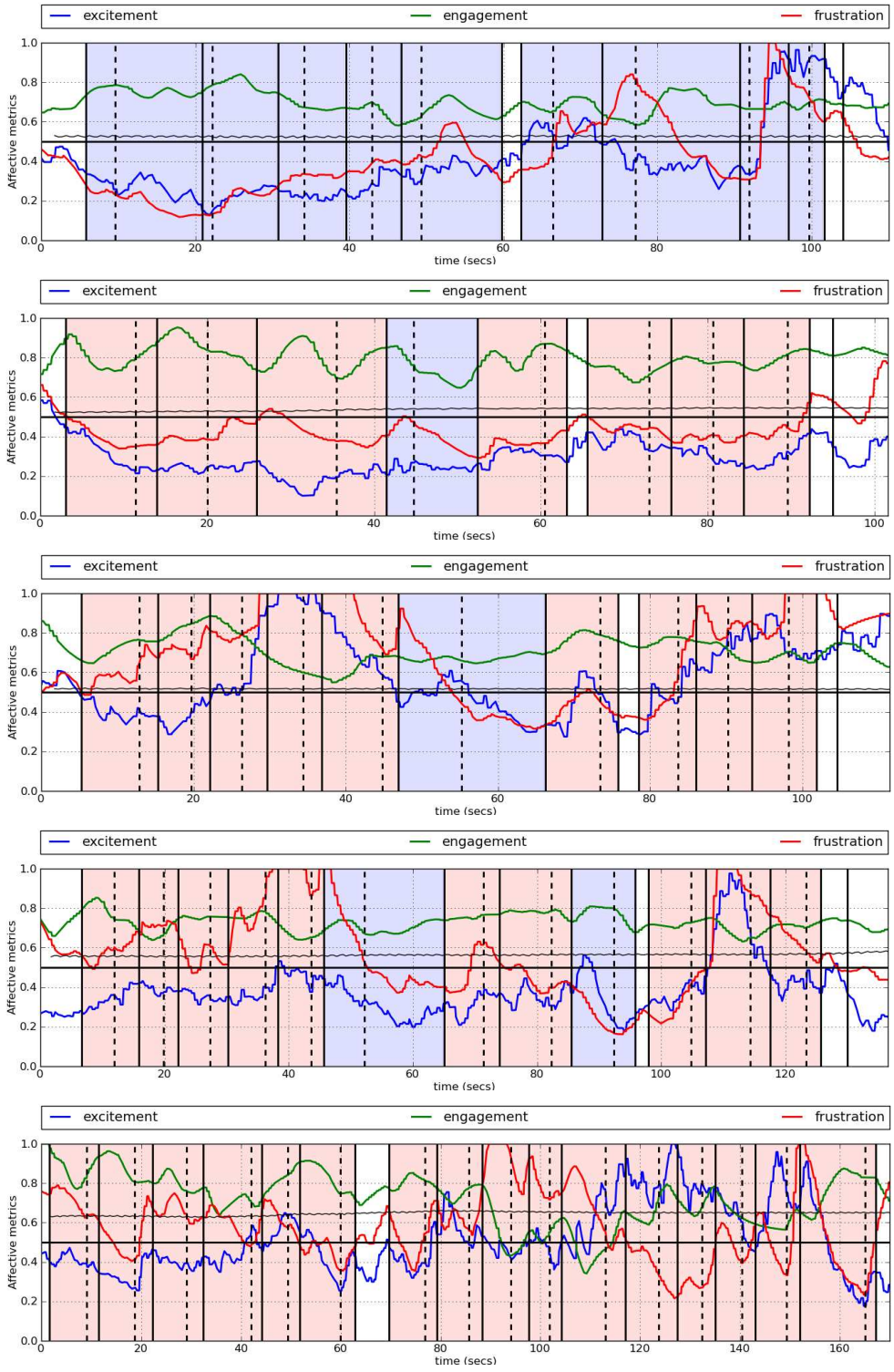


Figure 7.10: Sample evaluation sessions for usr4 (a)GO (b)CTT (c)OM (d)MS (e)SO

starts with a GSR value of 0.62 and reaches the levels of 0.65 at the end of the session. A possible explanation for these changes is that as the number of errors increases the difficulty and the effort needed by the user to finish the sessions yield an arousal and cognitive load increase which in turn seem to increase resulting GSR values as well.

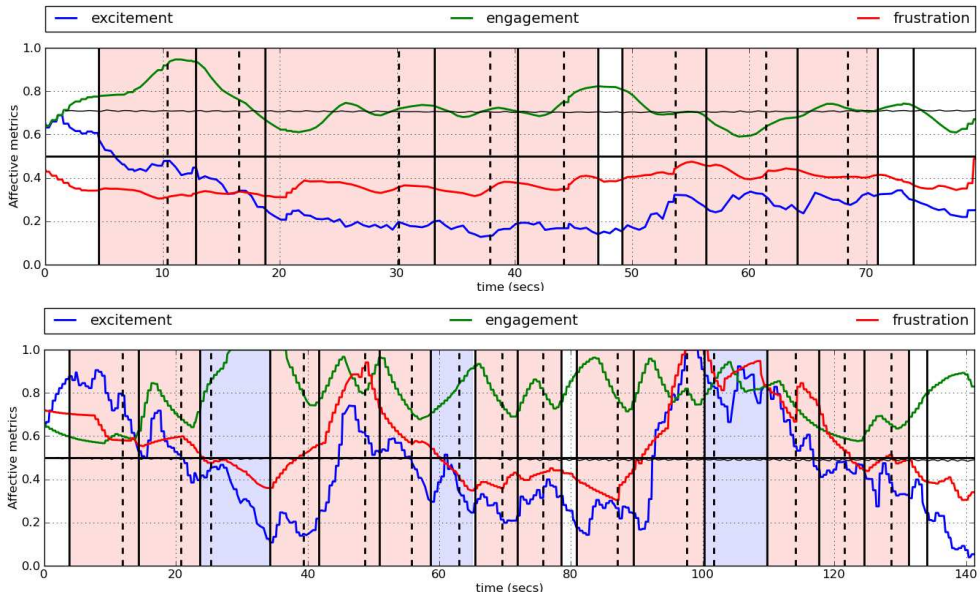


Figure 7.11: Additional example evaluation sessions (a) user 7 MT session (b) user 8 CT session

Figure 7.11 shows two more examples sessions for two more users. Figure 7.11(a) shows a representative session of usr7. This user has the lowest overall speech WER and was very confident in using speech. Notice how smooth is the plot for all the affective metrics and that levels of both excitement and frustration are low. Figure 7.11(b) shows an example session for usr8. What is interesting about this user is the way engagement changes through the turns. As can be seen in this example (but also in user's rest sessions), engagement rises at the start of each turn, reaching a highest value (usually near the interaction time - dashed line) and then decreases to reach the lowest point near the end of the turn.

It would also be useful to examine how affective metrics relate to input type and the different interaction systems. Table 7.1 shows the mean and standard deviation for the three affective metrics according to input type (GUI or speech); the last row shows the overall (GUI and speech) results. Notice that *for both excitement and frustration speech input has higher levels compared to GUI input* by 5% and 6% respectively. This happens because as shown previously in the affective plots, speech recognition errors cause both frustration and excitement changes. For engagement on the other hand, GUI input has slightly higher levels. Overall results, show that engagement levels are higher and have less variance compared with

Table 7.1: Affective metrics and turn input type

Input type	**engagement**		**excitement**		**frustration**	
	mean	std	mean	std	mean	std
GUI	**0.79**	0.11	**0.45**	0.19	**0.51**	0.17
Speech	**0.76**	0.11	**0.50**	0.19	**0.57**	0.19
overall	0.76	0.11	0.48	0.19	0.56	0.19

Table 7.2: Affective metrics and interaction system (plus input type)

Input type	**engagement**		**excitement**		**frustration**	
	mean	std	mean	std	mean	std
GO	**0.78**	0.11	**0.44**	0.19	**0.50**	0.15
CT (GUI input)	0.79	0.11	0.43	0.17	0.50	0.18
CT (speech input)	0.78	0.10	0.47	0.17	0.57	0.19
CT (overall)	0.78	0.10	0.46	0.17	0.56	0.19
OM (GUI input)	0.80	0.11	0.44	0.19	0.52	0.21
OM (speech input)	0.76	0.11	0.47	0.17	0.58	0.19
OM (overall)	0.77	0.11	0.46	0.18	0.57	0.19
MS (GUI input)	0.80	0.12	0.46	0.20	0.54	0.21
MS (speech input)	0.76	0.10	0.47	0.17	0.58	0.19
MS (overall)	0.77	0.11	0.47	0.18	0.57	0.19
SO	**0.73**	0.12	**0.54**	0.21	**0.59**	0.20

frustration and excitement.

Table 7.2 shows the mean and standard deviation for the three affective metrics for each of the five interaction modes. For the three multimodal modes, results are presented per input (GUI/speech) and overall (independent of input type, that is both GUI and speech input). Engagement for SO system is lower compared to all other systems; note that for all three MM modes GUI input has slightly higher engagement compared to speech input as shown in the previous table. Excitement is much higher for SO compared to GO system (0.54 & 0.44 respectively). Multimodal modes as a mixture of SO and GO systems have average excitement values lying between these two values and closer to that of GO. Similarly, for frustration, SO values are much higher compared to GO system (0.59 & 0.50 respectively) while multimodal modes have average values of around 0.57.

Table 7.3 shows the mean and standard deviation for the three affective metrics for all eight users. Notice the differences between users. For example usr7 has by far the lowest excitement and frustration levels (ASR WER 6%) while usr8 with the higher WER, has the highest levels for excitement and second highest for frustration.

Table 7.3: Users affective metrics

User	**engagement**		**excitement**		**frustration**	
	mean	std	mean	std	mean	std
usr1	0.80	0.09	0.51	0.20	0.61	0.18
usr2	0.74	0.09	0.47	0.15	0.54	0.19
usr3	0.82	0.14	0.51	0.24	0.55	0.20
usr4	0.73	0.10	0.45	0.17	0.54	0.19
usr5	0.79	0.09	0.46	0.19	0.54	0.19
usr6	0.79	0.07	0.46	0.14	0.56	0.17
usr7	0.75	0.10	**0.38**	0.16	**0.44**	0.13
usr8	0.71	0.12	**0.54**	0.21	**0.58**	0.18

7.5 Discussion

The recent release of the Emotiv device allowed developers and researchers outside of the neuroscience community to exploit EEG technology. In the context of HCI research, several demonstrations and research efforts have emerged mostly towards using the device as a BCI modality. For example, interfaces that exploit either expressive or cognitive events (such as P300 ERP) as a communication channel to control a robot, or to dial a phone contact [133].

Although verification and validation of such efforts is relatively straightforward, validation of Emotiv's affective metrics is way more difficult because quantification of emotional states is an open research question. The development of these metrics according to the company[10] exploited both EEG and a large number of other biosignals; subjective evaluations, labeling by experts and cross validation procedures were employed for the development of the metrics. According to the company the affective detections depend on the distribution and relative intensity of specific frequency bands, as well as some custom features based on fractal signal analysis. These are passed to a classifier system to detect specific deflections, are low-pass filtered and the outputs are self-scaled to adjust to each user's range of emotion.

Incorporation of biosignals such as GSR or EEG in the user experience design provides a rich amount of data not previously available. Yet the correct association of these data to underlying emotional or cognitive state is a challenging endeavour for the research community. Even for the simpler of the above modalities, the GSR, interpretation of measurements is a difficult task. For example although it is known from various independent studies that there is correlation between both arousal and cognitive load with skin conductance levels, accessing exactly how these parameters affect the resulting measurements is not well understood. Also, validating the results provided by other EEG studies in other domains and different conditions may not even be possible as the complexity and the number of factors affecting the measurements may

[10] http://emotiv.com/forum/messages/forum4/topic1262/message7401

be overly large. This of course is mostly manifested in the case of EEG, since the human brain is the most complex organ known.

Clinical neuroscience EEG studies usually take place in a completely controlled and strict environment in order to isolate certain cognitive phenomena they seek to study and minimize the effect of external factors. To achieve this they also minimize any sources of excessive EEG noise such as these resulting from muscle artifacts. Thus the validity of these studies can be verified relative easily. An important research question is whether the transferability of such results in complex uncontrolled real life scenarios, such as the evaluation of complex interfaces used in this thesis can be verified.

These two issues, namely validity of affective metrics and their use in complex settings make evaluation a challenging task. For example, it was found from this preliminary study that both excitement and frustration may increase in the case of speech errors or user confusion (recall affective plot examples). However there are also times when such fluctuations in affective metrics (especially in frustration and excitement) might not relate to e.g., a speech errors or to exhibit different patterns (e.g. cases when a lot of speech errors takes place in a row). This is to be expected however since a lot of cognitive processes takes place at the same time in the brain. Nevertheless, as shown clearly in the previous tables, there are differences in frustration and excitement between both input type (GUI/speech) and interaction modes (especially for SO and GO).

7.6 Future work

Although the affective metrics collected for the evaluation of multimodal system have shown to reveal valuable insights there are a number of future directions utilizing the EEG signal and the work already presented here to clean and process Emotiv's data that could potentially provide even more insight.

One interesting idea would be the investigation of error related negativity [134] potentials (ERNs). These are ERPs that are elicited when a participant realizes he has performed some kind of error. Their elicitation although complex could potentially be exploited for the detection of user's error response. It could then be used by an adaptive system to offer help or guidance on solving the error, provide alternative modality and so on.

Attention in a multimedia system could also provide additional insights. Although visual attention alone could be investigated using an eye-tracker, it would be interesting to research how user attention is divided between audio and visual channels in the multimodal system. Crossmodal attention and multisensory integration are in the forefront of neuroscience research and are topics of study that could benefit the multimodal research community too.

Another important attribute that could be investigated is cognitive load which relates to

the perceived mental effort. For example, a complicated interface would pose high demands of mental effort for the user to handle and should have higher cognitive load levels. In the case of the systems designed in this thesis, it would be useful to identify if there are any cognitive load differences between the different types of systems (GO/SO/MM) or between modalities for both input (speech/touch) and output (speech and vision). For this purpose a cognitive metric could be designed by the development of a classifier exploiting data from the n-back experiment as noted earlier.

7.7 Conclusions

Although until recently typical evaluation procedures such as objective (e.g. task completion times, error rates) and subjective (questionnaires) provided a basic tool for interaction designers, these techniques lack the ability to give more in-depth insight of the quality of interaction. Incorporation of user affective state and other cognitive features such as attention or cognitive load can prove valuable tools in both the evaluation and design of interaction systems. For example a video game with higher levels of engagement or excitement would enhance the interaction experience and thus be preferred by the users. Likewise a web application that is simple and easy to use (low levels of cognitive load) would be more popular than a complex and difficult to use one.

Use of physiological channels such as GSR, EEG, eye-traking (a method to detect user's visual attention) and their elaborated interpretation can potentially prove invaluable for the design process.

Chapter 8

Conclusions

This chapter presents a summary of the work performed and discusses its general implications for multimodal interaction design. Section 8.1 presents a short summary of the previous chapters highlighting the most important points. A list of the work items accomplished through this research is outlined in section 8.2. The main results of this work are presented and discussed in section 8.3. The chapter concludes with some possible future work at section 8.4.

8.1 Summary

Chapter 3 showcases how to design and build information-filling multimodal systems combining speech and GUI (e.g. pen or touch) input. From the interaction design standpoint, the main focus is on identifying and exploiting the synergies between the modalities and on the investigation of a variety of multimodal interaction modes in addition to "Click-to-Talk", namely "Open-Mike" and "Modality-Selection". The system architecture of a system that allows both unimodal and multimodal interaction and can be used across different platforms such as PCs, PDAs and mobiles is also examined.

In Chapter 4 the methodology used for evaluating the system is presented with a focus on the evaluation metrics used. Some of these metrics are standard objective metrics used in dialogue systems such as task completion ratio, number of turns and turn duration times. These metrics can be additionally measured per user, scenario or context. One important improvement is the break down of turn duration times into inactivity and interaction times which allows to separate system output processing by user from user input, in order to better study differences between the various interaction modes. In addition two new metrics were devised for the investigation of two important research questions, namely the relation of input modality choice to unimodal efficiency (relative modality efficiency) and the measurement of the combined synergies in multimodal interaction modes (multimodal synergy).

In Chapter 5 the two unimodal and three multimodal form-filling systems on the desktop and PDA environments are evaluated and compared in terms of efficiency and user satisfaction. Context statistics and the "relative modality efficiency" metric reveal input modality patterns and their relation to modality efficiency. The latest clearly shows how context and interaction mode affects input selection and reveals a non-linear abrupt switch from GUI to speech modality when GUI input becomes less efficient (speech overuse). User statistics highlight the differences in usage patterns and high variability in terms of unimodal efficiency and preferences towards modalities and interaction modes. Multimodal synergy showcases how during multimodal interaction users exploited the synergies in a degree that helped them improve their performance compared to unimodal interaction.

In Chapter 6, a more detailed investigation of individual user behavior is provided with emphasis on two important factors that affect modality usage and related to speech modality, namely speech verbosity and speech error correction patterns. A statistical model for predicting input modality selection is described, evaluated and discussed.

Chapter 7 investigated the use of affective evaluation. Skin conductance and EEG data were collected and analyzed. The data exploited the Emotiv Epoc neuroheadset and a custom made GSR apparatus. Emotiv's affective metrics such as frustration, engagement and excitement revealed some interesting results. More advanced techniques such as the design of a cognitive load metric could be employed in the future to gain even more insight that could be helpful in multimodal interaction design.

8.2 Work items accomplished

The motivation for the work described in this dissertation was to overcome the limited input methods found in mobile devices, by using speech as an additional modality, in order to build more efficient, robust and natural interfaces that advance the state of the art and offer improved user interaction experience. Towards that end several steps had to be undertaken, research questions investigated and aims attained. The following list summarizes the series of steps undertaken:

- Identify and exploit the combined modality synergies in the design of multimodal interaction modes.
- Investigate multimodal turn taking and modality mix by designing three different multimodal interaction modes.
- Design and implement a system that allows both unimodal and multimodal interaction and can be used across different platforms such as PCs, PDAs and mobile devices.

- Develop a methodology for the evaluation of such systems with metrics that will help investigate input modality selection patterns and synergies.
- Study and analyze user evaluation results to compare the various interaction modes and identify user behavior patterns.
- Investigate usage patterns and devise a statistical model for prediction of input modality selection.
- Employ physiological signals such as skin conductance and EEG to evaluate the designed systems in terms of affective metrics.

8.3 Results

The detailed evaluation of unimodal and multimodal interaction modes yielded some results that can help us better understand human-machine interaction for multimodal dialogue systems. Here are some important conclusions from the analysis:

- Synergies between the speech and GUI interaction modalities exist in multimodal interfaces; Visual feedback (GUI output), input modality choice (selection of most efficient/effective modality) and error correction, all play important role. When properly incorporated in the design process of multimodal interaction systems they can yield significantly higher interface efficiency and user satisfaction.
- The design of the multimodal interface (turn taking and default input modality) can affect user behavior e.g. users selected input modality based on unimodal efficiency considerations more frequently in "Modality-Selection" compared to "Open-Mike" mode (excessive use of speech).
- Compared to unimodal modes, multimodal interaction modes are almost always better in terms of shorter interaction times due to input modality choice and error correction synergies; however they may show increased inactivity times due to modality selection overhead.
- Unimodal efficiency clearly affects input modality choice. When changing the relative efficiency of the input modality in multimodal interfaces, user input modality usage also changes; users tend to use the most efficient modality but biases also exist, especially towards the speech modality. This is highlighted by the non-linear abrupt switch from GUI to speech modality when GUI input becomes less efficient. Generally the modality selection choice by users becomes more clear as the difference in unimodal efficiency between modalities increases and may become blurry as the difference approaches zero.

- As shown using all evaluation metrics (objective, relative modality efficiency, synergy) there is a great variability between users, in terms of unimodal performance, exploitation of synergies, behavior patterns (e.g. speech verbosity, overrides) and preferences towards certain modalities or systems. This makes modality and user behavior prediction in general a difficult task; however such an effort could potentially yield more adaptive interfaces.

- Affective evaluation revealed that speech recognition errors and user confusion may result higher levels of frustration and/or excitement. These differences are also found between input type (GUI/speech) and interaction modes. These results along with further exploration of more affective and cognitive attributes such attention or cognitive load could provide valuable insights for the interaction design process.

8.4 Future work

A possible future step is to evaluate the interaction systems for varying levels of unimodal interface efficiency. In the current evaluation it was possible to control the GUI input efficiency (as it relates to attribute size) but not speech efficiency. A possible solution to controlling speech recognition error rates is to use to WOZ (Wizard of Oz) method resulting error free recognition or additive noise resulting high WER. Also, in the current evaluation there was a clear difference in unimodal efficiency between the two modalities, especially for the long attributes (where speech much faster) and a less clear one for the short ones.

In addition the evaluation scenarios were balanced, since the number of long and short attributes was almost the same. This made relative clear for users which modality to use; it would be interesting to investigate what would happen if un-balanced scenarios were used or unimodal efficiency difference became very small .Through these experiments multiple measurement points for modality usage, unimodal and multimodal interface efficiency will be obtained; these results could help to better understand the relationship between efficiency, user satisfaction and input modality usage in a broader range of situations.

Although the detailed objective evaluation metrics used for the study of user behavior have revealed a great deal of information, they are only the results of inner cognitive processing and emotional states taking place during evaluation. Incorporation of such knowledge could greatly advance the understanding of usage patterns in multimodal interaction. The affective evaluation performed indicated some interesting results in terms of affective metrics such as frustration, engagement and excitement. Yet more work towards this direction (section 7.6) could potentially reveal even more insights and guidelines for multimodal interface design and also help in designing a better adaptive model.

Appendix A

Multimodal system design and implemenation details

A.1 Evolution of the original system to a multimodal platform

The initial system that was used as a base for the development of the multimodal system described in this thesis was the Bell Labs Communicator Spoken Dialogue System. The original Communicator uses the BLSTIP [117] telephony platform and thus speech interaction could take place only through a phone. To further develop and explore multi-modality features on the Communicator, a highly flexible audio platform was designed and implemented which can be run on both desktop computers and mobile devices. The dialogue system (often used the term backend to describe it) could also be used with a text I/O interface (keyboard) and a first version of a GUI had been implemented (that is the GUI parser/intepreter already existed).

Thus the steps that had to be taken to transform the initial system to a fully functional multimodal platform was to:

- Build a high performance audio platform (audio playing/recording) for various platforms.
- Integrate the audio platform with the backend, the speech synthesizer and recognition components thus turning the dialogue system to a fully functional Spoken Dialogue System again.
- Improve the initial GUI view and port it to other GUI toolkits for various platforms.
- The design and the implementation of the multimodal interaction modes.

The fourth item is alreadyt described in Chapter 3. In the next sections the audio platform is described and the porting of the system to the two portable devices.

A.2 Audio platform

The audio code was implemented using Open Sound System (OSS) the default audio driver untill Linux kernel version 2.4 (newer versions use ALSA as the default audio driver but also allow for an emulation layer for OSS). Since the code for capturing and playing audio should take place in a heavily multithreading environment (GUI, several network sockets, control logic, etc) a high performance audio related code had to be written in order to satisfy two requirements. First no audio samples should be lost and second the CPU time needed for interfacing to audio should be in the range of a few msecs per second. To achieve this, a simple yet elegant solution that was devised was to transfer (read/write) data between the audio device buffer and the application buffer in interval periods exactly propotional to data generation/consumption rate. This way no samples are ever lost (no way of having buffer over-run/under-run) and and processing time is minimal (a few msces instead of blocking I/O).

The native version of the audio platform can be run in Linux based PCs but also the Zaurus Linux PDA. Since, it is implemented in C while the rest system is implemented in Java, JNI (Java Native Interface) was used to call native methods and transfer data between the C and Java parts of the code. To be able to address the issue of using the system to more devices, the audio platform was also implemented using JMF (Java Media Framework). This allowed to run the audio platform to all three major desktop platforms (Linux, Mac OS, Windows).

Although the audio platform could run natively on the PDA device, one important issue was how to be able to achieve both recording and playing at the same time using just the one audio jack found in the device (audio I/O is done through stereo headset). This functionality known as *barge-in*, is needed for example when the user starts speaking at the system, while the TTS is still active, in which case the TTS should stop. To achieve this the solution given was to exploit the stereo support of the jack and being able to open the audio device for both recording and playing simultaneously by using a separate channel in mono mode for each functionality (e.g. left for recording, right for playing).

The audio controller interfaces with the speech recognition and synthesis components through network sockets acting as a proxy to the Spoken Dialogue System. The SDS text output is sent through the audio controller to the FreeTTS speech synthesizer and the produced samples are sent back to the audio device for rendering. Likewise audio samples from the audio device are sent to the Bell Labs recognizer and the recognition result is sent to the SDS component. Voice activity detection (VAD) is used (when enabled) to only send speech samples to the recognizer only after voice detection is done, e.g. in “Open-Mike” mode (in that case the Speech button turns from yellow color to red to indicate the VAD event to the user). In case no speech is detected in a short time period, no samples are sent to the recognizer at all

and the SDS input becomes null, causing it to proceed to a new turn. The VAD implemented is based on the spectral energy of the signal and had an overall good performance in practice.

A.3 Porting the system to mobile devices

The two mobile devices used in the evaluation of the thesis are the Zaurus Linux PDA and the iPod touch from Apple. In this section some more details are provided for these two devices and the porting and implementation details.

A.3.1 Zaurus Linux PDA

The PDA device used was the Zaurus SL-5500 (Collie), the first popular Linux PDA, released outside Japan in 2001. It is based on the Intel SA-1110 StrongARM processor running at 206 MHz, has 64 MB of RAM and 16MB Flash, a built-in keyboard, CompactFlash (CF) slot, Secure Digital (SD) slot, and infrared port. WiFi could only be used with an external wireless CF card. A second Zaurus device SL-5000D, a developer edition of the SL-5500, containing just 32 MB of RAM was also available (this one was won in an interational software development contest).

Porting the whole system to the PDA

An initial thought was trying to port the whole system code base to the device. Since the system was written in Java (J2SE 1.2) with some parts (parsers and audio platform) implemented in C, the port of the Swing GUI to AWT GUI toolkit was needed in order to be compatible with the J2ME CDC based java virtual machine available in the PDA device. From the three different java virtual machines available for Zaurus (Jeode, SUN CDC and IBM's J9) and the various available ROMs (Operating System variants that can flashed to the device) the only combination that worked was J9 with TK[1] ROM.

Despite the intense code optimization (ranging from source code optimization to code obfuscation) in order to provide a distribution of the the system as fast as possible and with minimal size (to fit to the space and limited RAM size of the device) the system became fully functional but with a slow interaction compared to Desktop environment. Thus the configuration of only running the speech recognition and synthesis components remotely, with all rest dialogue system and I/O channels (GUI/audio) running locally on the device was abandoned. To achieve comparable to desktop performance, a new configuration needed that would move the dialogue system (which was the most resource demanding part) remotely on a PC and only run the GUI and audio channels to the device.

[1] http://www.thekompany.com/embedded/rom/

The client-server GUI protocol approach

Since the GUI is built automatically and refreshed in each turn from the dialogue component (which should be now run remotely) and the java version running on the PDA device did not support remote method invocation (RMI a java implementation of remote procedure call - RPC) the whole communication between the server (dialogue manager) and remote client (GUI running on the PDA) was implemented as a custom communication protocol. This proxy like approach allowed to transparently call remote methods between the two sides thus making possible the deployement of a remote GUI interface to compatible devices. Transforming the tightly integrated GUI component to an independent process running remotely had the advantage that porting the system to a new device only required the port of the GUI view. Additionaly the whole system could be run in a standalone or client-server mode with just a change of a configuration variable.

To achieve this, two different actions needed. First the original GUI related code was split to two parts; one was the view creation part (GUI controller) and the other the GUI view (toolkit) specific one (e.g. Swing/AWT view). This allowed to have different GUI view implementations with the exactly same GUI controller (recall the MVC pattern). The second step was to split the GUI controller in two parts; the first (GUI frontend) running on the device and the second (GUI backend) running on the server side. The plroxy-like protocol implementation employed multi-threading network techniques in order to allow for asynchronous two way communication between the two sides.

Following is a small part of the definition of the proxy protocol used. Methods calls have been substituted by request IDs; the code interpreting the request is responsible for also passing the needed arguments and then calling the original code on each side (GUI front-end or GUI back-end); all this is done completely transparently.

```
...
public static final byte OPERATION_CONTEXT_GET_TITLE = 13;
public static final byte OPERATION_EFORM_GET_SCORE = 14;
public static final byte OPERATION_EFORM_GET_SCORES = 15;
public static final byte OPERATION_PROTOTYPE_TREE_GET_BY_EFORM_SCORE = 16;

public static final byte OPERATION_AGENDA_IS_STATE_ACTIVE = 17;
public static final byte OPERATION_AGENDA_GET_CURRENT_ACT = 18;
public static final byte OPERATION_AGENDA_GET_CONTEXT_FOR_ACT = 19;
public static final byte OPERATION_AGENDA_FILL = 20;

public static final byte OPERATION_SEMANTICS_GET_CONTEXT = 21;
```

Figure A.1: The Zaurus Linux PDA device. The GUI is operated with a stylus and both virtual and hardware keyboard are available for use.

```
public static final byte OPERATION_SEMANTICS_SET_CONTEXT = 22;
...
```

A.3.2 iPod touch

The iPod touch, a device running iPhone OS 2.2 (and thus with same UI and interaction methods with the iPhone device) was used as the second mobile device. It features a ARM11 620 MHz CPU, with 128 MB RAM, 8 GB storage, built-in wifi and a $320X480$ touch screen. In contrast with the Zaurus PDA which follows a desktop-like GUI interface and is controlled via a stylus the iPhone uses a touch interface optimized for simple finger gestures operations on the screen (refer to section 2.6.2).

Thus instead of the precise pointing of the stylus on PDAs, the larger less precise footprint of finger on the screen has certain implications in the design of the screen components. For example, in contrast with the traditional form views in desktop-like GUIs for which both the field labels and components that contain the fields values (e.g. combo-box) can been fit in a single view, the corresponding form in the iPhone requires a two level view hierarchy. The main (top) view (a table-view according to iPhone terminology) holds just the field labels and the corresponding selected value in each table row (see Fig. 3.4(a)). By touching each row, a new detailed (two-level) view containing all the possible values the user can select from, is shown (Fig. 3.4(b)). A navigation bar indicates the depth level in the hierarchy; after the user scrolls and selects the desired value the detailed view dissapears and the main view is shown again.

Thus porting the system to the iPhone required effort in both redesigning the GUI according to iPhone HCI guidelines but also the implementation of the new GUI with a different set of tools and developing environments (use of Objective-C programming language, developing with

XCode environment on MAC OS X operating system). Apart from the view code part, the client-side proxy protocol (GUI frontend) had to be re-implemented, since it was decided to exploit the protocol designed during the Zaurus porting process. Issues such as cross-compiling and endianess had to be addressed since the new GUI front-end was written in C instead of Java. Again both the GUI frontend and view code was multi-thtreaded in order to allow for both asynchronous two way communication and for the relaying of all GUI related methods to view's main drawing thread.

Appendix B

Evaluation and additional results

This Appendix provides additional material relating to evaluation results not included in the main manuscript. In section B.1 the evaluation scenarios are described in detail. Section B.2 contains additional evaluation results for the relative modality efficiency metric.

B.1 Evaluation Scenarios

The five evaluation scenarios used are summarized next anD are also shown in the following five tables. Note that only attribute values in bold are the ones required, since the rest information has been inputed to the system during the previous turns. e.g. Quantas in second leg of second scenario has been inputed during the first leg. The five scenarios are:

- From Las-Vegas to Miami on July 10th in the morning with Northwest airlines.
- From Orlando to Boston on July 9th in the morning using Quantas airlines. Return on July 10th in the evening.
- From Miami to Vienna on July 6th in the morning using United airlines. Return on July 7th in the evening. Reserved hotel is Four Seasons.
- From Tucson to Phoenix on July 6th in the morning using Southwest airlines. Return on July 8th, anytime. Car rental of a wagon type car from Budget.
- From Tucson to Orlando on July 6th in the morning using TWA airlines. Next flight to Phoenix on July 14th, anytime. Return to Tucson on July 16th in the evening.

Scenario 1 - One way trip

Departure City	Arrival City	Departure Date	Departure Time	Airline
Las-Vegas	**Miami**	**July 10**	**Morning**	**Northwest**

Scenario 2 - Round trip

Departure City	Arrival City	Departure Date	Departure Time	Airline
Orlando	**Boston**	**July 9**	**Morning**	**Quantas**
Boston	**Orlando**	**July 10**	**Evening**	Quantas

Scenario 3 - Round trip with hotel resrervation

Departure City	Arrival City	Departure Date	Departure Time	Airline
Miami	**Vienna**	**July 6**	**Morning**	**United**
Vienna	**Miami**	**July 7**	**Evening**	United

Arrival City	Arrival Date	Departure Date	Hotel Name
Vienna	July 6	July 7	**Four Seasons**

Scenario 4 - Round trip with car resrervation

Departure City	Arrival City	Departure Date	Departure Time	Airline
Tuscon	**Phoenix**	**July 6**	**Morning**	**Southwest**
Phoenix	**Tuscon**	**July 8**	**Anytime**	Southwest

Arrival City	Arrival Date	Departure Date	Car Type	Car Company
Phoenix	July 6	July 8	**Station Wagon**	**Budget**

Scenario 5 - Three-way trip

Departure City	Arrival City	Departure Date	Departure Time	Airline
Tuscon	**Orlando**	**July 6**	**Morning**	**TWA**
Orlando	**Phoenix**	**July 14**	**Anytime**	TWA
Phoenix	Tuscon	**July 16**	**Evening**	TWA

B.2 Relative modality efficiency for inactivity and interaction times

In this section more detailed results concerning relative modality efficency for inactivity and interaction times are provided in Fig. B.1 and Fig. B.2 respectively.

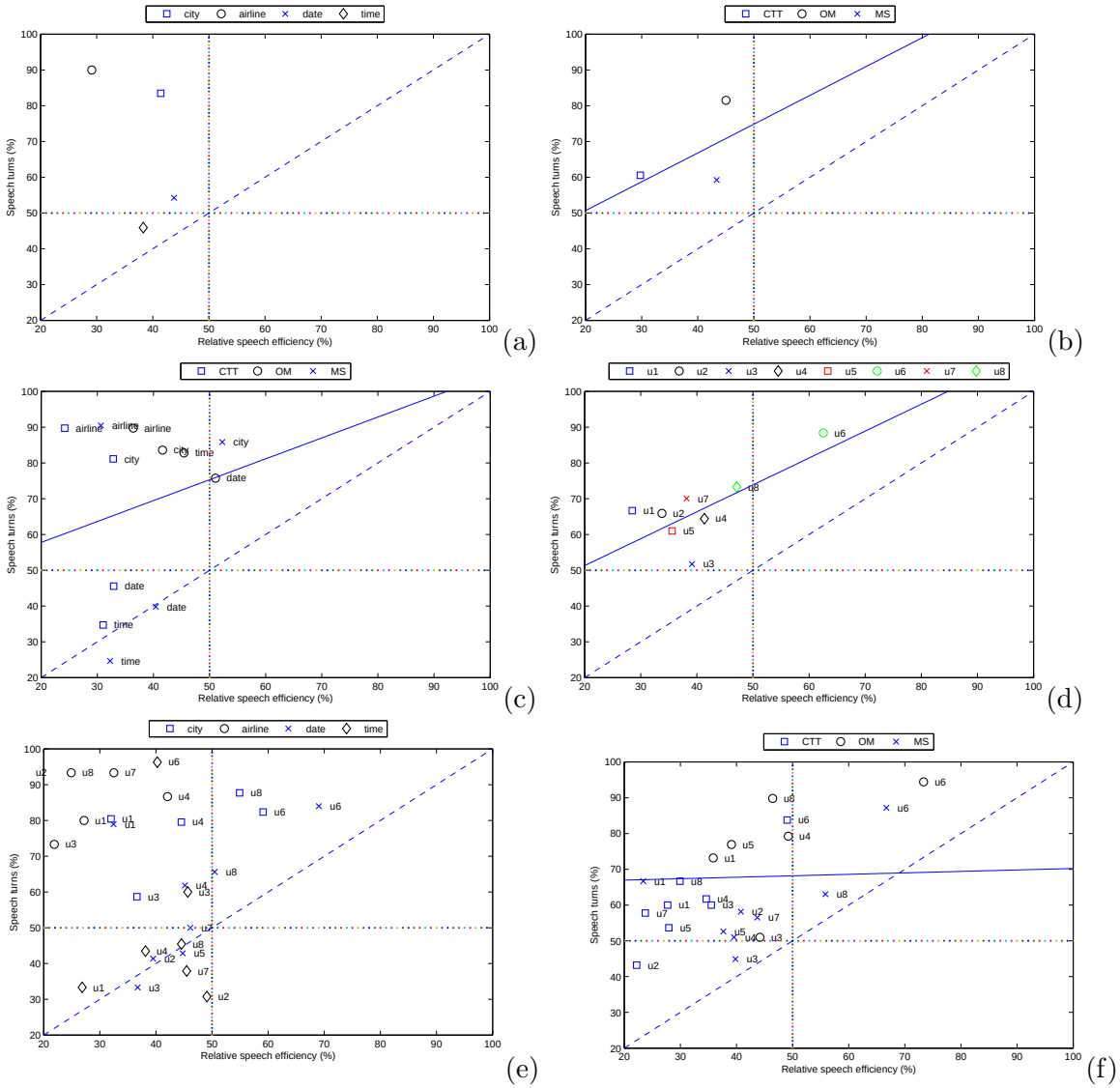


Figure B.1: Speech modality usage (QU_s) as a function of relative speech modality efficiency - inactivity times are shown. (a) context averaged over users and interaction modes (4 points). (b) interaction mode averaged over users and contexts (3 points). (c) combined data points for interaction modes and contexts over users (12 points). (d) user averaged over contexts and interaction modes (8 points). (e) combined data points for users and context over interaction modes (32 points). (f) combined data points for modes and users over contexts (24 points).

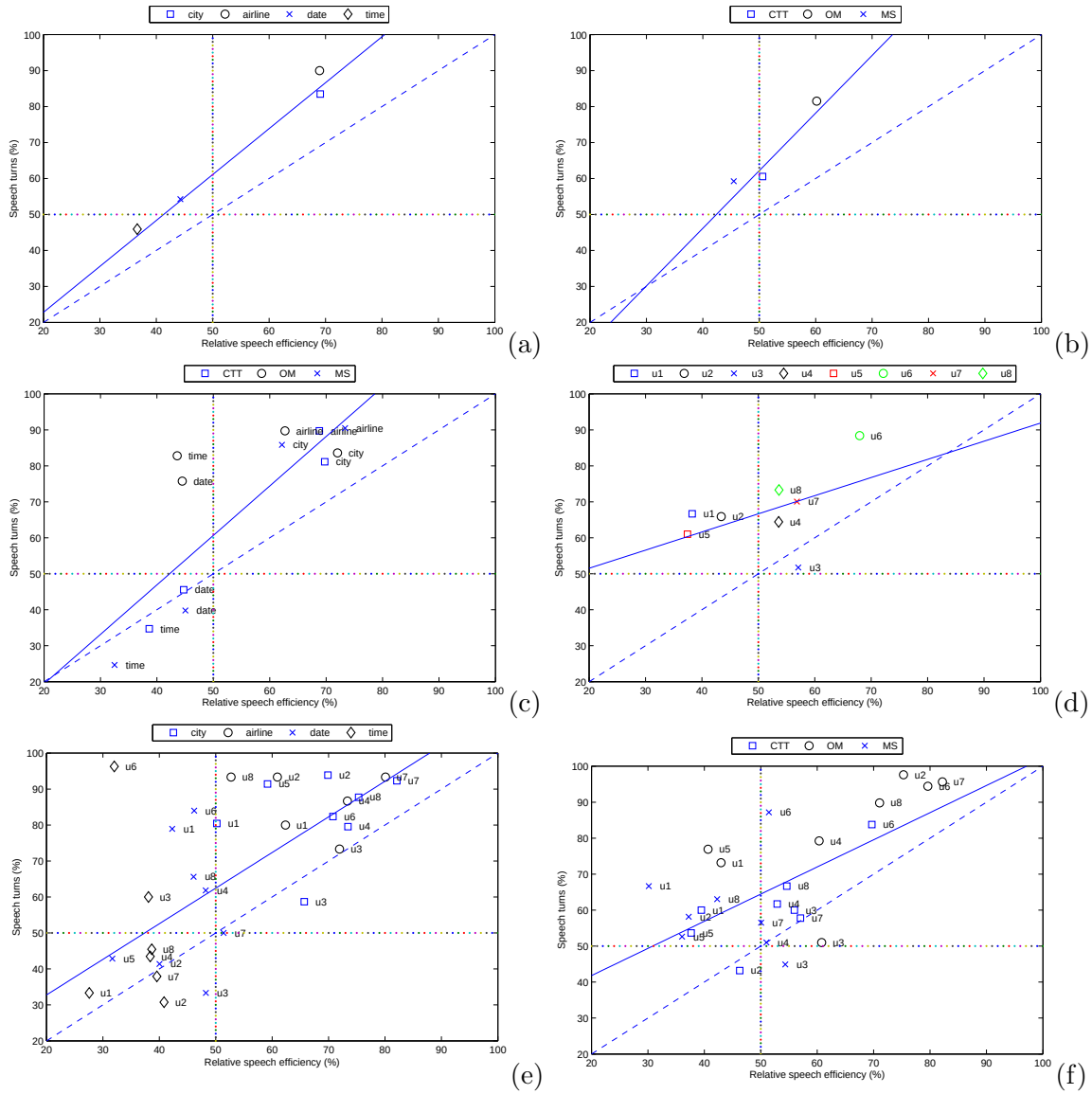


Figure B.2: Speech modality usage (QU_s) as a function of relative speech modality efficiency - interaction times are shown. (a) context averaged over users and interaction modes (4 points). (b) interaction mode averaged over users and contexts (3 points). (c) combined data points for interaction modes and contexts over users (12 points). (d) user averaged over contexts and interaction modes (8 points). (e) combined data points for users and context over interaction modes (32 points). (f) combined data points for modes and users over contexts (24 points).

Appendix C

List of Publications

Multimodal Interfaces:

[1] A. Potamianos and M. Perakakis. *Design principles for multimodal spoken dialogue systems.* In P. Maragos, A. Potamianos, P. Gros, editor, Multimodal Processing and Interaction: Audio, Video, Text, New York, 2008. Springer-Verlag, New York, NY.

[2] A. Potamianos and M. Perakakis. *Human-computer interfaces to multimedia content: a review.* In P. Maragos, A. Potamianos, P. Gros, editor, Multimodal Processing and Interaction: Audio, Video, Text, New York, 2008. Springer-Verlag, New York, NY.

[3] M. Perakakis and A. Potamianos. *A study in efficiency and modality usage in multimodal form filling systems.* IEEE Transactions on Audio, Speech and Language Processing, Vol. 16, pp. 1194-1206, Aug. 2008.

[4] A. Potamianos, E. Fosler-Lussier, E. Ammicht, and M. Perakakis. *Information Seeking Spoken Dialogue Systems Part II: Multimodal Dialogue.* IEEE Transactions on Multimedia, 9(3):550-566, Apr. 2007.

[5] M. Perakakis and A. Potamianos. *Multimodal system evaluation using modality efficiency and synergy metrics.* Proceedings of International Conference on Multimodal Interfaces (ICMI 2008), Chania, Greece, Oct. 2008.

[6] M. Perakakis and A. Potamianos. *The effect of input mode on inactivity and interaction times of multimodal systems.* Proceedings of International Conference on Multimodal Interfaces (ICMI 2007), Nagoya, Japan, pages 102-109.

[7] M. Perakakis, M. Toutoudakis, and A. Potamianos. *Blending speech and visual input in Multimodal Dialogue Systems.* IEEE Spoken Language Technology (SLT) Workshop, Aruba, 2006, pages 142-145, 2006.

[8] M. Toutoudakis, M. Perakakis, and A. Potamianos. *Mode selection in multimodal dialogue systems.* Invited paper to International Conference on Intelligent Systems And Computing (ISYC - Agia Napa, Cyprus), 2006.

[9] M. Perakakis, M. Toutoudakis, and A. Potamianos. *Modality selection for multimodal dialogue systems.* International Conference on Multimodal Interfaces (ICMI 2005), Trento, Italy, 2005.

Augmented Reality Interfaces:

[10] T. Salonen, M. Hakkarainen, T. Kannetis, M. Perakakis, A. Potamianos, and C. Woodward. *Demonstration of assembly work using augmented reality.* Proceedings of the 6th ACM international conference on Image and video retrieval, pages 120-123, 2007.

[11] S. Siltanen, M. Hakkarainen, O. Korkalo, T. Salonen, J. Saaski, C. Woodward, T. Kannetis, M. Perakakis, and A. Potamianos. *Multimodal User Interface for Augmented Assembly.* IEEE Multimedia Signal Processing (MMSP) Workshop, Chania, 2007, pages 78-81, 2007.

Distributed Speech Recognition

[12] V. Digalakis, L. Neumeyer, and M. Perakakis.*Quantization of cepstral parameters for speech recognition over the World Wide Web.* IEEE Journal on Selected Areas in Communications, 17(1):82-90, 1999.

[13] V. Digalakis, L. Neumeyer, and M. Perakakis. *Product-code vector quantization of cepstral parameters for speech recognition over the WWW.* Fifth International Conference on Spoken Language Processing, 1998.

[14] V. Digalakis, L. Neumeyer, and M. Perakakis. *Quantization of cepstral parameters for speech recognition over the WWW.* Proceedings of ICASSP'98, (2):989-992, 1998.

Bibliography

[1] S. K. Card, A. Newell, and T. P. Moran, *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, 1983.

[2] I.S. MacKenzie and S.X. Zhang, "The immediate usability of Graffiti," in *Proceedings of Graphics Interface*. Citeseer, 1997, pp. 129–137.

[3] Shumin Zhai and Per-Ola Kristensson, "Shorthand writing on stylus keyboard," in *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, New York, NY, USA, 2003, pp. 97–104, ACM.

[4] David J. Ward, Alan F. Blackwell, and David J. C. MacKay, "Dasher—a data entry interface using continuous gestures and language models," in *UIST '00: Proceedings of the 13th annual ACM symposium on User interface software and technology*, New York, NY, USA, 2000, pp. 129–137, ACM.

[5] A. Potamianos, E. Fosler-Lussier, E. Ammicht, and M. Perakakis, "Information seeking spoken dialogue systems - Part II: Multimodal dialogue," *IEEE Transactions on Multimedia*, vol. 9, no. 3, pp. 550–566, 2007.

[6] H H Jasper, "The 10-20 electrode system of the internation federation," *Electroencephalography and Clinical Neurophysiology*, vol. 10, pp. 371–375, 1958.

[7] Arnaud Delorme and Scott Makeig, "Eeglab: an open source toolbox for analysis of single-trial eeg dynamics including independent component analysis.," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[8] J. Gustafson, *Developing Multimodal Spoken Dialogue Systems. Empirical Studies of Human-Computer Interaction*, Ph.D. thesis, Department of Speech, Music and Hearing, KTH, 2002.

[9] J. Lai and N. Yankelovich, "Conversational speech interfaces," in *The Human-Computer Interaction Handbook: Fundamentals, evolving technologies and emerging applications*, pp. 698–713. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, 2003.

[10] S. Oviatt, "Multimodal interfaces," in *The Human-Computer Interaction Handbook: Fundamentals, evolving technologies and emerging applications*, J. Jacko & A. Sears, Ed., pp. 286–304. Lawrence Erlbaum: New Jersey, 2003.

[11] D. B. Koons, C. J. Sparrell, and K. R. Thórisson, "Integrating simultaneous input from speech, gaze, and hand gestures," in *Proc. of Artificial Intelligence Workshop on Intelligent Multimedia Interfaces*, Menlo Park, CA, USA, 1993, pp. 257–276.

[12] A. Waibel, M. Tue Vo, P. Duchnowski, and S. Manke, "Multimodal interfaces," *Artif. Intell. Rev*, vol. 10, no. 3-4, pp. 299–319, 1996.

[13] S. Oviatt, "Mutual disambiguation of recognition errors in a multimodal architecture," *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pp. 576–583, 1999.

[14] S. Oviatt, "Multimodal interfaces," in *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*, J. Jacko & A. Sears, Ed., pp. 286–304. Lawrence Erlbaum: New Jersey, 2003.

[15] P. R. Cohen, M. Johnston, D. McGee, S. Oviatt, J. Pittman, I. Smith, L. Chen, and J. Clow, "Quickset: multimodal interaction for distributed applications," in *Proc. of ACM International Conference on Multimedia*, Seattle, Washington, United States, 1997, pp. 31–40.

[16] M. Johnston, S. Bangalore, G. Vasireddy, A. Stent, P. Ehlen, M. Walker, S. Whittaker, and P. Maloor, "MATCH: an architecture for multimodal dialogue systems," in *Proc. of the 40th Annual Meeting on Association for Computational Linguistics*, Philadelphia, Pennsylvania, 2002, pp. 376–383.

[17] S. Dusan, G. J. Gadbois, and J. Flanagan, "Multimodal Interaction on PDA's Integrating Speech and Pen Inputs," in *Proc. of Eighth European Conference on Speech Communication and Technology*, Geneva, Switzerland, 2003, pp. 2225–2228.

[18] R. A. Bolt, "Put-That-There : Voice and gesture at the graphics interface," in *Proc. of Computer Graphics and Interactive Techniques*, Seattle, Washington, United States, 1980, pp. 262–270.

[19] X. Huang, A. Acero, C. Chelba, L. Deng, D. Duchene, J. Goodman, H. Hon, D. Jacoby, L. Jiang, and R. Loynd, "MiPaD: A Next Generation PDA Prototype," in *Proc. of the International Conference on Spoken Language Processing*, Beijing, China, 2000, pp. 33–36.

[20] L. Deng, K. Wang, A. Acero, H. W. Hon, J. Droppo, C. Boulis, Y. Y. Wang, D. Jacoby, M. Mahajan, and C. Chelba, "Distributed speech processing in MiPaD's multimodal user interface," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 8, pp. 605–619, 2002.

[21] L. Nigay and J. Coutaz, "A design space for multimodal systems: concurrent processing and data fusion," in *Proc. of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*, Amsterdam, The Netherlands, 1993, pp. 172–178.

[22] V. Bilici, E. Krahmer, S. te Riele, and R. Veldhuis, "Preferred Modalities in Dialogue Systems," in *Proc. of Sixth International Conference on Spoken Language Processing*, Beijng, China, 2000, pp. 727–730.

[23] N. O. Bernsen and L. Dybkjaer, "Is Speech The Right Thing For Your Application?," in *Proc. of Fifth International Conference on Spoken Language Processing*, Sydney, Australia, 1998, pp. 3209–3212.

[24] M. Grasso, D. Ebert, and T. Finin, "The Integrality of Speech in Multimodal Interfaces," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 5, no. 4, pp. 303–325, 1998.

[25] P. Cohen and S. Oviatt, "The Role of Voice in Human-Machine Communication," in *Voice Communication Between Humans and Machines*, pp. 34–75. National Academy Press, Washington D.C., 1994.

[26] P. Cohen, M. Johnston, D. McGee, S. Oviatt, J. Clow, and I. Smith, "The Efficiency of Multimodal Interaction: a Case Study," in *Proc. of Fifth International Conference on Spoken Language Processing*, Sydney, Australia, 1998, pp. 249–252.

[27] B. Suhm, B. Myers, and A. Waibel, "Multimodal error correction for speech user interfaces," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 8, no. 1, pp. 60–98, 2001.

[28] H. Mitchard and J. Winkles, "Experimental Comparisons of Data Entry by Automated Speech Recognition, Keyboard, and Mouse.," *Human Factors*, vol. 44, no. 2, pp. 198–210, 2002.

[29] L. M. Reeves, J. C. Martin, M. McTear, T. V. Raman, K. M. Stanney, H. Su, Q. Y. Wang, J. Lai, J. A. Larson, and S. Oviatt, "Guidelines for multimodal user interface design," *Communications of the ACM*, vol. 47, no. 1, pp. 57–59, 2004.

[30] K. Stanney, S. Samman, L. Reeves, K. Hale, W. Buff, C. Bowers, B. Goldiez, D. Nicholson, and S. Lackey, "A Paradigm Shift in Interactive Computing: Deriving Multimodal Design Principles from Behavioral and Neurological Foundations," *International Journal of Human-Computer Interaction*, vol. 17, no. 2, pp. 229–257, 2004.

[31] D.J. Litman and S. Pan, "Designing and Evaluating an Adaptive Spoken Dialogue System," *User Modeling and User-Adapted Interaction*, vol. 12, no. 2, pp. 111–137, 2002.

[32] J. Polifroni, L. Hirschman, S. Seneff, and V. Zue, "Experiments in evaluating interactive spoken language systems," *Proceedings of the workshop on Speech and Natural Language*, pp. 28–33, 1992.

[33] M. Walker, C. Kamm, and D. Litman, "Towards developing general models of usability with PARADISE," *Natural Language Engineering*, vol. 6, no. 3&4, pp. 363–377, 2001.

[34] N. Beringer, U. Kartal, K. Louka, F. Schiel, and U. Türk, "Promise: A procedure for multimodal interactive system evaluation," in *Proc. of the LREC Workshop on Multimodal Resources and Multimodal Systems Evaluation*, Las Palmas, 2002, pp. 77–80.

[35] L.B. Larsen, "Issues in the evaluation of spoken dialogue systems using objective and subjective measures," *IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 209–214, 2003.

[36] A. Oppenheim, *Questionnaire Design, Interviewing and Attitude Measurement*, Continuum International Publishing Group, 1992.

[37] J. Chin, V. Diehl, and L. Norman, *Development of an instrument measuring user satisfaction of the human-computer interface*, ACM Press New York, NY, USA, 1988.

[38] K. Hone and R. Graham, "Towards a tool for the Subjective Assessment of Speech System Interfaces (SASSI)," *Natural Language Engineering*, vol. 6, no. 3&4, pp. 287–303, 2001.

[39] R. Mason, R. Gunst, and J. Hess, *Statistical design and analysis of experiments*, Wiley, 1989.

[40] J. Myers and A. Well, *Research Design and Statistical Analysis*, Lawrence Erlbaum Associates, 2003.

[41] A. Potamianos and M. Perakakis, "Human-computer interfaces to multimedia content a review," in *Multimodal Processing and Interaction: Audio, Video, Text*, P. Maragos, A. Potamianos, and P. Gros, Eds., vol. 33 of *Multimedia Systems and Applications*, pp. 49–87. Springer US, 2008.

[42] "HCI definition," http://en.wikipedia.org/wiki/Human-Computer_Interaction.

[43] M. L. Dertouzos, *The Unfinished Revolution: Making Computers Human-Centric*, HarperCollins Publishers, 2001.

[44] B. Shneiderman, *Leonardo's Laptop: Human Needs and the New Computing Technologies*, MIT Press, Cambridge, MA, USA, 2002.

[45] Abowd G. Dix A., Finlay J. and Beale R., *Human-Computer Interaction*, Prentice Hall, 2004.

[46] C.D. Wickens and Hollands J. G., *Engineering psychology and human performance*, Prentice Hall, NJ, 2000.

[47] G. Salvendy, *Handbook of Human Factors and Ergonomics*, John Wiley & Sons, Inc. New York, NY, USA, 2005.

[48] B. Shneiderman, *Designing the user interface: strategies for effective human-computer interaction*, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1997.

[49] G. A. Miller, "The magical number seven, plus or minus two: Some limits on our capacity for information processing," *Psychological Review*, vol. 63, no. 2, pp. 81–97, 1956.

[50] B.E. Stein and M.A. Meredith, *The merging of the senses*, MIT Press Cambridge, MA, 1993.

[51] M.D. Byrne, "Cognitive architectures," in *The human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications*, J. Jacko & A. Sears, Ed., pp. 97–117. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, 2003.

[52] Wilbert O. Galitz, *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques*, John Wiley & Sons, 2002.

[53] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns: elements of reusable object-oriented software*, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1995.

[54] "W3C Multimodal Architecture and Interfaces," http://www.w3.org/TR/mmi-arch.

[55] J. Nielsen, *Usability Engineering*, Academic Press, 1993.

[56] J. Nielsen and R. Molich, "Heuristic evaluation of user interfaces," *Proceedings of the SIGCHI conference on Human factors in computing systems: Empowering people*, pp. 249–256, 1990.

[57] S. Sherr, *Input Devices*, Academic Press, Inc. Orlando, FL, USA, 1990.

[58] S.K. Card, J.D. Mackinlay, and G.G. Robertson, "A morphological analysis of the design space of input devices," *ACM Transactions on Information Systems (TOIS)*, vol. 9, no. 2, pp. 99–122, 1991.

[59] S. Zhai and P. Milgram, *Quantifying coordination in multiple DOF movement and its application to evaluating 6 DOF input devices*, ACM Press/Addison-Wesley Publishing Co. New York, NY, USA, 1998.

[60] R.M. Baecker et al., *Readings in human-computer interaction: toward the year 2000*, Morgan Kaufmann Publishers, 1995.

[61] W. Chou and B.H. Juang, *Pattern Recognition in Speech and Language Processing*, CRC Press, Inc. Boca Raton, FL, USA, 2002.

[62] L. Rabiner and B.H. Juang, *Fundamentals of speech recognition*, Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1993.

[63] X. Huang, A. Acero, H.W. Hon, et al., *Spoken language processing*, Prentice Hall PTR Upper Saddle River, NJ, 2001.

[64] S. Young, "A review of large-vocabulary continuous-speech," *Signal Processing Magazine, IEEE*, vol. 13, no. 5, 1996.

[65] G.B. Varile and A. Zampolli, *Survey of the State of the Art in Human Language Technology*, Cambridge University Press, 1997.

[66] M.F. McTear, "Spoken dialogue technology: enabling the conversational user interface," *ACM Computing Surveys*, vol. 34, no. 1, pp. 90–169, 2002.

[67] J. Lai and N. Yankelovich, "Conversational speech interfaces," in *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*, pp. 698–713. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, 2003.

[68] M. Turunen, *Jaspis - A Spoken Dialogue Architecture and its Applications*, Ph.D. thesis, University of Tampere, Department of Information Studies, 2004.

[69] C. Benoit, J.C. Martin, C. Pelachaud, L. Schomaker, and B. Suhm, "Audio-visual and multimodal speech systems," *Handbook of Standards and Resources for Spoken Language Systems*, 2000.

[70] "W3C MultiModal Interaction Requirements," http://www.w3.org/2002/mmi-reqs/.

[71] Wolfgang Wahlster, *SmartKom: Foundations of Multimodal Dialogue Systems (Cognitive Technologies)*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[72] R. Engel and N. Pfleger, *SmartKom: Foundations of Multimodal Dialogue Systems (Cognitive Technologies)*, chapter Modality Fusion, pp. 223–236, Springer-Verlag New York, Inc., 2006.

[73] M. Johnston and S. Bangalore, "Finite-state multimodal integration and understanding," *Nat. Lang. Eng.*, vol. 11, no. 2, pp. 159–187, 2005.

[74] M. Minksy, "A framework for representing knowledge," *The psychology of computer vision*, pp. 211–277, 1977.

[75] M. Kay, "Functional Grammar," *Proceedings of the Fifth Annual Meeting of the Berkeley Linguistics Society*, pp. 142–158, 1979.

[76] B. Carpenter, *The logic of typed feature structures*, Cambridge University Press New York, NY, USA, 1992.

[77] A. Shaikh, S. Juth, A. Medl, I. Marsic, C. Kulikowski, and J. Flanagan, "An architecture for multimodal information fusion," *Proceedings of the Workshop on Perceptual User Interfaces (PUI 97)*, pp. 91–93, 1997.

[78] M. Johnston, P.R. Cohen, D. McGee, S.L. Oviatt, J.A. Pittman, and I. Smith, "Unification-based multimodal integration," *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pp. 281–288, 1997.

[79] M. Johnston, "Unification-based multimodal parsing," *Proceedings of COLING-ACL*, vol. 98, pp. 624–630, 1998.

[80] P.R. Cohen, M. Johnston, D. McGee, S. Oviatt, J. Pittman, I. Smith, L. Chen, and J. Clow, "QuickSet: multimodal interaction for distributed applications," *Proceedings of the fifth ACM international conference on Multimedia*, pp. 31–40, 1997.

[81] P. Poller and V. Tschernomas, *SmartKom: Foundations of Multimodal Dialogue Systems (Cognitive Technologies)*, chapter Multimodal Fission and Media Design, pp. 379–400, Springer-Verlag New York, Inc., 2006.

[82] E. Andrè and T. Rist, "Presenting Through Performing. On the Use of Multiple Lifelike Characters in Knowledge-Based Presentation," *Proceedings of the Second International Conference on Intelligent User Interfaces (IUI 2000)*, pp. 1–8, 2000.

[83] W. Wahlster, "Towards Symmetric Multimodality: Fusion and Fission of Speech, Gesture, and Facial Expression," *KI*, pp. 1–18, 2003.

[84] J. Cassell, T. Bickmore, M. Billinghurst, L. Campbell, K. Chang, H. Vilhjálmsson, and H. Yan, "Embodiment in conversational interfaces: Rea," *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pp. 520–527, 1999.

[85] S. Oviatt, R. Coulston, S. Tomko, B. Xiao, R. Lunsford, M. Wesson, and L. Carmichael, "Toward a theory of organized multimodal integration patterns during human-computer interaction," *Proceedings of the 5th international conference on Multimodal interfaces*, pp. 44–51, 2003.

[86] P.R. Cohen and S.L. Oviatt, "The role of voice in human-machine communication," *Voice communication between humans and machines table of contents*, pp. 34–75, 1994.

[87] S. Oviatt, "Ten myths of multimodal interaction," *Communications of the ACM*, vol. 42, no. 11, pp. 74–81, 1999.

[88] J.G. Neal and S.C. Shapiro, "Intelligent Multi-Media Interface Technology," *ACM SIGCHI Bulletin*, vol. 20, no. 1, pp. 75–76, 1988.

[89] J. Siroux, M. Guyomard, F. Multon, and C. Remondeau, "Oral and gestural activities of the users in the georal system," *International Conference on Cooperative Multimodal Communication (CMC'95)*, vol. 2, pp. 287–298, 1995.

[90] "The NICE (Natural Interactive Communication for Edutainment) project," http://www.niceproject.com/.

[91] S. V. Mulken, E. André, and J. Müller, "The persona effect: How substantial is it?," in *HCI '98: Proceedings of HCI on People and Computers XIII*, London, UK, 1998, pp. 53–66, Springer-Verlag.

[92] A.K. Jain and A. Ross, "Multibiometric systems," *Communications of the ACM*, vol. 47, no. 1, pp. 34–40, 2004.

[93] L. Rothrock, R. Koubek, F. Fuchs, M. Haas, and G. Salvendy, "Review and reappraisal of adaptive interfaces: toward biologically inspired paradigms," *Theoretical Issues in Ergonomics Science*, vol. 3, no. 1, pp. 47–84, 2002.

[94] A. Jameson, "Adaptive interfaces and agents," in *The human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications*, J. Jacko & A. Sears, Ed., pp. 305–330. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, 2003.

[95] R.B. Segal and J.O. Kephart, "Swiftfile: An intelligent assistant for organizing e-mail," *Proceedings of AAAI 2000 Spring Symposium on Adaptive User Interfaces*, pp. 107–112, 2000.

[96] E. Horvitz, J. Breese, D. Heckerman, D. Hovel, and K. Rommelse, "The Lumiere Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users," *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pp. 256–265, 1998.

[97] D.J. Litman and S. Pan, "Predicting and adapting to poor speech recognition in a spoken dialogue system," *Proc. of the Seventeenth National Conference on Artificial Intelligence, AAAI-2000*, 2000.

[98] D. Bohus and A. I. Rudnicky, "Error handling in the ravenclaw dialog management framework," in *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Morristown, NJ, USA, 2005, pp. 225–232.

[99] "W3C MultiModal Interaction Working Group : Multimodal Interaction Framework," http://www.w3.org/TR/mmi-framework/.

[100] Fred Paas, Juhani Tuovinen, Huib Tabbers, and Pascal Van Gerven, "Cognitive load measurement as a means to advance cognitive load theory," *Educational Psychologist*, vol. 38, no. 1, pp. 63–71, 2003.

[101] "W3C Mobile Web Best Practices," http://www.w3.org/TR/mobile-bp/.

[102] Réjean Plamondon and Sargur N. Srihari, "On-line and off-line handwriting recognition: A comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 63–84, 2000.

[103] S. Jaeger, S. Manke, and A. Waibel, "Npen++: An on-line handwriting recognition system," in *in 7th International Workshop on Frontiers in Handwriting Recognition*, 2000, pp. 249–260.

[104] S. Seneff, E. Hurley, R. Lau, C. Pao, P. Schmid, and V. Zue, "GALAXY-II: A Reference Architecture for Conversational System Development," *Fifth International Conference on Spoken Language Processing*, 1998.

[105] S. Seneff, R. Lau, and J. Polifroni, "Organization, Communication, and Control in the GALAXY-II Conversational System," *Sixth European Conference on Speech Communication and Technology*, 1999.

[106] S. Kumar and P.R. Cohen, "Towards a fault-tolerant multi-agent system architecture," *Proceedings of the fourth international conference on Autonomous agents*, pp. 459–466, 2000.

[107] D.L. Martin, "The open agent architecture: A framework for building distributed software system," *Applied Artificial Intelligence*, vol. 13, no. 1, pp. 91–128, 1999.

[108] M. Turunen and J. Hakulinen, "Jaspisˆ 2-An Architecture for Supporting Distributed Spoken Dialogues," *Eighth European Conference on Speech Communication and Technology*, pp. 1913–1916, 2003.

[109] F. Flippo, A. Krebs, and I. Marsic, "A framework for rapid development of multimodal interfaces," *Proceedings of the 5th International Conference on Multimodal interfaces*, pp. 109–116, 2003.

[110] C. Elting, S. Rapp, G. Möhler, and M. Strube, "Architecture and implementation of multimodal plug and play," *Proceedings of the 5th international conference on Multimodal interfaces*, pp. 93–100, 2003.

[111] "Apple Human Interface Guidelines," http://developer.apple.com/documentation/UserExperience/Conceptual/OSXHIGuidelines/.

[112] "Apple iPhone Human Interface Guidelines," http://developer.apple.com/documentation/iPhone/Conceptual/iPhoneHIG/iPhoneHIG.pdf.

[113] "IBM WebSphere Voice Server," http://www-306.ibm.com/software/pervasive/voice_server.

[114] "W3C MultiModal Interaction Working Group : System and Environment Framework," http://www. w3.org/TR/sysenv/.

[115] "W3C Extensible MultiModal Annotation markup language (EMMA)," http://www.w3.org/TR/emma/.

[116] "W3C Ink Markup Language," http://www.w3.org/TR/InkML/.

[117] Q. Zhou, A. Saad, and S. Abdou, "An enhanced BLSTIP dialogue research platform," in *Proc. of Sixth International Conference on Spoken Language Processing*, Beijng, China, 2000, pp. 1061–1064.

[118] "FreeTTS," http://freetts.sourceforge.net/docs/.

[119] A. Potamianos, E. Ammicht, and H.-K. Kuo, "Dialogue management in the Bell Labs communicator system," in *Proc. of Sixth International Conference on Spoken Language Processing*, Beijng, China, 2000, pp. 603–606.

[120] A. Potamianos, E. Ammicht, and E. Fosler-Lussier, "Modality tracking in the multimodal Bell Labs Communicator," in *Proc. of Automatic Speech Recognition and Understanding Workshop*, St. Thomas, U.S. Virgin Islands, 2003, pp. 192–197.

[121] M. A. Walker, D. J. Litman, C. A. Kamm, and A. Abella, "PARADISE: A framework for evaluating spoken dialogue agents," in *Proc. of the Association for Computational Linguistics (ACL)*, Somerset, New Jersey, 1997, pp. 271–280.

[122] R W Picard, *Affective Computing*, MIT Press, 1997.

[123] Gert Pfurtscheller and F H Lopes Da Silva, "Event-related eeg/meg synchronization and desynchronization: basic principles.," *Clinical Neurophysiology*, vol. 110, no. 11, pp. 1842–1857, 1999.

[124] Christina M. Krause, "Cognition- and memory-related ERD/ERS responses in the auditory stimulus modality.," *Progress in brain research*, vol. 159, pp. 197–207, 2006.

[125] Robert Plutchik, "The nature of emotions," *American Scientist*, vol. 89, no. 4, pp. 344+, 2001.

[126] Richard J Davidson, "What does the prefrontal cortex do in affect: perspectives on frontal eeg asymmetry research.," *Biological Psychology*, vol. 67, no. 1-2, pp. 219–233, 2004.

[127] David Grimes, Desney S. Tan, Scott E. Hudson, Pradeep Shenoy, and Rajesh P.N. Rao, "Feasibility and pragmatics of classifying working memory load with an electroencephalograph," in *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*. 2008, CHI '08, Florence, Italy, pp. 835–844, ACM.

[128] Alan Gevins and Michael E Smith, "Neurophysiological measures of cognitive workload during human-computer interaction," *Theoretical Issues in Ergonomics Science*, vol. 4, no. 1, pp. 113–131, 2003.

[129] P.J. Lang, "The emotion probe: Studies of motivation and attention," *American psychologist*, vol. 50, pp. 372–372, 1995.

[130] Yu Shi, Natalie Ruiz, Ronnie Taib, Eric Choi, and Fang Chen, "Galvanic skin response (gsr) as an index of cognitive load," in *CHI '07 extended abstracts on Human factors in computing systems*, New York, NY, USA, 2007, CHI EA '07, pp. 2651–2656, ACM.

[131] Benjamin Blankertz, Michael Tangermann, Florin Popescu, Matthias Krauledat, Siamac Fazli, Marton Donaczy, Gabriel Curio, and Klaus-Robert Mller, "The berlin brain-computer interface," *Computational Intelligence Research Frontiers*, vol. 5050, pp. 79–101, 2008.

[132] T P Jung, S Makeig, C Humphries, T W Lee, M J McKeown, V Iragui, and Terrence J Sejnowski, "Removing electroencephalographic artifacts by blind source separation.," *Psychophysiology*, vol. 37, no. 2, pp. 163–178, 2000.

[133] A T Campbell, Tanzeem Choudhury, Shaohan Hu, Hong Lu, Mashfiqui Rabbi, R D S Raizada, and M K Mukerjee, "Neurophone : Brain-mobile phone interface using a wireless eeg headset categories and subject descriptors," *Design*, pp. 3–8, 2010.

[134] Ricardo Chavarriaga and Jos Del R Millan, "Learning from eeg error-related potentials in noninvasive brain-computer interfaces.," *IEEE Transactions on Neural and Rehabilitation Systems Engineering*, vol. 18, no. 4, pp. 381–388, 2010.