

# Towards Intelligent Interfaces for Neural Signal Users

Lilia Moshkina

Georgia Institute of Technology  
College of Computing

800 Atlantic Drive, Atlanta, GA 30332 USA  
lilia@cc.gatech.edu

Melody M. Moore

Georgia State University  
Computer Information Systems  
35 Broad Street, Atlanta, GA 30303 USA  
melody@gsu.edu

## ABSTRACT

In this paper, we describe the usage of *application context*, *user history* and *environmental context* in the environmental controls interface for *neural signal users* (people with complete paralysis operating computer-based equipment through *direct brain-computer interface* technologies). The goal is to minimize the required navigation in order to alleviate the problem of imprecise and error-prone nature of the neural signal controls.

## INTRODUCTION

Traditionally, people with complete paralysis, or "locked-in" syndrome (alert, intelligent but unable to move or talk) have been deprived of the ability to effectively communicate and control their environment. Our research group conducts research and development on *direct brain-machine interface* technologies to assist such people [2]. In order to communicate, the clients have two electrodes implanted directly in the motor cortex of the brain; the electrodes capture and process neural signals. These two signals are mapped onto unidirectional (right and down only) cursor movements, thus allowing the clients to use computer-based equipment. Due to the imprecise and error-prone nature of the captured neural signals, interaction with computers is an arduous and error-prone task for such people.

To minimize the amount of interaction/navigation required, we have considered applying intelligence to interfaces taking into account the following three features: **application context** (e.g., the most plausible option is displayed based on the current cursor position); **user history** (e.g., a sequence of functions is automatically performed based on past user behavior); and **environmental context** (e.g., the TV volume turns down if a knock on the door is detected). We have initially explored these issues in the context of an environmental controls application – software that allows neural signal patients to control a number of devices, such as lights, TV, VCR, etc. First, the current user interface design will be described. Second, examples for the three above-mentioned features will be presented, and finally, future research will be considered.

## BACKGROUND

Research in Brain-Computer Interfaces (BCI) is new, but fast-growing: there are over 20 active BCI research groups now compared to 6 in 1995 [3]. At the current time, however, the developed BCI technologies are painfully slow - they offer maximum information transfer rates of 5-25 bits/min at best [3], and the signals received are of low precision. Our approach to alleviating the consequences of this problem is to minimize the amount of interaction required by the user to perform everyday tasks (e.g., turning the lights on) both by designing *minimal interfaces*, or interfaces that require minimum navigation, and applying intelligence techniques to them.

## INTERFACE DESCRIPTION

In our design we took into consideration the following constraints imposed by the users' limited capabilities: unidirectional cursor movement: only right and down; difficulty in changing the direction of the cursor movement from horizontal to vertical and vice versa (this resulted in flattened hierarchy); and high perceived error-rate of navigation (up to 75%, as reported by a neural signal user). The current design follows the menu metaphor (Figure 1): the menu-bar (visible at all times) across the top of the screen contains a list of appliances the user can interact with, and pull-down menus below each appliance contain a list of corresponding terminal functions (visible only as the cursor moves over that appliance). The user selects a menu-item/function by holding the cursor ("dwelling") over the item for a certain period of time (e.g., 2 sec) which results in the performance of the appropriate task (e.g., the lights will be turned on if 'All Lights ON' task is selected). To overcome the limitation of

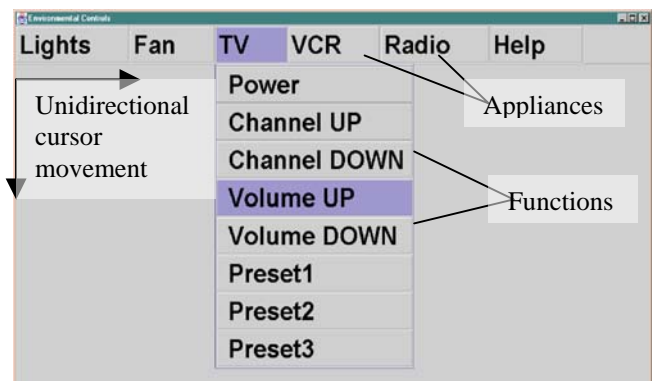


Figure 1: Environmental Controls Interface

unidirectionality, we designed the cursor to wrap around (return to the top left menu item of the menu) at the end of both the menu-bar and pull-down menus, as well as once a task is selected [2]. To facilitate error recovery in navigation, we modified cursor behavior in navigation of pull-down menus: moving the cursor to the right during the navigation will initiate the wrap-around behavior, thus offering the user an alternative way to return to the top left position if a navigation error occurs.

### **APPLYING INTELLIGENCE**

Although we designed the interface with the goal of minimizing the required screen navigation, applying intelligence to it offers new ways to reduce the amount of navigation. In particular, we have looked at the following areas: 1) automatic execution of a (sequence of ) step(s) - no navigation required; 2) automatic positioning of the cursor on the next most likely menu-item; 3) disabling options based on the current environmental state. We have also considered automatically changing options in the menus (*e.g.*, changing the order of options based on the frequency of use) and making the cursor skip over the disabled options. However, there is a trade-off between minimizing the navigation, and predictability and stability of the interface over time: if different options are displayed at different times, the interface becomes unpredictable and unstable. This could also inhibit pattern-formation in navigation: *e.g.*, the user could mentally encode that 2 options across and 1 option down is equivalent to turning TV on, which wouldn't be true if the options changed. The following features can be applied to all of the aforementioned areas, and are frequently combined to achieve the most effective result.

#### **Application Context**

Application context refers to the current state of the application, to include the context types of activity and time [1]. Application context can be primarily applied in the automatic positioning of the cursor on the next most likely option. Its usage is often combined with the results of user history, although it is possible to determine rules that depend on past user behavior to a lesser degree. An example of utilizing application context would be positioning the cursor on the 'TV Power' option right after 'Watching TV' option in the 'Lights' menu is selected, or moving the cursor to the help option if the 'Play' option on the 'VCR' menu is disabled (*e.g.*, if the tape is not in).

#### **User History**

User history encompasses capturing user behavior, discovering the patterns in action sequences and applying the results to the automation of actions or positioning of the cursor. Consider the following example of applying user history to automating action execution. When 'TV Power' option is selected, the TV turns on with the channel the user most likely would like to watch at this particular moment. In order to implement this particular scenario,

information on channel selection is captured in a database in day/time/channel triples (in half-hour intervals). To determine the most likely channel, we propose a version of reinforcement learning:  $\text{ChannelChoice} = R * \text{MAX}(\text{Quantity per day/time})$ , where R is a reinforcement constant; value of R for each channel/day/time triple is diminished if the user switched to a different channel within a certain amount of time, and is increased if not; a temporal degradation technique is used to put more emphasis on recent entries. A similar algorithm could be used to change the presets for TV channels displayed under the 'TV' menu depending on day/time and past user behavior, as well as for controlling volume for the TV, VCR and radio, and other options.

#### **Environmental Context**

Environmental context refers to the current state of the appliances and to certain people-related events (*e.g.*, knock on the door). It differs from application context in the source of the context: external *vs.* application-specific. It can be used to both automate certain actions and to disable currently unavailable options, with or without relying on user history. For example, if a knock on the door, or someone's entering the room is detected, the system can automatically turn the volume down (if TV or radio are on) until the person leaves. Similarly, as a response to the same event, a blackout screen can be automatically displayed to protect the user's privacy.

#### **CURRENT STATE AND RESEARCH AGENDA**

We have implemented our environmental controls interface in Java Swing and connected it to the commercial product Slink-e by Nirvis, Inc. through which it is capable of sending commands to IR and X10 controllable devices.

We are currently developing a number of other applications for neural signal users, such as a Virtual Keyboard, Web Browser, E-mail, and a number of training and biofeedback applications, and are exploring integrating the applications into a single system, which would leverage from application and environmental context, and user history between a number of applications. Finally, we consider applying application context, environmental context and user history to a wider domain: other limited-input settings such as cell phones, PDAs and other mobile devices.

#### **REFERENCES**

1. Dey, A.K. and Abowd, G.D. "Towards a Better Understanding of Context and Context-Awareness", in CHI'00 Workshop on The What, Who, Where, When, Why and How of Context-Awareness
2. Moore, M.M. and Kennedy, P.R. "Human Factors Issues in the Neural Signals Direct Brain-Computer Interface", in Proc. of ASSETS'00, 114-120
3. Wolpaw, J.R. et al. "Brain-Computer Interface Technology: A Review of the First International Meeting", IEEE Trans of Rehabilitation Engineering, 8(2), 6/00, 164-17