Assistive Technology
C. Sik-Lányi et al. (Eds.)
IOS Press, 2015

doi:10.3233/978-1-61499-566-1-586

Sweet Cheese Back to the Physical World

Jerome DUPIRE[a,1]
[a] *CEDRIC, Conservatoire National des Arts et Métiers, Paris , France*

Abstract. *Sweet Cheese* (switches) is a physical device that brings back the virtual keyboards to the physical world. Although the virtual keyboards are useful for people with motor impairment, they require to be used in a windowed environment. This is sometimes impossible due to the imposed full screen mode. We present inthis paper our solution that mixes advantages of both physical and virtual keyboards.

Keywords. virtual keyboard, physical computing, interaction, video game, accessibility

1. Introduction

Computers accessibility for people with motor disabilities can go through multiple solutions. One of these adaptations is the virtual keyboard (or software keyboard). By transposing the physical keyboard to a screen representation, it allows the user to use suitable hardware devices to achieve tasks usually performed through the conventional keyboard inputs.

A particular variation of this keyboard is the scrolling virtual keyboard. In the case of a heavy motor disability, in which a person would only have the use of a limited motor ability (to control a single switch, for instance), scrolling keyboard is particularly efficient.

However, this kind of adaptation is only possible if the working environment is compliant with the windowed mode. Only this condition makes it possible to simultaneously display the virtual keyboard and the application aimed with the text input. Moreover, displaying a virtual keyboard is always done at the expense of the available display area (Fig.1).

The purpose of this work is to go back from the virtual world to the real (physical) one, by taking advantages from both sides and designing a usable, adaptable control device to enhance digital accessibility.

[1] *Corresponding Author, CEDRIC, Conservatoire National des Arts et Métiers, 292, rue St Martin 75003 Paris , France; E-mail: dupire@cnam.fr*

2. Related Work

The "virtual keyboard" expression seems to define any mean (hardware, software or mixed) which allows the user to operate a computer or computer based device without a classical keyboard. A review of some virtual keyboards, according to this definition, can be found in [1]. Some mixed configuration attempts seems to be promising, as described in [2]. The authors propose a multimodal input system (voice, electromyography,) to control a pointer over a simplified software keyboard.

Today, almost every operating system brings its own (software) virtual keyboard: On Screen Keyboard (Windows), onboard (Ubuntu), etc. Individual initiatives have developed some products as well (cClick 'n type, Clavicom, etc.).

Software virtual keyboards have also generated a lot of research effort since their first appearances. Vella et al. [1] for example, have studied the adaptation of Fitts's Law to software virtual keyboards for people with motor disabilities. Isokoski [2] studied the advantages of adding contextual menus around these virtual keyboards. Malsattar et al. [3] tested a script virtual keyboard for Indic communication and Ghosh et al. [4] studied how to design an efficient virtual keyboard for Bengali writing. Rick [5] try to optimize software virtual keyboards in a touch-based interaction context. More recently, the emergence and development of mobile devices (phones and tablets) has launched a new field of application and research. Hoggan et al. [6], for example, studied the value of haptic feedback in the use of a virtual keyboard on mobile devices. Al Faraj et al. [7] worked on a target resizing feature, depending of the proximity of the stylus with a key.

In the specific context of motor impairment, software virtual keyboards are one of the more suitable solution for text input. Since the keyboard is software based, people are able to use them under very different types of hardware configurations, depending on their habits and/or needs. For example, some research focuses on the control of such software virtual keyboards through a Brain Computer Interface [9]. The authors tried to improve the spelling rate, which is the major drawback of this architecture (only a few letters by minute). From this point of view, software virtual keyboards offer better performances.

Human eye tracking has been tested as well, as described in Kumar et. al [10]. Their system allows the user to select a letter from an on-screen keyboard by using eye movements. Doyle et. al [11] have explored an electro-ocular based interaction to control the mouse pointer on the computer screen. Facial movements can be used as well, as described in Tu et. al [12].

All these works imply that the user is able to control a pointer, in order to select the keys on the virtual keyboard. This is not always possible and, in this case, the scrolling software virtual keyboard is a relevant proposal.

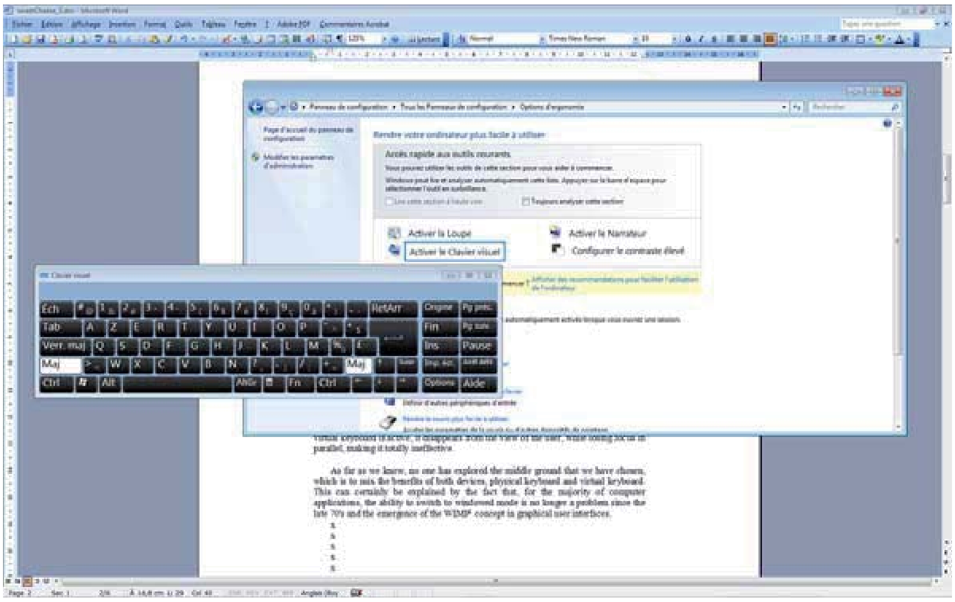

Fig.1 - A software virtual keyboard (Microsoft Windows)

Moreover, in the context of video games, applications are usually launched in full screen mode, without almost any chance for the gamer to change this. This is true on PC platforms and even more on consoles, which usually don't comply with the WIMP (*Windows Icons Menus Pointer*) paradigm.

This full screen mode prevents the use of software virtual keyboard solutions. Even if the software virtual keyboard is active, it disappears from the view of the user, while losing it's active status, making it totally ineffective.

As far as we know, no one has explored the area that we have chosen, which is to mix the benefits of both devices, the physical keyboard and the scrolling software virtual keyboard. This can certainly be explained by the fact that, for the majority of computer applications, the ability to switch to windowed mode is no longer a problem since the late 70's and the emergence of the WIMP concept in graphical user interfaces.

3. Hardware

Sweet Cheese (switches) is a physical device that brings back the software virtual keyboards to the physical world. Our device combines the advantages of scrolling keyboard with an expected hardware modularity, to remain adaptable to different contexts and different users.

Indeed, one of the major advantage of a virtual keyboard is to be widely adaptable. The user (even sometimes the system itself) is able to choose the best configuration of letters, rows and columns, targets' size and so on. A review of existing attempts for designing a efficient virtual keyboard can be found in [10]

3.1. Design

The design is based on the idea of modularity and adaptability. The basic building block of our system is a 3D printed cube. Each cube can be associated either with a key-strike (acting thus like a standard keyboard), or any combination of keys (like macros, shortcuts or combos).

The cubes can be connected together to create as many rows and columns as necessary. The user is free to arrange the various pieces to optimize the range and accessibility of keys (i.e. the features) that it wants to use. To do this, all cubes have two male and two female micro USB 3.0 connectors. The advantage of this device is that it enables the design of matrices whose columns and/or lines are not homogeneous (i.e. of different lengths). The whole device may be attached by a system of magnets, to a base placed on a table or clipped on the screen.

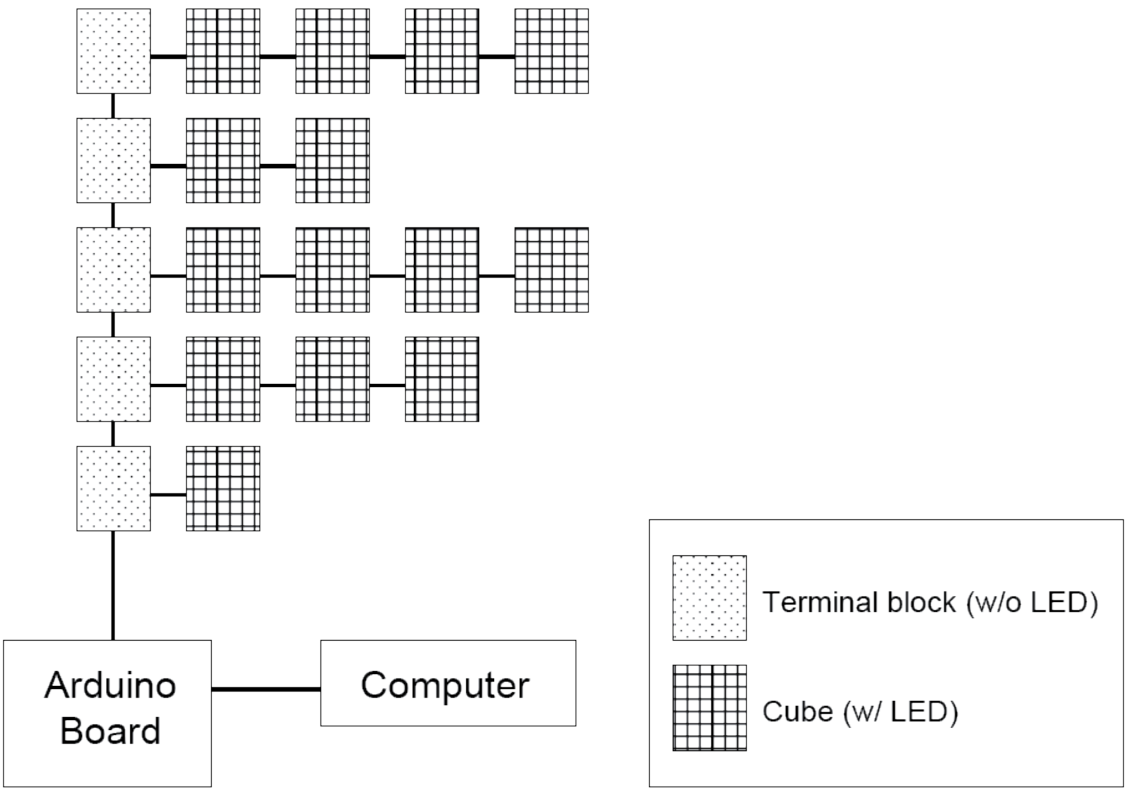

Fig.2 - Overview of the architecture

There are two types of cubes. First, the generic cube, which composes the rows and the columns of the physical matrix of symbols. The front face of generic cubes is provided for receiving a symbolic mask that is illuminated by a single LED, placed inside of it. This mask could be made from a single, hand written, sheet of paper or from a 3D printing.

On the other hand, the peripheral cube (*terminal block* on Fig.2) provides an interface between the cubes matrix and the electronics module. They are identified by a specific, non-movable, front face printing.

A major benefit of the hardware modularity is the ability of weighting the desired actions. Indeed, since the top row is illuminated first, on every cycle, the most used actions should be placed on it. This works for the ordering of the cubes on a given row as well. Finally, the first cubes have a greater priority than the last ones. Our device is thus able to manage a two dimensions priority system.

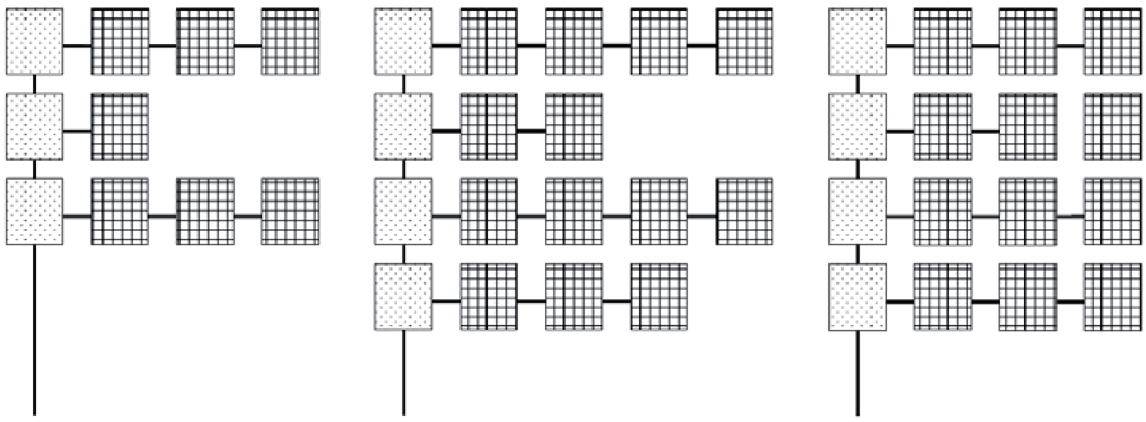

Fig. 3 - Three possible configurations

3.2. Electronics

The entire device is built around an Arduino Leonardo. The choice of this model is based on the embedded microcontroller, the ATmega32u4, which has built-in USB communication, eliminating the need for a secondary processor. It is recognized as a HID[1] device by operating systems. When connected, it is directly considered a standard keyboard and can send signals identical to the striking classic keys.

The entire matrix of LED is controlled by a set of shift registers (74HC595 type). This classic component allows to individually manage the switching on and off of each LED of the device.

[1] Human Interface Device

3.3. Software

Illuminating the matrix' rows and individual cubes is done in a usual (scrolling virtual keyboard) manner: full rows are illuminated one after the other, from the top downwards. The user interacts with the device of his/her choice (single button, head switch, straw, etc.) based on its preferred installation.

When the first interaction is performed by the user (button click), the current (i.e. illuminated) line is selected and the cubes of this line are then illuminated one after the other, from the left to the right. A second interaction causes the selection of the current cube. Then the embedded software transmits to the PC the key or combination of keys (i.e. the macro) mapped to the selected cube, simulating the presence and use of a standard keyboard.

4. Experiment

Preliminary tests were performed on a few volunteers and were encouraging (healthy users). The task was simple: it consisted in "typing" a given short text onto a digital document, using our device. Users were free to choose and build the configuration of the device (i.e. numbers of rows and columns). Only one hand switch was used by the to control the system.

All the participants were able to input the whole text. Interviews were individually conducted with the participants and we recorded the following feedbacks: We were asked to give a control on the luminosity of the cubes' LEDs. The keyboard assembly was not so easy and some of the participants asked for help.

Since our device is dedicated to motor impaired people, one other drawback is that a extra person is needed for the hardware assembly.

5. Future Work

It is important to allow each user to be able to adapt the device settings to optimize comfort and efficiency. A dedicated interface for changing the frame rate or mapping cubes with a simple or composed command should be added.

An other idea is to replace the current, paper made, masking system by a dynamic, electronic one. Tiny OLED screens are the current best candidate. They will allow dynamic and fast (re)configuration and deserve to be tested.

To go further in the intuitive nature of the object, we plan to study the possibility of transposing this cube-control association memory, from the microcontroller embedded software, to each cube's new memory. We think this feature would bring users together, by not being considered a technical aid for people with disabilities but as a device capable of rendering services to very different people, disabled or not.

Finally, we would confront our device to various game situations, in order to test the reliability and efficiency in most situations. Extensive tests have to be conducted on games with different time constraints, from the most asynchronous case to the opposite extreme: First Person Shooters (FPS), Real Time Stategy games (RTS), Action games, Turn-based games, etc.

References

[1] Mathias Kölsch, Matthew Turk. Keyboards without Keyboards: A Survey of Virtual Keyboards, In Proceedings of Sensing and Input for Media-centric Systems, 2002
[2] Vella F., Vigouroux N., Gorce P. Fitts's law interpretation as ergonomic metrics for motor impaired users, Computer Methods in Biomechanics and Biomedical Engineering, Taylor & Francis Group, Vol. 12 (2009), 249-250.
[3] Poika Isokoski. 2004. Performance of menu-augmented soft keyboards. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04). ACM, New York, NY, USA, 423-430.
[4] Nirav Malsattar, Nagraj Emmadi, and Manjiri Joshi. 2014. Testing the efficacy of an Indic script virtual keyboard: Swarachakra. In Proceedings of the India HCI 2014 Conference on Human Computer Interaction (IndiaHCI '14). ACM, New York, NY, USA, , 160-166.
[5] Soumalya Ghosh, Sayan Sarcar, and Debasis Samanta. 2011. Designing an efficient virtual keyboard for text composition in Bengali. In Proceedings of the 3rd International Conference on Human Computer Interaction (IndiaHCI '11). ACM, New York, NY, USA, 84-87.
[6] Jochen Rick. Performance optimizations of virtual keyboards for stroke-based text entry on a touch-based tabletop. In Proceedings of the 23nd annual ACM symposium on User interface software and technology (UIST '10). ACM, New York, NY, USA, 77-86.
[7] Eve Hoggan, Stephen A. Brewster, and Jody Johnston. 2008. Investigating the effectiveness of tactile feedback for mobile touchscreens. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08). ACM, New York, NY, USA, 1573-1582.
[8] Khaldoun Al Faraj, Nadine Vigouroux, and Mustapha Mojahid. 2009. SmartKey: a multi purpose target expansion based virtual keyboard. In Proceedings of the 11th international ACM SIGACCESS conference on Computers and accessibility (Assets '09). ACM, New York, NY, USA, 251-252.
[9] R. Scherer, G.R. Muller, C. Neuper, B. Graimann, G. Pfurtscheller. An asynchronously controlled EEG-based virtual keyboard: improvement of the spelling rate. IEEE Biomedical Engineering, 979 - 984, 2004
[10] Kumar, M., Paepcke, A., Winograd, T. EyePoint: Practical pointing and selection using gaze and keyboard. In Proc. CHI'07, ACM Press (2007), pp. 421–430.
[11] Doyle, T. E., Kucerovsky, Z., Greason, W. D. Design of an Electroocular Computing Interface. In Proc. CCECE '06, (2006).
[12] Tu, J., Huang, T. and Hai Tao, Face as Mouse Through Visual Face Tracking, In CVIU Special Issue on HCI 2006.
[13] Jochen Rick. 2010. Performance optimizations of virtual keyboards for stroke-based text entry on a touch-based tabletop. In Proceedings of the 23nd annual ACM symposium on User interface software and technology (UIST '10). ACM, New York, NY, USA, 77-86.