Design and implementation of a virtual keyboard to aid handicapped people to use computers

Julio A. Pérez Garcia
Rafael Reilova Barletta
Graduation candidates in electrical engineering, UPPR

Abstract

This article describes our Capstone Design Project. The project requirements were to design and implement an interface software, to create a virtual keyboard on screen, optimized for typing Spanish language, controlled by simple input devices to allow people with severe movement handicaps to use a computer keyboard.

Sinopsis

Este artículo describe nuestro proyecto de diseño de integración. Los requisitos del proyecto son los de diseñar e implantar un programa de enlace para crear un teclado virtual, optimizado en español, controlado por interruptores sencillos que les pemiten a personas con impedimentos físicos operar una computora.

Introduction

To introduce the concept of a virtual keyboard and its peripherals, we must understand who is our client. Our project is targeted to users with severe physical limitations.The condition that causes these limitations is quadriplegic body handicap, which impacts the victim mentally and emotionally. In our interviews with victims of this condition we saw the emotions, feelings and the frustration of being ignored because of this condition, and being enslaved to a wheelchair with limited movement. We also observed the potential of these persons to be productive to our society. It is important to give them a chance of more freedom in their lives, to overcome another physical or architectural barrier for communication. This virtual keyboard will provide the handicapped the capacity to type and interact with almost any software available in the computer industry. One of the objectives is to provide the quadriplegic control over their environment using electric devices attached to the computer where the software is located. The keyboard is optimized for typing in Spanish, but can be used in other languages.

Design objectives.

To develop this software for a personal computer screen, we considered a user with very limited mechanical movement:

- The device must understand, respond, correct and manage command inputs in Spanish. To help the intended user the system must store the most common words in a data base, thus reducing user input to a minimum and adjusting to the user's writing style and capabilities.
- The total cost of this project for the user should not exceed $250.00.
- The project must be finished by the end of the Capstone Design Course (within the six months time limit).
- The operating system program for this project must be compatible with most existing applications.
- The user's hardware should not require any modifications.

Possible solutions

We evaluated various possible solutions:

1. Using a PLC (Programmable Logic Controller) to handle our virtual keyboard. This alternative establishes total mechanical and environment control. The PLC can support the standard platform and the parameters for screen control and management. This

alternative is costly and we have a time constraint in providing the physical device itself, including its customized service parts and reconfiguring the system for our purposes.

2. Using two personal computers one to support the virtual keyboard software, providing input management, screen scan option, environmental control and text on screen and the other to support the user's application. The cost of having two machines is a critical constraint factor for our project.

3. Writing software to perform the virtual keyboard function in the same computer where the application software is running. This alternative is more cost effective in terms of hardware. It also should be easier to set up and use. Its only disadvantage is the overhead that the new software will add to the system, although this is not critical since the intended user will not notice the reduced speed.

The development team, after investigating standard software platforms and verifying the hardware required to suit our software, selected the last alternative. Even if we need to work with an overhead for this equipment to establish the necessary time to work with this choice, but an important factor for this choice is to keep the cost of the system as low as possible, centralizing the hardware into one machine.

System description.

The virtual keyboard module presents a user friendly interface which allows for alphanumerical input (keyboard) to other Windows applications. Additionally, the keyboard represents the application front-end, and therefore must allow for direct configuration and customization by the user.

The main input of the module is the standard Microsoft Windows mouse emulation. Whether this is a real mouse or the output of a software driver for some custom hardware is irrelevant. This includes the output for the alternate input processor module except when in scan mode. In scan mode it is more effective to read the switch directly and implement the scanning locally on the virtual keyboard window.

Keyboard events are decoded from the mouse events on the display window or by the local scanning algorithm when in scan mode. When in scan mode the module will switch the alternate input process into raw mode and will process the switch events locally. A provision is made to allow the switch back on the alternate input processor when the user wishes to select an object or application outside the virtual keyboard window.

This application is less expensive in terms of hardware. It supports our standard software platform and establishes a software link in terms of application to support other applications. We use the screen for environment control. Then we must produce the necessary software algorithms to compile the following required tasks:

- Virtual keyboard presentation and layout.
- Input device management
- Windows scanning
- Vocabulary management.
- Providing the capability of output code to drive logical controller devices.

In addition to the keyboard, the Windows environment is tightly dependent on the use of a pointing device. Almost all functions of the environment and applications can be controlled by the mouse, except those actions that require explicit keyboard input. We intend to exploit this dependency to our advantage by simulating first a mouse, whenever possible, from the special user input device. Then, we will build a keyboard interface based on this mouse input, thus maintaining modular structure.

If the software package is to allow the handicapped user an almost full control of the computer, as if he or she were using a standard keyboard and mouse, then we must provide a way to intercept system calls from applications and the windows manager, redirecting them to our software. From the above model, it is clear that we need to build a module capable of converting the special input devices to standard Windows mouse events. We call this module the alternate input processor. It is important to note here that commercially available input devices for the handicapped usually provide part of this module with their equipment, either as built-in hardware emulation or software support. The possibility of simplifications or complications due to this fact are discussed in detail on the description of this module.

In addition, the visual part of the application package presents the user with a virtual keyboard drawn on-screen and by different methods, depending on the input device, allows him to enter keyboard input to other applications. This part of the software is called the virtual keyboard display module. This module reads the standard mouse input if possible, and in this mode will not require any low-level hardware functions. The alternate input processor module would provide the converted mouse input from the user input hardware.

One of the characteristics that distinguishes our project from similar systems commercially available is the ability to complete words, thereby reducing the necessary time to input them in Spanish. The problem of word completion is best solved by designing a logical module to perform this task. We use the term logical because this module is integrated into the finished executable of our application. This software piece will be called the word completion module.

User interface description

We have identified three input modes depending on the type of input device available to the user. With a single digital input device or switch, the method of input is through scanning. When analog input is available in the form of two position coordinates the method is termed dwelling. With two analog and one digital inputs, the operation is exactly the same as standard mouse, in other words, point and click.

Scanning

In scanning mode an array of options is displayed on-screen, organized by rows and columns. First, the rows are scanned from top to bottom by highlighting each one in succession. The user presses the switch when the row containing his choice is highlighted. Then, the columns of this particular row are scanned until the user's intended choice is highlighted. When this occurs the user signals the application by pressing the switch again. The user's choice is now fully determined. Note that the user's choice could be icons representing keys on a virtual keyboard or could be screen pixels. The later allow the user to specify in general any mouse event with a single switch. A way to allow for repeat clicking the same position if so desired is implemented by an auto-repeat option. The optimal position of virtual keys and other icons is considered to minimize the amount of wait time for the scanning to proceed.

Dwelling

In dwelling mode the user is capable of specifying a position coordinate on-screen by using an analog input device, but cannot signal the mouse click event. To overcome this difficulty, we use a timer which times-out after the user has left the pointing device almost stationary on the same screen position. The time-out signals a mouse click on the current position coordinates. An averaging algorithm is useful to compensate for erratic motion and to avoid the possibility of erroneously repeat clicking the same position.

Point-and-click

This is the standard operation mode of a Windows mouse. No additional conversion is required except an allowance for erratic motion. The same averaging algorithm used for dwelling may be used in this mode.

Word completion module

The word completion module uses a word completion algorithm for Spanish words, using probabilistic techniques, including the knowledge of the most common character sequences and a user profile of previous word usage. The module maintains a simple word dictionary with the additional information required for the probabilistic word completion and a stream of completed words entered by the user which were not in the offered choices. When the user's next entered word can be predicted to less than a given number of choices, the module sends a data structure containing the proposed compilations. The module should adapt to a given user and maintain a database to aid in predicting future word compilations. A lot of research and testing was required in order to fine-tune this module, which is capable of successfully completing words most of the time.

The option to use a commercial database program or a small special database written in C language will depend on other design constraints such as overall cost of the system and product delivery time. The possibility to interfacing with FoxPro or Access has already been verified, but we still have to consider whether the user can tolerate the burden of the overhead of the program. Only after a preliminary development and testing stage will we be able to answer this problem.

Alternate input processor

The alternate input processor is responsible for reading devices not supported by the standard Microsoft Windows GUI (Graphical User Interface) but required by our design, and translates the input read to mouse movements and click events.

Devices support/input requirement

The module can run independently of the main application, but also accepts direct configuration commands from the virtual keyboard module. The following devices require preprocessing by our module before forwarding to the windows environment.

switch - A simple single-pole switch connected to any available serial port of the PC.

mouse (without button input) - This a mouse compatible serial device without the button function. Preprocessing is required to implement the mouse-click event with the dwell function.

output: - Sends the current active application or the windows display manager the translated / preprocessed mouse events. It also has the capability of transmitting raw device events to our main module via an interpose link.

Control specification - The virtual keyboard module must be capable of configuring the alternate input processor. We foresee a set of commands to be sent by a dedicated interpose link (OLE or DDL).

Project software development solutions.

For this project we visualized three possible solutions for our concept, our criterion is critically based within the constraints of the budget, time and easy tool access for the successful accomplishment of this project. We used Microsoft Windows environment for the project because this operating system is the standard in the computer industry.

We used Visual Basic for the object oriented programming. This program provided the capacity to integrate algorithms from other programs using *.obj files. As a result we can integrate some specific programming to our function keys using other programming software. This integration provides for a successful software interface with commercial applications so the user will not be limited to this application only. Moreover, we intended to establish direct access to the parallel port to be decoded in an external logic box to control external appliances and electrical instruments. Within this program we intended to create the whole virtual application and we included a CC compiler which provides very powerful software tools for applied software design and development. The programming will help us to develop the necessary algorithms for Spanish language prediction, to windows-coordinate scanning in click action input and to create the data base required to store the most common words used. With this mechanism we achieved intelligent word recognition to complete words using the first three or four characters written in the virtual keyboard.

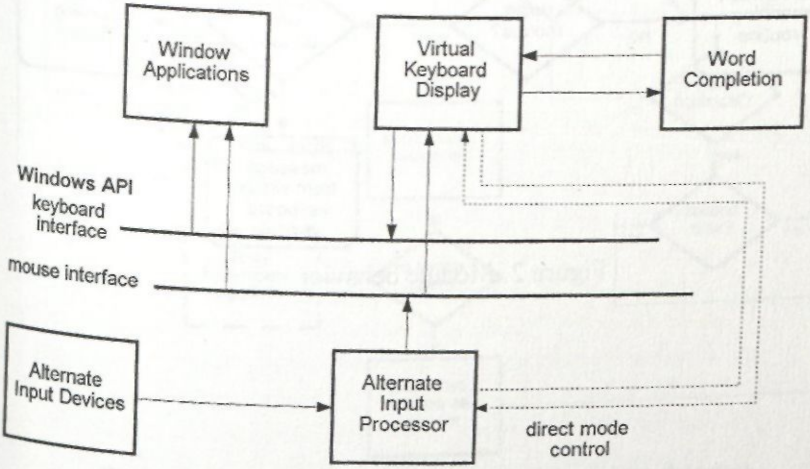

Figure 1. Word completion module

Figure 2 shows a flowchart of the behaviour of the word completion module.

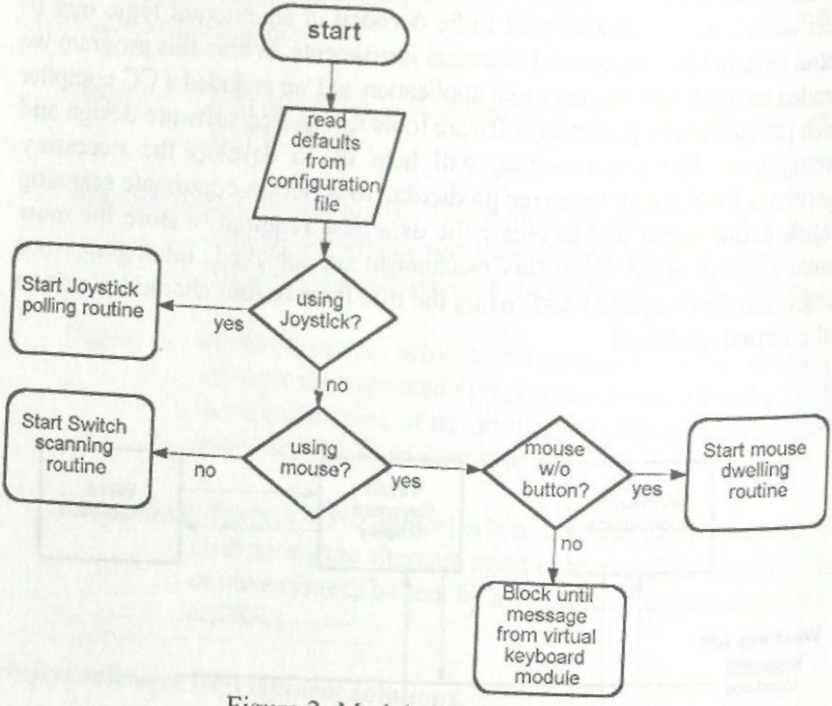
start
read defaults from configuration file
using Joystick?
yes
Start Joystick polling routine
no
using mouse?
no
Start Switch scanning routine
yes
mouse w/o button?
yes
Start mouse dwelling routine
no
Block until message from virtual keyboard module

Figure 2. Module behavior

Figure 3 shows the scanning and virtual keyboard capture diagram.

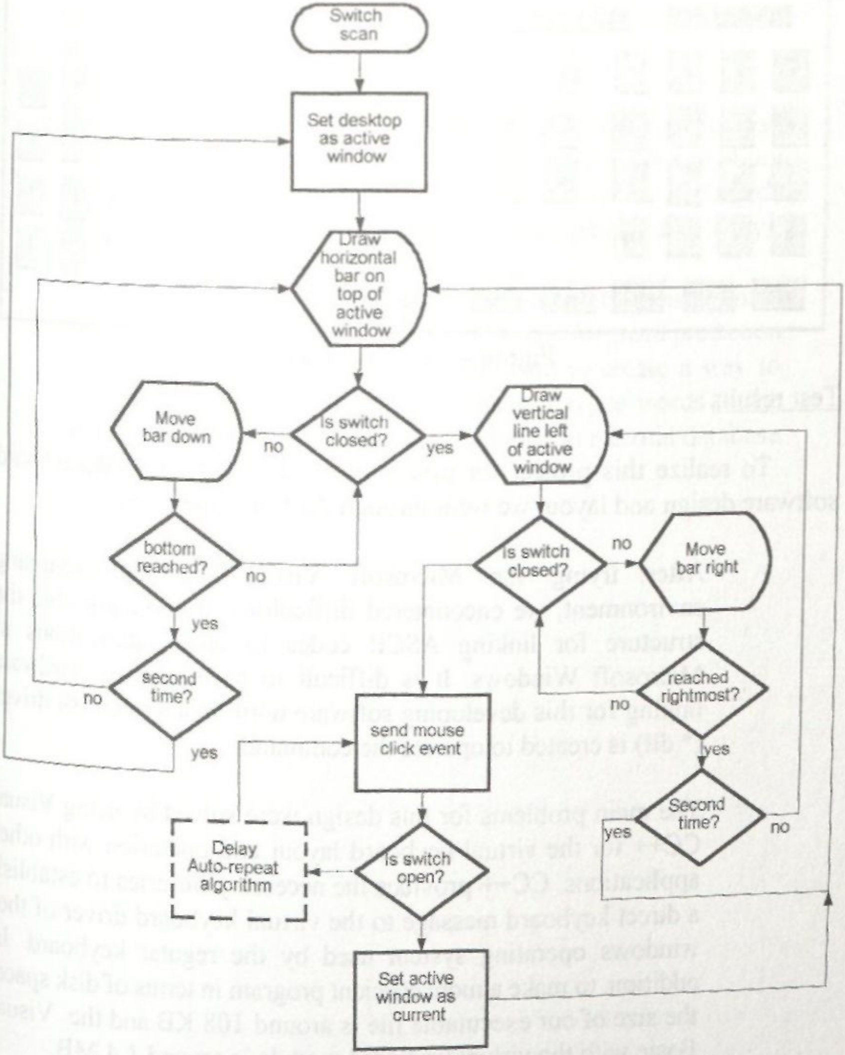

Figure 3. Scanning and virtual keyboard capture diagram

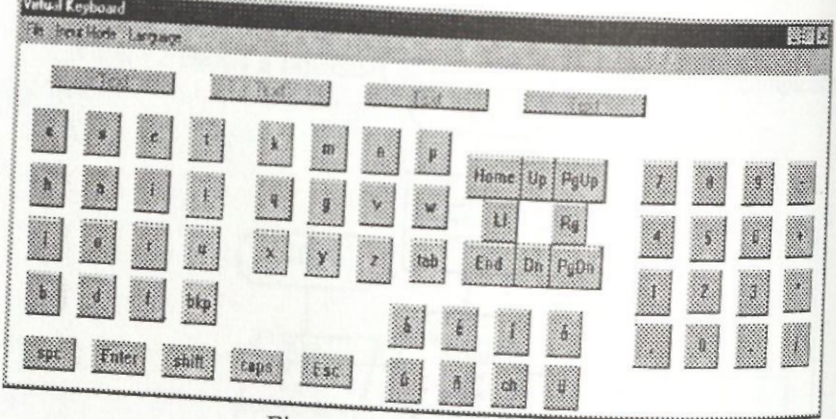

Figure 4. Keyboard layout

Test results

To realize this project for proper operation of the virtual keyboard software design and layout we went through the following steps:

- After trying the Microsoft Visual Basic programming environment, we encountered difficulties to make possible the structure for linking ASCII codes to other applications in Microsoft Windows. It is difficult to establish any hardware linking for this developing software until another control driver (*.dll) is created to operate the command.

- The main problems for this design were solved by using Visual CC++ for the virtual keyboard layout and operation with other applications. CC++ provides the necessary libraries to establish a direct keyboard message to the virtual keyboard driver of the windows operating system used by the regular keyboard. In addition, to make a more efficient program in terms of disk space, the size of our executable file is around 108 KB and the Visual Basic with the virtual keyboard module is around 1.4 MB.

- We used the Borland CC++ to make possible the modules of active window scanning, dwelling and point and click. This development software provided the necessary tools to make possible the word completion module with integrated Spanish dictionary.

- We considered the thesis of Mauricio Lizama for Spanish keyboard layout design, a survey of five possible customers of our product to establish an average for the "user friendly keyboard" also considering the most used words in the Spanish dictionary.

- Using the C language, we created an algorithm to establish a small dictionary for Spanish language correction and for word prediction module. This feature helps the design team to create a way to make a learning application. When the user types words to the virtual keyboard these words are stored into an internal database so they can be used later on.