Implementation of a Computer Game Voice Command Board with a Speaker-dependent Recognition Chip

Lee Kyung Mog
E-Commerce Dept. Semyung University
Jecheon, ChungBuk, South Korea
ahkm@semyung.ac.kr

Abstract

A computer game voice command board with a speaker-dependent recognition chip was constructed as an auxiliary joystick pad for a space-war shooting game. Normally, the game was operated with a key board, but the voice board was used for special events such as multi-bullet shooting, jumping, etc. The command board consisted of an ATME microprocessor, a speaker-dependent voice recognition chip, and a Bluetooth communication chip. The communication protocol was framed with 'ready-to-receive' bits, or 'send-now' bits which were combined with the recognized word for synchronized communication. The communication between a PC’s COM port and the voice board was through the Bluetooth connection. The game was played by five voice commands. The recognizing time was measured approximately to be 100 milliseconds for each of the commands. Then, the curve fitting result of all the measured data showed that the recognizing time was increased about 2.5 milliseconds for each additionally added command. The game was tested with the voice aided controller, and its performances were convenient with the controller. However, more improvement in the voice recognition accuracy was required so that the voice controller can be used in speedy and multi-voice-command games.

Keywords: Voice game controller, Speaker-dependent voice recognition, Voice recognition keypad

1. Introduction

Nowadays, game industry has been rapidly developed around the world. Variety facilitating game controllers or game input devices has been suggested such as a steering wheel and pedals for driving games, a floor pad for RPGs[1], motion-based controllers[2], the EyeToy for PCs[3][4], etc. Other application, like the brain-computer interface (BCI), was published for the control of the game by the commands of the brain signal translated into machine commands[5][6].

For modern communication technology, the Bluetooth application is also quite popular[7]. Its communication distance is more than 100 meters long, and many kinds of Bluetooth communication devices are available on the market.

Some applications of speaker-independent voice recognition techniques to wheel chairs were published[8]. However, in those papers, speaker-independent voice recognition chips were used, and a digital board game with speech recognition system was published for children[9].

While playing computer games, sometimes, it is more convenient for gamers to speak commands than to use a key board or a joystick pad. In this paper, a space war game was programmed in C++ language, and the game was tested for five voice commands. All the images of the background, the crafts and the bullets of the game were prepared in 24-bit bmp graphics. When the game was played, all those images were loaded into an off-screen surface first and flipped to the destination surface memory by the DirectDraw functions of the DirectX9.0 software development tool kit(sdk). The DirectX sdk provides the DirectSound class for the game program to handle sound related functions, too.

Here, a voice control board using a voice recognition chip was designed to be used as the command pad to substitute a joystick via the Bluetooth communication. The chip was a speaker-
dependent voice recognition chip with the feature that only allows the owner to access the game. So, the chip could be used as a voice command device without building the database of the command words. Generally speaking, software recognition is much more convenient, but hardware recognition is still faster[10].

2. Body of Paper

After a space war shooting game had been programmed in the Visual C++ language, all images of the graphics, including the background, the crafts and the bullets of the game, were prepared in 24-bit bitmap graphics. However, the crafts’ images were created as 3D objects in 3d MAX, first. Then, only the required images were captured and manipulated as 24-bit bitmap images with the Photoshop graphic tool. Once the game was started, all of those prepared graphics were loaded into an off-screen surface, and drawn into the destination surface memory by the DirectDraw methods of the DirectX 9. To handle the graphics and the sound, the DirectX 9.0 SDK was needed to be included into the game program.

The voice commands for the game’s operation were inputted from an external speaker-dependent recognize board through the Bluetooth communication as a command pad operating like a joystick. The board included the Sensory’s voice recognition chip. The Sensory voice recognition chip can recognize a few spoken words regardless the user’s language nationality when it is used as a speaker-dependent mode. To be used as a speaker-dependent function, the chip should be trained with the speaker’s voice commands first. Then the trained words could be recognized when the same words were spoken into it.

In the communication between the voice chip and the PC, two communication protocols, a ‘ready-to-receive’ frame for a request signal and a ‘send-now’ frame for a response signal, were used. The ‘ready-to-receive’ frame was sent to the control board, and the board responded with the ‘send-now’ frame which was combined with the recognized word. During the game, the recognized voice commands were sent to the PC via the Bluetooth communication. The communication status of the PC port were set up in the 9600 bits/sec board rate, 8 bits data, 1 bit parity, 1 stop bit, and no hardware flow control.
Figure 1 shows an example of the system of the space war shooting computer game that was commanded from an external speaker-dependent voice recognition controller. The game operating PC had a Bluetooth device attached to COM-6 serial port, and the controller also had a Bluetooth module on itself for communication.

When the space game was started on the computer, the computer opened the COM-6 port of the PC for the Bluetooth communication. Then, it sent a ‘ready-to-receive’ signal, which was in the form of “01010010”, to the communication board by using the Bluetooth. If a voice command were uttered to the recognition chip, the chip began to compare the previously recorded pattern with the command one, and generated the recognized word. After the ‘ready-to-receive’ signal was received from the PC, as soon as the voice recognition chip was ready to send the recognized word to the ATMEL microprocessor, and then it sent the word immediately. Then the microprocessor generated the response protocol that consisted of ‘send-now’ frame bits, “01010011” and the recognized word together as a frame. After the PC had analyzed the received information, it retrieved the spoken word, and the space-craft game executed the command according to the word. Depending on the voice command, the game could control the space ship and shoot down enemy ships by voice commands.

In this experiment, the COM-6 port was used for the Bluetooth serial communication. The port was connected with a commercially available IVT-BlueSoleil Dongle as a USB-port, and the Dongle was paired with the Bluetooth module on the control board.

Figure 2 shows the game program flow chart. When the game was started, it sent the “Ready to receive” signal to the voice aided controller via the Bluetooth device, and checked the response from the controller. If there was no voice command, the game repeated above process. However when a voice command was received, the game checked the type of the command, and, depending on the order, performed its proper operation, such as jumping left, right, up, down, and shooting multiple bullets.

Figure 3 shows an external voice-recognizing controller board. A Bluetooth chip on the control board was used for the communication between the PC and the controller board. The board sent the protocol including the recognized word.

When a player uttered voice commands, the speaker-dependent voice recognize chip, which was
mounted on the controller board, recognized them. In this experiment, a Sensory Company’s voice VLSI chip was used to recognize the player’s commands. The chip worked in a following way; the user’s command was sampled, and converted into a digital signal by an Analog-to-Digital. After bandpass filters, the signal was normalized in time base. The player’s voices were prerecorded in a RAM, and they were compared with the spoken voice. Then, the ASCII code corresponding to the word on the best probability among all the commands was selected and sent to the ATMEL processor.

An ATMEL 89C2051 microprocessor was used to handle communication between the board and the PC. The program of the microprocessor was written in C language and downloaded into the processor via a serial port. In the experiment of the round trip time from the PC to the controller, it took less than 1 millisecond.

The measurement of the chip’s spoken word recognition time was as follows; the Sensory chip sent two kinds of marks, the beginning and the ending of the voice recognition. When the beginning mark was received in the microprocessor, the microprocessor sent a starting signal of the recognizing process to the PC where the system time was recorded. When the recognition process was finished, it sent the ending mark to the microprocessor, and the processor sent an ending signal to the PC which calculated the elapsed time between the starting and the ending signals.

Figure 4 shows the measured recognition times verse the number of prerecorded Korean words. All of the words were trained and prerecorded into the voice chip. When the board was received the starting order of the recognition process from the PC, as soon as the command was spoken, the board began to recognize the voice by returning the starting signal.

When the recognition was finished, the board sent the ending signal to the PC, and the PC calculated the elapsed time between the starting and the ending signals. The measured recognition time depended on the number of commands. It took less than 100 milliseconds for five commands, and 110 milliseconds for 10 commands.

By the curve fitting of the recognition time with those measured data, the equation of the measured time was shown as follows in the number of the commands,

$$t = 80\text{msec} + \frac{2.5\text{msec}}{\text{word}} \times W$$  \hspace{1cm} (1)

Here, $t$ is the recognition time, $W$ is the number of commands, and, all the units are in the millisecond. The minimum required time was about 80 milliseconds including the waiting time for the end of the command, the preparing time of the recognition process, and some delay time.

From the equation (1), the recognition time for each added word was found to be 2.5 milliseconds.
Figure 4. Measured recognition times verse the number of prerecorded Korean commands

Table 1 shows the measured error rate for five Korean commands. After all of the commands were trained into the voice chip, each command was spoken once for the correct recognition test and repeated 10 times. After the recognition test, the failed numbers were counted to calculate the error rate.

The formula used to calculate the recognizing error rate is shown in equation (2).

\[
\text{Recognizing error rate} = \left( \frac{\text{Failed number}}{\text{Total number}} \right) \times 100\%
\]

(2)

For a certain command, “Oreunzzok”, the error rate was measured to be 30%.

Here, the control board was constructed with a Bluetooth chip, an ATTEL 89C2051, and a Sensory Voice recognition Chip. The result of voice recognition was sent to the AT89C2051 as a word, and the ATTEL sent the responding protocol to the PC through the Bluetooth module. After the protocol having been interpreted as the game control signal, the game was played with the aid of the voice commands for special actions such as “Balsa” for shooting multiple bullets, or the word ‘Wizzok’ for jumping the craft into the top side, etc.

Figure 5 shows the space war shooting game programmed in C++ language. When a gamer said the shooting command, “Balsa”, multiple bullets were launched to the enemy craft. Using voice command was pretty convenient to an emergency situation instead of using a mouse or a keyboard. For the five commands, the program worked well, but the response time and the accuracy of the voice board should be improved more for some speedy games.

<table>
<thead>
<tr>
<th>Command</th>
<th>“Wenzzok”</th>
<th>“Oreunzzok”</th>
<th>“Wizzok”</th>
<th>“Arae”</th>
<th>“Balsa”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed number for 10 time tries</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Error rate for 10 time tries</td>
<td>20%</td>
<td>30%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>
3. Conclusion

The main purpose of this paper was to use a speaker-dependent voice game controller board as an auxiliary joystick pad for a space war shooting game. The controller board was constructed with an ATMEL processor, a speaker-dependent voice recognition chip and a Bluetooth communication module. A communication protocol was created, which included ‘ready-to-receive’ bits or ‘send-now’ bits to which the recognized word was combined. By the protocol, the communication between the PC and the board could be synchronized, and the information of the recognized word could be secured. The PC’s COM port and the control board were connected by the RS-232C cable for communication test. Voice commands during the game were sent through the Bluetooth communication. The measured times of the command recognition process depended on the number of the trained words. For five Korean commands, it took approximately 100 milliseconds, but for 35 commands, it took about 160 milliseconds. All of the game performances were satisfied with the five voice aided commands, but the recognizing error rate for a certain command was even 30% as the worst case. Still more improvement in the voice recognizing accuracy is required so that the voice controller can be used in speedy and multi voice-commands games.

4. References

Implementation of a Computer Game Voice Command Board with a Speaker-dependent Recognition Chip
Lee Kyung Mog


