The Implementation of Digital Pattern Playback on DSP

Kensuke Ohta, Takayuki Arai, Keiichi Yasu

Department of Electrical & Electronics Engineering, Sophia University
7-1, Kioi-cho, Chiyoda-ku, Tokyo 102-8854, Japan
ohta-k@sophia.ac.jp

ABSTRACT

This paper describes the implementation of a digital signal processing technique on DSP. This technique aims to reconstruct a speech from the visual representation of the speech. Techniques for inversion, know as the Pattern Playback problem, have been studied for the analysis, the transformation, and the resynthesis of sound. This project tried to play back a spectrogram of a line along the time axis, as a simple example. In this paper, the history of Pattern Playback is described first, and the principles of the method to be implemented on DSP are explained next, and details of the implementation are discussed. Finally, the result of this project and the work to be done in the future are considered.

1 INTRODUCTION

In the field of acoustics, visual representations of sound are often used to see the sound information. One of the most popular and useful ways to visualize sound is sound spectrograms. To validate the effects of modifications in the spectrogram, or to investigate the effects of those modifications in the sound as heard, Franklin S. Cooper and Haskins Laboratories researchers developed pattern playback in 1950s [1]. Figure 1 shows the operating principle of the pattern playback. The light generated by the light source is modulated by a tone wheel. After the light is collected by the lens, the mirror makes it a line of light that goes through the film, on which the spectrogram is drawn. The light is collected by an optical system, and led to a phototube. The amplified photocurrent from the phototube is fed into a loudspeaker, and finally produces the sound. Such pattern playback has greatly contributed to the development of speech science. The ana-

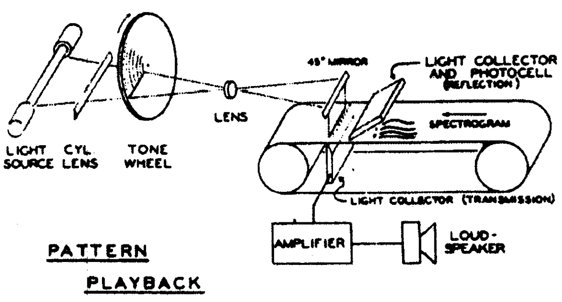

Figure 1: Operating principle of the pattern playback [1]

log version of the pattern playback invented by Cooper et al. was digitized by Nye et al. as a new research tool for the analysis, the manipulation, and the resynthesis of speech data [2]. The digital version of pattern playback, called digital pattern playback (DPP), was developed with PDP-11/45 manufactured by Digital Equipment Corporation (DEC); PDP-11/45 uses four disc drives, 96kB of core memory, GT40 display system, and other peripherals [2]. The DPP of Nye et al. was a large computer system using the software and hardware technologies at that time; thus a new DPP system was expected to be developed with more modern technologies [2]. New DPP algorithms using modern digital technologies: the AM method (based on amplitude modulation) and the FFT method (based on fast Fourier transform), were proposed by Arai et al.

[3, 4]. As the FFT method is thought to be more appropriate for real-time DPP, the FFT method was chosen for this project. Details of the FFT method will be described in the next section. The importance of the real-time aspect, especially in a pedagogical situation, was pointed out by Arai et al. [3, 4]. This paper describes the implementation of the FFT method on DSP. The TMS320DM642 DSP evaluation board, manufactured by Texas Instruments (TI), was used for the development of this project. Also, this paper proposes several future forms of DPP, and discusses them considering the possibility of real-time DPP.

2 PRINCIPLES

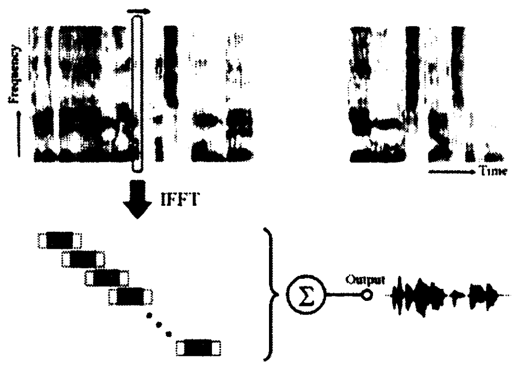

Figure 2: The concept of the FFT method [3, 4]

Figure 2 illustrates the concept of the FFT method [3, 4]. In this algorithm, the time axis is divided into time slices, called frames. And, the spectral envelopes are obtained by the reduction of the spectral resolution of these frames on the frequency axis. Here, several ways are available to obtain the spectral envelopes, and this project used an easy way, averaging each set of contiguous values on the frequency axis, considering the simplicity of the algorithm. The spectral envelopes are next converted back into the time domain by taking the inverse FFT (IFFT). From the IFFTs of each spectral envelope, the impulse responses of each frame are obtained. The main parts of these impulse responses are combined into a sequence, which produces the sound to be output. Because the aim of the FFT method is a simple reconstruction of the speech, the pitch is not changed during the playback [3, 4]. Also, the phase components are set to zero.

3 IMPLEMENTATION

Figure 3 shows the environment of the system in this project. The digital image of the spectrogram in the bitmap format is stored in the host computer. And the DSP board, TMS320DM642 evaluation board, is connected to the host computer through the XDS510 USB JTAG emulator, which is manufactured by Spectrum Digital. And the USB JTAG emulator is connected to the host computer via a USB2.0 cable. Also, the loudspeaker is connected to the DSP board, so that the sound can be output. This system

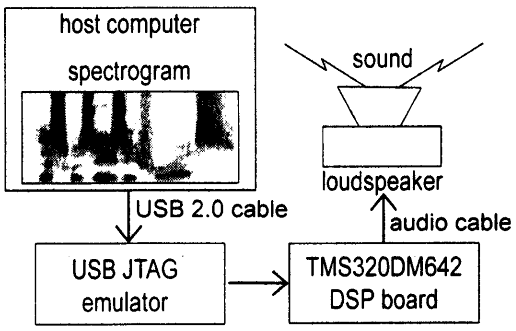

Figure 3: The environment of the system

works as follows. The digital image of the spectrogram is read into the memory on the DSP board through the USB JTAG emulator. After reading the spectrogram from the host computer, the DSP converts the image data into a sequence in time, according to the FFT method. Finally, the sequence is put into the codec, and the speech can be heard out of the loudspeaker. This procedure is implemented in C language, mainly with the nine functions in Table 1. Also, the development has been done under the environment of Code Composer Studio (CCS) version 2.21. Details of the functions in Table 1 are described below. The first function in Table 1, ***readFrmFile()***, reads the bitmap image of the spectrogram from the host computer. This function opens the "file

Table 1: Functions implementing the procedure

Name of the function
readFrmFile()
imRead()
get_Envelope()
set_SpeakingRate()
playback()
get_mainPart()
createStream()
prime()
tskPlayBack()

pointer" of the specified file by the *char* type array parameter, and reads every character of the file using the function, *fgetc()* in the standard I/O library of the C language. The characters read by *fgetc()* are stored in the memory area of the *char* type array, *bmpBuffer[WIDTH*HEIGHT *3+BMPHEADER]*. Here, *WIDTH* is the width of the bitmap image, and similarly for *HEIGHT*. Also, *BMPHEADER* is the header size of the bitmap image. The next function, *imRead()*, sets the data in the *bmpBuffer* into a two dimensional array, *xyData[WIDTH][HEIGHT]*, to make the data operable. *get_Envelope()* obtains the spectral envelopes. In practice, *xyData* is passed to the *get_Envelope()* as a parameter, and the spectral envelopes of *xyData* in each frame get stored in another *char* type array, *xyData_env[WIDTH][HEIGHT/2]*. *set_SpeakingRate()* works in the similar way to that of *get_Envelope()*. As *get_Envelope()* reduces the resolution on the frequency axis, *set_SpeakingRate()*, on the other hand, reduces the resolution on the time axis. This reduction by *set_SpeakingRate()* changes the speaking rate of the speech to be output. *playback()* is the one of the key functions in this system. The main work of this function is taking the IFFT of the input sequence. The input sequence gets fixed to a symmetric form before IFFT. And, only the real part is taken from the impulse response, which is obtained by performing IFFT. Also, the main part of the impulse response is taken in this function by calling *get_mainPart()*. *get_mainPart()* obtains the main part of the impulse response. The number of samples to obtain from the impulse response is specified by an *int* type parameter. *createStream()* creates the output stream to get the sequence into the codec through the stream. The Stream is created by the function, *SIO_create()* which DSP/BIOS API provides. *prime()* allocates the buffer for the output stream, and issues an empty buffer to the stream. The buffer issued here is to be reclaimed in the *tskplayBack()*. *tskplayBack()* is also the key function in this system. Because this is a task function, *tskplayBack()* is called after *main()* returns. First, *readFrmFile()* is called in this function. After that, *createStream()*, *prime()*, *imRead()*, *get_Envelope()*, and *set_SpeakingRate()* follow. And, *playBack()* is iteratively called within a *for()* loop in this function. At the end of this loop, the output sequence is obtained. After the buffer is reclaimed from the output stream, the sequence gets into the codec by the issuing of the buffer to the output stream again. Finally, the speech can be heard out of the loudspeaker.

4 DISCUSSION

This project tried to implement the FFT method on DSP. A spectrogram of a line along the time axis was chosen for the playback because of the simplicity. In theory, the sound of this simple spectrogram is that of a sine wave, and the output signal of the system should be a signal of a sine wave. The algorithm used in this project to implement the FFT method can be revised. In this project, the input sequence gets fixed to a symmetric form, as mentioned in the section of implementation. This is a way to make the result of IFFT have only the real part [5]. In theory, the FFT of an asymmetric signal produces a complex signal. And if the latter half of the input signal, from $\frac{N}{2}+1$ to $N-1$ (where N is the point of FFT), is all zero, then the imaginary part of this complex signal is the Hilbert transform of the real part [5]. The FFT of a symmetric signal, on the other hand, produces a sequence which has only real numbers [5]. However, if the real part is to be taken from the impulse response which the result of IFFT gives, this process, fixing the sequence to a symmetric form, can be removed. Considering the possibility of real-time DPP, the time needed for the playback should be shortened. For the reconstruction of a human speech at 100Hz, the playback of one frame needs to be done within 10ms. The removal of the process is thus worth considering. The present form of DPP in this project is a milestone for the other future forms of DPP, including real-time DPP. In this project, bitmap image is used for the target of playback. This is because another form of DPP, using a video camera as the input, is being considered. In general, a video camera is thought to produce a movie. However, a movie is actually a series of static images, and the algorithm used in this project can be applied to play back each image. The DSP board, TMS320DM642, used in this project has the interfaces for the video input, and this form of DPP, called video version, will be possible to realize on this board. The video version of DPP is also a milestone for the ideal DPP, which works in real-time. The usefulness of the real-time DPP in a pedagogical situation is pointed out by Arai et al. [3, 4]. However, real-time DPP has some problems to realize, such as the way to playback one frame within 10ms. In terms of the easiness of the distribution, the software version of DPP, which works on personal computers (PCs), is considered to be another form of DPP. The software version of DPP is useful because it works on usual computers. Anyone will be able to use it, by just downloading and installing it. Also, the web version of DPP is considered as a useful form of the other DPP. The web version will work on a website in real-time. Both the easiness to use and the effects of real-time aspect are the important points of the web version. The web version of DPP is expected to be used all over the world.

5 CONCLUSION

This project tried to implemented the FFT method on TMS320DM642 Evaluation board. As a simple example, a spectrogram of a line along the time axis was chosen for the target of the playback. In the same way, other spectrograms, such as those of human speeches, are expected to be played back. Therefore, this project is surely on the way to the ideal DPP.

6 ACKNOWLEDGEMENT

We would like to appreciate the help from Isamu Tsunoda and Makoto Ohmaru, in Partner Network & University Program of Texas Instruments Japan. We are grateful not only for their pieces of advice to the project, but also for the assistance in making the environment for the development. We would also like to thank Prof. F. Scott Howell in Sophia University. His proofreading greatly helped to correct this paper.

References

[1] F. S. Cooper, A. M. Liberman and J. M. Borst, "The interconversion of audible and visible patterns as a basis for research in the perception of speech," *Proceedings of National Academy of Sciences*, 37, 318-325, 1951.

[2] P. W. Nye, L. J. Reiss, F. S. Cooper, R. M. McGuire, P. Mermelstein and T. Montlick, "A digital pattern playback for the analysis and manipulation of speech signals," ***Haskins Lab. Status Report on Speech Research,*** SR-44, 95-107, 1975.

[3] Takayuki Arai, Keiichi Yasu, and Takahito Goto, "Digital Pattern Playback," *Proc. Autumn Meet. Acoust. Soc. Jpn,* 429-430, 2005.

[4] Takayuki Arai, Keiichi Yasu, and Takahito Goto, "Digital Pattern Playback: Converting Spectrograms to Sound for Educational Purposes," *Acoustical Science and Technology* (to be published).

[5] Alan V. Oppenheim, Ronald W. Schafer with John R. Buck, *Discrete-Time Signal Processing Second Edition,* Prentice Hall, 1998.