Hardware Acceleration of the Gabor Wavelet Transformation to Estimate Interruptibility for Working Productivity

Nam Nguyen Hai†, Kaoru Kozuka‡, Binh Huynh Thi Thanh†, Kinya Fujita‡ and Hironori Nakajo‡

Abstract– We propose hardware acceleration of the Gabor wavelet transformation for the estimation of interruptibility in order to improve working productivity. The wavelet transformation has been implemented on an FPGA board to speed up the transformation calculation by implementing its Verilog codes to be synthesized to run on an FPGA board.

Keywords: Hardware Acceleration, FPGA, Interruptibility Estimation, Gabor Wavelet Transformation

1. Introduction

Interruptions happen every day at work or in the school as the requests or notifications in both oral and PC-based forms. It is reported by a prior study that receiving frequent interruptions incurred at inappropriate times decreases one's productivity during his/her mental activities at the workplace.

Having drawn researchers' interest in the last decade, interruptibility estimation came up as a promising solution whose applications in enhancing the working productivity are solid and practical. There are several approaches towards this problem. In this article, we specifically consider the research work conducted by Hashimoto et. al. [1], in which the raw audio data recorded at the workplace, after being processed by the Gabor wavelet transformation resulting in the data in time-frequency domain, is further classified into two newly proposed conversational statuses based on a set of pre-determined rules. The experimental result tested on the 50-hour recorded audio data proved the reliability and the efficiency of this method. However, this solution turned out to offer poor performance in term of execution time which takes the root from the high computational demand of the transformation; thus, reducing the possibility for its application on real-time systems.

Regarding to the hardware acceleration, [2] presented the idea of designing and implementing the Morlet wavelet transform so that the algorithm is synthesizable on an FPGA board. In particular, the required controllers for the continuous wavelet transformation (CWT) design are built using VHDL programming that works along with schematic design available in the Altium designer software package. The experimental data shows a significant improvement in the computational time. In other words, it is illustrated that FPGA implementation of CWT greatly reduces its execution time.

For this article, we focus on minimalize the computational time of the interruptibility estimation algorithm presented in [1]. To achieve this goal, we propose the hardware acceleration of the Gabor CWT investigated within the algorithm since the module itself accounts for the longest execution time compared to the total one. In particular, the detailed solution is described in section 2.

2. Proposed strategy for hardware acceleration of the Gabor CWT

Given that the Gabor CWT algorithm presented in [1] was originally implemented in C programming, our goal becomes parsing the essential modules needed in the Gabor CWT from C programming algorithm to HDL logic. In this research, we use Verilog HDL; which means we focus on translating the Gabor CWT from C to Verilog such that the resulting modules are synthesizable on an FPGA board.

To achieve this goal, we investigate two steps. Firstly, simplify the Gabor CWT module written in C based on HDL-oriented programming logic, partially reducing its computational demand. Secondly, design and implement the hardware logic of the Gabor CWT modules after re-adjustments. After these two steps, the algorithm is guaranteed to have the complexity and computational demand lessened, which means it is supposed to offer shorter execution time.

† Hanoi University of Science and Technology
‡Tokyo University of Agriculture and Technology

2.1. HDL-oriented optimization on Gabor CWT

By the first glance at the Gabor CWT, it is clear that the algorithm is complicated and computationally demanding in comparison with the limited computation capability offered by HDL programming languages and FPGA boards.

Module SetParameters()

```
for( i=0; i<m_nGaborN; i++ ) {
 double x = (double)(i - m_nGaborOffset )/a;
 double d = exp( -x*x/(2.0*sigma*sigma) )
              /
sqrt( 2.0*_PI*sigma*sigma );
 m_pGaborR[i]  =  d  *  cos(  omega*x  )  /
sqrt( a );
 m_pGaborI[i]  =  d  *  sin(  omega*x  )  /
sqrt( a );
}
```

As an illustration, all equations in the above module are nearly impossible to implement in HDLs. However, since related variables are either known or constant, we could pre-compute the entire parameter set, thus significantly omitting a numerous of complex calculations. Similarly, we reduce or simplify most of the complex equations required in the algorithm, and what left are solely basic HDL-supported calculations.

In addition, the floating point decimal data type is not programmable in HDL; therefore, we have to store data in other supported types. For this research, we choose the single precision floating-point binary format to store data, which requires 32 bits in the binary presentation of the data to follow the IEEE-754 standard.

By the end of this step, every complex equation are either simplified or omitted such that its value is stored as a constant parameter, and the data has fully been encoded in form of the IEEE-754 format that HDL and FPGA boards support.

2.2. Hardware designing and implementing in HDL

In this step, we not only implement the hardware design of Gabor CWT module but also utilize the best of what HDL and FPGA are capable. In particular, we apply several techniques to reduce the number of clocks required for some specific operations.

To begin with, we use BRAM to store most of the variables in the cache, reducing the data retrieval time thank to the high data reading rate offered by the cache. Since hundreds of calculation is repeated every 0.5 seconds, the total number of data reading operations becomes enormous; thus using BRAM vastly increases the execution time. We also implement a nested BRAM of two, which is considered as a two dimensional BRAM, allowing better data retrieval schema that takes only one single clock to complete.

We apply the IP-core generators supported in ISE design suite to automatically design the adder, shifter, etc with correspond to the IEEE-754 floating-point format. By setting up the IP-core functions at the highest performance level, it is expected that the computational time is further shortened.

After the second step, the HDL Gabor CWT module is optimized in both algorithmic and hardware driven aspects and it is ready to run on an FPGA.

3. Conclusion and future works

We focus on applying hardware acceleration on Gabor CWT module of the Interruptibility estimation. The proposed optimization strategy covers both algorithmic and hardware-driven accelerations. Even though the final HDL Gabor CWT has not yet been fully examined with the practical recorded audio data, this strategy gives the idea of a comprehensive optimization to achieve the best result possible instead of performing only either hardware implementations or algorithmic improvements.

For future development, applying Synthesijer to parse directly from Java code to Verilog code and comparing the resulting Gabor CWT modules' performance with that derived by this research work in term of execution time is one practical direction.

ACKNOWLEDGEMENTS

This work was partially supported by MEXT Fund for Smart Space Technology toward Sustainable Society and by JSPS Grant-in-Aid for Scientific Research (C) Number 25330067.

References

[1] S. HASHIMOTO, T. TANAKA, K. AOKI, K. FUJITA, “Improvement of Interruptibility Estimation during PC Work by Reflecting Conversation Status,” IEICE Trans. Inf. & Syst. E97.D(12), 3171-3180, 2014.

[2] Y. T. Qassim, T. Cutmore, D. A. James, D. Rowlands, “FPGA Implementation of Morlet Continuous Wavelet Transform for EEG Analysis,” ICCCE 2012, At Kuala Lampur, DOI: 10.1109/ICCCE.2012.6271152