Biometric Analysis of Ear Matching using Feature Detectors

Soumyajit Sarkar

Submitted to the Department of Electrical Engineering & Computer Science and the Faculty of the Graduate School of the University of Kansas in partial fulfillment of the requirements for the degree of Master's of Science

Thesis Committee:

Dr. Guanghui Wang: Chairperson

Dr. Bo Luo

Dr. Jerzy Grzymala-Busse

Date Defended

The Thesis Committee for Soumyajit Sarkar certifies
That this is the approved version of the following thesis:

Biometric Analysis of Ear Matching using Feature Detectors

Committee:

_____

Chairperson

_____

_____

Abstract

Biometric ear authentication has received enormous popularity in re-cent years due to its uniqueness for each and every individual, even for identical twins. In this paper, two scale and rotation invariant feature detectors, SIFT and SURF, are adopted for recognition and authentication of ear images. An extensive analysis has been made on how these two descriptors work under certain real-life conditions; and a performance measure has been given. The proposed technique is evaluated and compared with other approaches on two data sets. Extensive experimental study demonstrates the effectiveness of the proposed strategy. Deep Learning has become a new way to detect features in objects and is also used extensively for recognition purposes. Sophisticated deep learning techniques like Convolution Neural Networks(CNNs) have also been implemented and analysis has been done.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Biometric authentication of people based on various anatomical characteristics, like eye, ear, face, iris, and fingerprint have attracted lots of attention during the past few decades, and some of these techniques have already been successfully applied for recognition and authentication. However, many systems are not very robust and may fail to work under certain conditions. Biometric ear recognition is a relatively new technique that may surpass the existing systems due to several significant reasons. For example, the acquisition of ear images is relatively easy and, unlike iris, can be captured without the co-operation of individuals [**?**]

Human ear contains rich and stable features which are more reliable than face fea- tures, as the structure of the ear is not subject to change with age. It has also been found out that no two ears are exactly the same even for identical twins [3]. The detailed structure of ear is not only very unique but also permanent, since the shape of a human ear never shows drastic changes over the course of life. The research on ear identification was first conducted by Bertillon, a French criminologist, in 1890. The process was refined by American police officer, Iannarelli [20], who divided the ear based on vari- ous distinctive features of seven parts: i.e.

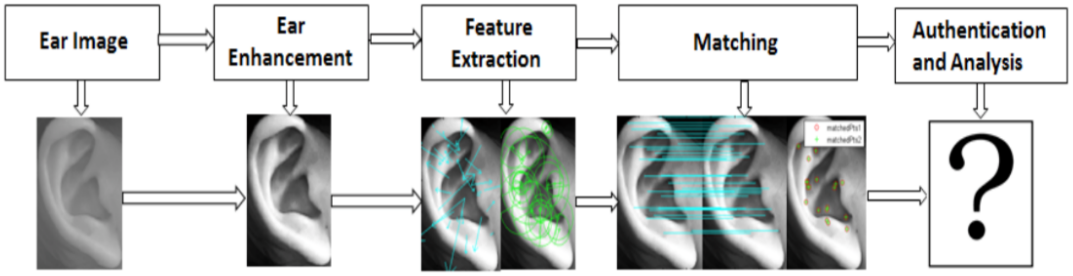

Figure 1.1. The pipeline of the proposed Ear Recognition System

helix, concha, antihelix, crux of helix, inter- tragic notch, tragus, and antitragus [3].

In this Paper, we propose to use two scale and rotation invariant feature detectors, i.e. SIFT (scale invariant feature transform) and SURF (speed up robust features), for ear recognition. Both SIFT and SURF extract specific interest points from an image and generate descriptors for the feature points to a form a reliable matching results.

Extensive experiments have been carried out on two different sets of databases to eval- uate their performance with respect to various rotations and scales. One of the most important feature of ear images is its easiness in acquisition, however, the acquired images may be in different scales, rotations, and illumination. The scale and rotation invariant property of the SIFT and SURF algorithms makes them perfect for ear authentication under various circumstances.

The rest of the paper is organized as follows. Some background and related research are discussed in Section 2; the proposed method is presented in details in Section 3; some experimental results and analysis are given in Section 4; and the paper is concluded in Section 5.

Chapter 2

Statement of the Problem

2.1 Types of Biometrics

Hybrid CPU/FPGA systems form a union between two different areas of expertise: hardware design and software

2.2 Purpose of Biometric Recognition

FPGAs provide a "blank" slate of computational components. Standard interfaces and services have not

2.3 Contributions of this Project

The main goal of this work is to develop IP cores capable of providing run-time scheduling services to all threads, both

Chapter 3

Background

Most real-time operating systems, such as RTLinux [**?**], attempt to minimize the jitter introduced by asynchronous requests by pre-processing external requests within a small, fast interrupt service routine that determines if the request should be immediately handled, or can simply be marked as pending for future service. While this approach does reduce jitter, it is still based on the semantics of allowing asynchronous invocations to interrupt the CPU, and incurring the overhead of a context switch to an interrupt service routine.

Chapter 4

Related Works

increase the size of the ready-to-run queue, only slightly more logic resources for writes and 2 clock cycles for reads.

Chapter 5

Design of Proposed Approach

5.1 Traditional Approach

The initial scheduler in the HybridThreads system used a simple FIFO scheduling mechanism that was internal to the Thread Manager (TM) [**?**]. The add_thread system call would be routed to the Thread Manager and the TM would then insert the thread into a singly-linked FIFO ready-to-run queue. The main problem with this scheduling structure is that the ready-to-run queue was built in to the Thread Manager's attribute structures used for data storage for thread management status. This means that thread management (allocation, creation, and control) and thread scheduling (ready-to-run queue management, and scheduling decisions) activities could not occur in parallel. Additionally, any changes in scheduling mechanisms and scheduling data arrangement would also affect the management mechanisms, and vice-versa, so maintenance of the management and scheduling services would be difficult, cumbersome, and error-prone.

Both the thread management and thread scheduling mechanisms would have to be modified if either were to be extended in their functionality or data storage

requirements. The scheduling mechanism was going to be upgraded to allow for priority scheduling and eventually would have to handle the scheduling of both SW and HW threads, so it was decided to make the scheduler a separate IP module that would have its own interface to the HybridThreads system bus. Many thread management operations result in the invocation of scheduling operations, so essentially the TM uses the scheduler module as a coprocessor for any and all scheduling operations. Many of these coprocessor operations can only occur as a result of a management operation so the TM will always be the "caller" in these cases. This, in conjunction with the scheduler becoming a separate module, means that all outgoing operations from the TM to the scheduler will result in a bus operation; however if the TM is using the scheduler to complete a management operation, then the bus will already be locked by the caller of the TM operation. This meant that a special interface must be created between the TM and the new scheduler module to allow access to scheduling operations while the system bus was locked. Additionally, since other scheduler specific operations are not ever called as the result of a thread management operation, then these scheduling operations can be called via the new interface used to attach the independent scheduler module to the HybridThreads system bus.

The TMcom interface is a dedicated hardware interface between the scheduler module and the TM that consists of a total of seven control and data signals as well as access to a read-only interface (B-port) of the TM's Block RAM (BRAM). The data signals include `Next_Thread_ID`, `Current_Thread_ID`, and `Thread_ID_2_Sched`. The `Next_Thread_ID` signal represents the identifier of the thread chosen to run next on the CPU. This signal is writable by the scheduler module and readable by the TM. The `Current_Thread_ID` signal repre-

sents the identifier of the thread that is currently running on the CPU (PowerPC 405). This signal is readable by the scheduler module and writable by the TM. The `Thread_ID_2_Sched` signal contains the identifier of the thread that is being added to the ready-to-run queue by the TM. This signal is readable by the scheduler and writable by the TM. The control signals include `Next_Thread_Valid`, `Dequeue_Request`, `Enqueue_Request`, and `Enqueue_Busy`. The `Next_Thread_Valid` signal represents whether or not that the scheduling decision available from the scheduler module on the `Next_Thread_ID` signal is valid or not (Valid = 1, Invalid = 0). This signal is writable by the scheduler module and readable by the TM. The `Dequeue_Request` signal is used by the TM to request the scheduler module to perform a dequeue operation of the thread whose identifier is on the `Next_Thread_ID` signal. This signal is readable by the scheduler module and writable by the TM. The `Enqueue_Request` signal is used by the TM to request the scheduler module to perform an enqueue operation of the thread whose identifier is on the `Thread_ID_2_Sched` signal. This signal is readable by the scheduler module and writable by the TM. The `Enqueue_Busy` signal represents whether or not the scheduler is currently busy performing an enqueue operation (Busy = 1, Not Busy = 0). This signal is writable by the scheduler module and readable by the TM. The B-Port interface to the TM's BRAM allows the scheduler module to query thread management information in order to perform error-checking that concerns the parent-child relationships of threads that the TM's data structures hold. The purpose of the TMcom interface is to allow the TM to request scheduling operations as a result of thread management operations whose side-effects alter the scheduling status of the system (i.e. the status of the ready-to-run queue). The operations available through the TMcom interface can be seen in table 5.1.

Table 5.1. Command Set of Scheduler Module, Build 1

Type	Name	Actions
TMcom	Enqueue	Schedules a thread
TMcom	Dequeue	Removes a thread from the ready-to-run queue
BUScom	Get_Entry	Returns a thread's table attribute entry
BUScom	Toggle_Preemption	Toggle preemption interrupt on/off
BUScom	Get_Entry	Returns a thread's table attribute entry (for debug use)
BUScom	Get_Priority	Returns the priority-level of a thread
BUScom	Set_Priority	Sets the priority-level of a thread
BUScom	Set_Default_Priority	Sets the priority-level of a thread (no error-checking)

5.2 SIFT and SURF Descriptor

The main goal of the second redesign of the scheduler module is to further reduce the amount of overhead and jitter involved in thread scheduling

5.3 Training Model

One of the main goals of The new hardware components used to schedule both SW and HW threads were fully integrated into the existing scheduler module. Results from simulation and synthesis tests to verify scheduler correct scheduler functionality of the new hybrid features along with O(1) ready-to-run queue structure can be found in chapter 6.

5.4 Deep Learning Approach

5.5 Convolution Neural Network

5.6 Our Deep Network and Model

Chapter 6

Implementation Results

The modules from each of the scheduler redesigns have undergone a series of tests in both simulated and synthesized forms. These tests help show the performance differences between the different ready-to-run queue structures in terms of scheduling overhead and jitter as well as the effects of the hybridization of the scheduler on the entire HybridThreads operating system.

6.1 Results of the Traditional Approach

making a scheduling decision is hidden because it occurs during a context switch. However, if a scheduling event occurs very soon after a thread is initially context switched to, the system may have to wait for the scheduler to finish calculating the next scheduling decision resulting from the context switch. This is more likely to happen with more threads in the ready-to-run queue, due to increased execution-times of scheduling operations due to the O(n) nature of the queue itself. Although the amount of jitter is in the microsecond range, it can still be further reduced by restructuring the ready-to-run queue so that its functions

are able to operate in constant amounts of time.

6.2 Results of the Deep Approach

Synthesis of the second redesign of the scheduler module targeting a Xilinx [**?**] Virtex-II Pro 30 yields the following FPGA resource statistics: 1,034 out of 13,696 slices, 522 out of 27,392 slice flip-flops, 1,900 out of 27,392 4-input LUTs, and 2 out of 136 BRAMs. The module has a maximum operating frequency of 143.8 MHz, which easily meets our goal of a 100 MHz clock frequency.

6.3 Comparison of the Approaches

Synthesis of the third redesign of the scheduler module targeting a Xilinx [**?**] Virtex-II Pro 30 yields the following FPGA resource statistics: 1,455 out of 13,696 slices, 973 out of 27,392 slice flip-flops, 2,425 out of 27,392 4-input LUTs, and 3 out of 136 BRAMs. The module has a maximum operating frequency of 119.6 MHz, which easily meets our goal of a 100 MHz clock frequency.

Chapter 7

Conclusion

In this paper we have presented the scheduling operations that execute in under 50 clock cycles (500 ns) or less at the base hardware level. From the system level, the hardware scheduler module provides very fast scheduling operations with an end-to-end scheduling delay of 1.9 μs with 1.4 μs of jitter with 250 active threads running on a Xilinx [**?**] Virtex-II Pro FPGA. The integrated system level tests have shown that the migration of scheduling services into the fabric of the FPGA have drastically reduced the amount of system overhead and jitter related

HybridThreads project [**?**], which furthers the future impacts of the project even more. More detailed descriptions of the hybrid threads project can be found in [**?**].

Acknowledgment

The work in this article is partially sponsored by National Science Foundation EHS contract CCR-0311599. The opinions expressed are those of the authors and not necessarily those of the foundation.