creative commons

COMMONS DEED

저작자표시-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.
- 이 저작물을 영리 목적으로 이용할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

[UCI]I804:24011-200000239585

2010年 2月
碩士學位論文

지문 퍼지볼트 시스템의 VLSI 구현에 관한 연구

朝鮮大學校 大學院
情報通信工學科
林 成 晉

지문 퍼지볼트 시스템의 VLSI 구현에 관한 연구

A Study on the VLSI Implementation of the Fuzzy Fingerprint Vault System

2010年　2月　　日

朝鮮大學校 大學院

情報通信工學科

林 成 晉

지문 퍼지볼트 시스템의 VLSI 구현에 관한 연구

指導敎授　潘　聲　範

이 論文을 工學 碩士學位申請 論文으로 提出함

2009年　10月

朝鮮大學校 大學院

情報通信工學科

林 成 晉

林成晉의 碩士學位 論文을 認准함

委員長　朝鮮大學校 敎授 高 樂 溶 印

委　員　朝鮮大學校 敎授 郭 根 昌 印

委　員　朝鮮大學校 敎授 潘 聲 範 印

2009年　11月

朝鮮大學校 大學院

저작물 이용 허락서

학 과	정보통신공	학 번	20087132	과 정	석사
성 명	한글: 임 성 진 한문: 林 成 晉 영문: Lim Sung Jin				
주 소	광주광역시 서구 치평동 라인동산아파트 105동 806호				
연락처	E-MAIL : gigasj83@gmail.com				
논문제목	한글 : 지문 퍼지볼트 시스템의 VLSI 구현에 관한 연구 영문 : A Study on the VLSI Implementation of the Fuzzy Fingerprint Vault System				

본인이 저작한 위의 저작물에 대하여 다음과 같은 조건아래 조선대학교가 저작물을 이용할 수 있도록 허락하고 동의합니다.

- 다 음 -

1. 저작물의 DB구축 및 인터넷을 포함한 정보통신망에의 공개를 위한 저작물의 복제, 기억장치에의 저장, 전송 등을 허락함
2. 위의 목적을 위하여 필요한 범위 내에서의 편집·형식상의 변경을 허락함. 다만, 저작물의 내용변경은 금지함.
3. 배포·전송된 저작물의 영리적 목적을 위한 복제, 저장, 전송 등은 금지함.
4. 저작물에 대한 이용기간은 5년으로 하고, 기간종료 3개월 이내에 별도의 의사 표시가 없을 경우에는 저작물의 이용기간을 계속 연장함.
5. 해당 저작물의 저작권을 타인에게 양도하거나 또는 출판을 허락을 하였을 경우에는 1개월 이내에 대학에 이를 통보함.
6. 조선대학교는 저작물의 이용허락 이후 해당 저작물로 인하여 발생하는 타인에 의한 권리 침해에 대하여 일체의 법적 책임을 지지 않음
7. 소속대학의 협정기관에 저작물의 제공 및 인터넷 등 정보통신망을 이용한 저작물의 전송·출력을 허락함.

동의여부 : 동의(●) 반대( )

2010년 2월 일

저작자: 임 성 진 (서명 또는 인)

조선대학교 총장 귀하

목 차

표목차 iii

도목차 iv

ABSTRACT vi

제 1 장 서 론 1

제 1 절 연구 배경 1

제 2 절 연구 목적 2

제 3 절 연구 내용 및 방법 3

제 2 장 지문인식 5

제 1 절 등록 과정 6

1. 전처리 과정 6

2. 특징점 추출 10

제 2 절 인식 과정 12

1. 정렬 과정 12

2. 정합 과정 14

제 3 장 지문 퍼지볼트 16

제 1 절 퍼지볼트 16

제 2 절 기하학적 해싱 기법 19

제 3 절 기하학적 해싱 기반 지문 퍼지볼트 22

제 4 장 지문 퍼지볼트 시스템의 하드웨어 구현 26

제 1 절 하드웨어 구조 26

제 2 절 병렬처리 기법을 적용한 하드웨어 구조 31

제 3 절 성능 분석 39

제 5 장 결 론 52

참고문헌 53

표 목 차

표 1. 제안한 하드웨어 모듈의 주요자원 42
표 2. 매칭모듈의 수에 따른 병렬처리 기법 하드웨어 구조의 자원량..... 49
표 3. 거짓 특징점 수에 따른 정합과정의 cycle 수 50

도 목 차

그림 1. 바이오인식의 종류 2
그림 2. 지문의 구조 5
그림 3. 지문인식의 흐름도 6
그림 4. 전처리 과정 7
그림 5. 정규화된 지문영상 7
그림 6. 지문의 방향성분 추출 8
그림 7. 지문영상의 이진화 9
그림 8. 지문영상의 세선화 10
그림 9. 지문영상의 특징점 10
그림 10. 특징점 타입과 각도 측정 11
그림 11. 최종 특징점 12
그림 12. 3개의 특징점을 이용한 특징점 구조 13
그림 13. 특징점 보정(Δx : −65, Δy : 23, Δ Θ : −2) 13
그림 14. 특징점 오차 허용범위 14
그림 15. 특징점 대조 15
그림 16. 퍼지볼트의 볼트 생성 과정 16
그림 17. 퍼지볼트의 인코딩 과정 17
그림 18. 퍼지볼트의 디코딩 과정 18
그림 19. 기하학적 해싱의 블록도 20
그림 20. 해시 테이블 생성(기준점 1) 21
그림 21. 기하학적 해싱 기반 지문 퍼지볼트의 등록과정 23
그림 22. 등록 해시 테이블 생성 과정 24
그림 23. 기하학적 해싱 기반 지문 퍼지볼트의 인식과정 25
그림 24. 지문 퍼지볼트 시스템의 소프트웨어와 하드웨어 통합 구현 26
그림 25. 제안하는 하드웨어 모듈의 구조도 27

그림 26. 변환 특징점의 구조 28
그림 27. 제안하는 하드웨어 모듈의 구조도 28
그림 28. 제안하는 하드웨어 모듈의 흐름도 29
그림 29. 제안하는 하드웨어 모듈의 동작 과정 30
그림 30. 제안하는 병렬처리 기법 하드웨어의 매칭모듈 구조도 32
그림 31. 2개의 매칭모듈을 가진 하드웨어 모듈의 구조도 32
그림 32. 2개의 매칭모듈을 가진 하드웨어의 흐름도 34
그림 33. 4개의 매칭모듈을 가진 하드웨어 모듈의 구조도 35
그림 34. 4개의 매칭모듈을 가진 하드웨어의 흐름도 37
그림 35. 64개의 매칭모듈을 가진 하드웨어의 구조도 38
그림 36. 제안하는 병렬처리 하드웨어 동작 과정 39
그림 37. Spartan 3E starter board 40
그림 38. 제안한 하드웨어 매칭모듈의 시뮬레이션 결과 41
그림 39. 제안한 하드웨어 구조의 회로 합성결과 41
그림 40. 2개의 매칭모듈을 가진 하드웨어 구조의 실험결과 43
그림 41. 4개의 매칭모듈을 가진 하드웨어 구조의 실험결과 44
그림 42. 8개의 매칭모듈을 가진 하드웨어 구조의 실험결과 45
그림 43. 16개의 매칭모듈을 가진 하드웨어 구조의 실험결과 46
그림 44. 32개의 매칭모듈을 가진 하드웨어 구조의 실험결과 47
그림 45. 64개의 매칭모듈을 가진 하드웨어 구조의 실험결과 48
그림 46. 거짓 특징점 수에 따른 정합과정의 수행시간(sec) 51

ABSTRACT

A Study on the VLSI Implementation of the Fuzzy Fingerprint Vault System

Lim, Sung Jin
Advisor : Prof. Pan, Sung Bum Ph.D.
Department of Information & Communications,
Graduate School of Chosun University

Along with the development of internet, we could find and read the desired information anytime and anywhere we want. On the other hand, the outflow of personal information or crime using a blind spot of authentication system has been occurred frequently. Presently, the password method used in most of applications consists of the characters that can be easily memorized by user and the same password is used for several applications. Such password is possible for easy guess and outflow and has a risk of forgetting. Accordingly, the need for safer and more powerful authentication system is increasing. As the system that can make up the weakness of existing authentication system, the studies on the biometric system of using bio-information have been actively conducted. The fingerprint authentication of using fingerprint information is convenient and provides a powerful means of security at the same time. Since the fingerprint information change cannot be done freely due to the limited number of fingers, it may cause a serious problem if the fingerprint information saved for user authentication is exposed to other people. In order to protect the fingerprint information safely, the studied on the fuzzy fingerprint vault system have been conducted. The fuzzy fingerprint

vault system generates a polynomial by using the secret keys of user and protects the fingerprint minutiae of user by constructing a fingerprint template together with the fingerprint minutiae after generating the chaff minutiae randomly. Although a few studies have been reported to protect the secret key of user as well as the fingerprint information of user by using the fuzzy fingerprint vault, they have a problem that cannot perform the alignment process due to the absence of fingerprint reference point. In order to solve this problem, the method to apply a geometric hashing technique to the fuzzy fingerprint vault has been proposed. The fingerprint alignment problem was resolved by performing a matching process upon comparing the enrollment hash table obtained through the geometric hashing technique with the verification hash table. However as a large number of chaff minutiae are added, the number of fingerprint templates is increased, thereby delaying the matching time from an increased number of hash tables generated.

In order to reduce the execution time of fuzzy fingerprint vault system, this thesis proposes hardware architecture of fuzzy fingerprint vault system based on the geometric hashing. The proposed architecture is constructed by integrating the software and hardware modules. The hardware module consists of the matching module, polynomial restoration module, verification module, two memory for storing the enrollment hash table and verification hash table. The matching module compares the transformation minutiae of enrollment hash table with the transformation minutiae of verification hash table. The polynomial restoration module refers to the module that restores the polynomial coefficients by using the matched conversion minutiae. The verification module is the module that compares the coefficients of restored polynomial with the coefficients of the polynomial in the enrollment process. Also for the high-speed processing of fuzzy fingerprint vault system, this

thesis proposes the hardware architecture that has applied a parallel processing technique. The hardware module consists of two memory for saving the enrollment hash table and the verification hash table, number of matching modules, and verification module.

Based on the experimental results, the execution time in the matching process for 36 fingerprint minutiae, 200 chaff minutiae and 34 authentication fingerprint minutiae was 1.13 second when having implemented the system only with the software, 0.24 second with the proposed hardware architecture, and 0.01 second with the parallel processing hardware architecture of holding 64 matching modules.

제1장 서 론

제1절 연구 배경

현대 사회에서는 인터넷과 같은 통신 매체의 발달로 언제, 어디서든 원하는 정보를 손쉽게 접근하고 열람할 수 있다. 정보에 대한 접근이 용이해짐에 따라 사용자 인증과 개인의 정보를 보호할 수 있는 인증 시스템의 필요성이 증가하고 있다. 그러나 현재 대부분의 응용에서 지원하는 패스워드 또는 PIN(Personal Identification Number) 방식은 패스워드의 길이가 충분히 크지 않으며, 지원되는 패스워드 크기 내에서 사용자들은 랜덤한 패스워드를 사용하는 대신 사용자가 기억하기 쉬운 문자 등으로 여러 응용에 동일한 패스워드를 사용하고 있다. 이러한 기존의 인증 시스템은 추측, 유출 또는 망각의 위험이 있어 보안상의 문제가 있다[1-2].

바이오인식(Biometric)은 사용자가 따로 소지하거나 암기할 필요가 없는 편리성과 반드시 본인이 있어야 인식되는 높은 신뢰성으로 패스워드, PIN, 그리고 스마트카드 등을 보완할 수 있는 새로운 보안 수단이다. 바이오인식이란 그림 1과 같이 사람의 신체적, 행동적 특징을 추출하고 분석하여 정확하게 개인의 신원을 확인하는 기술을 말한다. 신체적 특성을 이용한 기술로는 지문, 눈의 망막 및 홍채, 얼굴 인식 등이 있으며, 행동적 특성을 이용한 기술로는 서명과 음성 인식 등이 있다. 바이오인식 기술의 핵심은 빠르고 정확한 인증이다. 가격, 정확도, 편의성 등을 고려하여 사용 가능성이 가장 높은 분야는 지문 인식 기술로서 전체 바이오인식 시장의 절반을 차지하고 있으며, 현재는 전자여권 및 신원 확인 등의 공공분야를 중심으로 확산되는 추세이다[3].

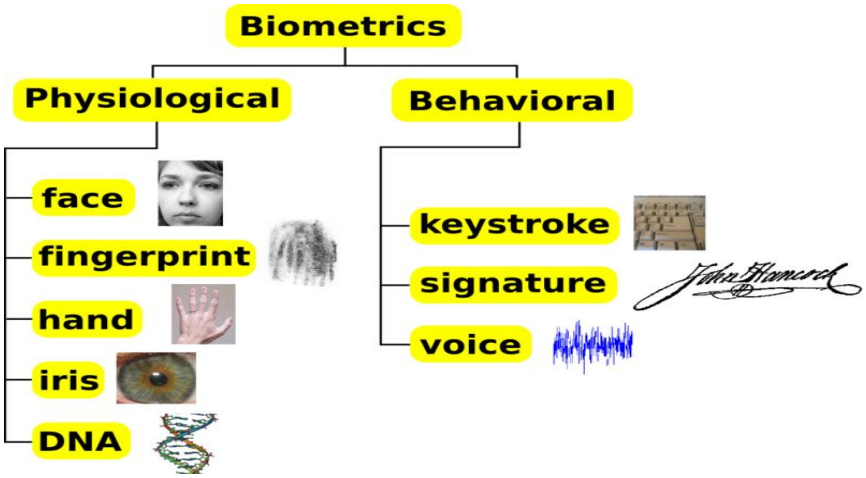


그림 1. 바이오인식의 종류

지문정보를 이용한 사용자 인증 시스템에서도 몇 가지 문제가 있다. 먼저, 패스워드와 달리 동일인의 동일 손가락으로부터 입력되는 지문정보가 매번 유사하지만 완전히 일치되지 않는다는 단점이 있다. 또한, 사람의 손가락은 한정되어 있기 때문에 유출 시 변경이 어려우며, 일반적으로 사용자는 동일한 바이오 정보를 다양한 응용에 사용하기 때문에 유출된 바이오 정보가 모든 응용에 재사용될 수 있다는 문제가 있어 지문정보 보호 문제가 심각한 이슈로 등장하고 있다.

제 2 절 연구 목적

인터넷이 발달함에 따라 개인 정보의 유출이나 인터넷뱅킹에 관련한 사건 등과 같이 인터넷의 인증 시스템의 허점을 이용한 범죄가 빈번하게 발생되고 있다. 이에 따라 인가된 사람과 그렇지 않은 사람을 구별하여 정보를 보호하는 일이 어느 때보다도 중요하다. 바이오 정보를 이용한 바이오인식 시스템은 이러한 사회에 적합한 인식 시스템으로 연구가 활발하게 이루어지고 있다. 그 중에 지문정보를 이용한 지문인식은 지문정보의 유일성과 불변성을 가지고 있으며, 다른 바이오인식에 비해 인증 정확도가 높다. 또한, 장비의 소형화가 가능하여 다양한 분야에 적용할 수 있

다. 하지만 지문정보가 도용이나 유출될 경우 변경이 어려우며 지문인식을 사용할 수 없는 문제가 발생된다. 따라서 지문정보를 보호해야 한다[4].

본 논문에서는 지문정보 보호를 위해 기하학적 해싱 기법을 적용한 지문 퍼지볼트 시스템을 이용하였다. 이는 퍼지볼트에 기반하여 지문 템플릿에 거짓 특징점을 추가하여 지문 특징점을 보호하는 지문 퍼지볼트에 지문 정렬문제를 해결하기 위해 기하학적 해싱 기법이 적용된 시스템이다. 기하학적 해싱을 통해 생성된 등록 해시 테이블과 인증 해시 테이블을 비교하여 정합과정을 수행함으로써 지문 정렬 문제를 해결하였지만, 지문 특징점을 보호하기 위해 다수의 거짓 특징점이 추가될수록 지문 템플릿의 수가 증가하게 되고 그에 따라 생성되는 해시 테이블의 수도 증가하여 정합 수행시간이 길어지게 된다. 본 논문에서는 기하학적 해싱 기반 지문 퍼지볼트 시스템의 실시간 처리를 위한 하드웨어 구조를 제안하였다.

제 3 절 연구 내용 및 방법

본 논문에서 사용된 지문 퍼지볼트는 사용자의 비밀키와 지문 정보를 통합하여 정당한 사용자만이 비밀키를 획득할 수 있는 암호학적 방법이다. 사용자의 비밀키를 이용하여 다항식을 생성하고, 거짓 특징점(Chaff minutiae)을 임의적으로 생성한 후 사용자의 지문 특징점(Real minutiae)과 함께 지문 템플릿을 구성함으로써 사용자의 지문 특징점을 보호한다. 지문 퍼지볼트를 이용하여 사용자의 지문정보와 그 사용자의 비밀키를 동시에 보호하려는 연구가 보고되고 있으나, 지문의 기준점 부재에 따른 정렬 과정을 생략하였기 때문에 실용화 할 수 없는 문제점이 있다. 이 지문 정렬 문제를 해결하기 위하여 지문 퍼지볼트 시스템에 기하학적 해싱 기법을 적용하는 방법이 제안되었다. 기하학적 해싱 기법은 객체인식 알고리즘으로 객체의 정보를 추출하고 기하학적 변형을 수행하여 변형된 객체의 정보를 데이터베이스에 저장, 검색하는 알고리즘이다.

본 논문에서는 기하학적 해싱 기반 지문 퍼지볼트 시스템의 하드웨어 구조를 제안하였다. 제안한 구조는 소프트웨어와 하드웨어 모듈을 통합 구현한다. 소프트웨

어 모듈은 지문 특징 정보 추출, 지문 템플릿 생성, 지문 해시 테이블 생성, 그리고, 데이터베이스에 저장하는 모듈로 구성된다. 하드웨어 모듈은 매칭모듈, 다항식 복원 모듈, 인증모듈 및 등록 해시 테이블과 인증 해시 테이블을 저장하는 메모리로 구성된다. 매칭모듈은 등록 해시 테이블의 변환 특징점과 인증 해시 테이블의 변환 특징점을 비교한다. 다항식 복원모듈은 일치된 변환 특징점을 이용하여 다항식의 계수를 복원하는 모듈이며, 인증모듈은 복원된 다항식의 계수와 등록과정에서의 다항식의 계수를 비교하는 모듈이다. 또한, 지문 퍼지볼트 시스템의 고속처리를 위해 병렬처리 기법을 적용한 하드웨어 구조를 제안한다. 소프트웨어 모듈은 앞서 제안한 하드웨어 구조와 동일하다. 하드웨어 모듈은 등록 해시 테이블과 인증 해시 테이블을 저장하는 메모리, 2^n개의 매칭모듈, 그리고 인증모듈로 구성된다. 제안한 구조가 정상적으로 동작함을 확인하기 위해 소프트웨어 모듈은 Visual C++ 6.0에서 C언어를 사용하여 구현하였다. 하드웨어 모듈은 Xilinx ISE 9.2에서 VHDL(Very high speed integrated circuit Hardware Description Language)언어를 사용하여 설계하였다. 하드웨어 시뮬레이션은 Modelsim XE 6.0에서 실험하였다.

제 2 장 지문인식

지문의 역사는 고대사회부터 시작되었지만 현대의 지문비교기술은 1684년부터 시작 되었다. 영국의 Royal Society 의 Nehemiah Gruw는 사람마다의 지문이 다르며, 환상선(Loop), 와상문(Whorls), 궁상문(Arch)으로 구별 된다는 것을 처음으로 발견하였다. 오늘날 이러한 특징들은 융선의 단점과 분기점으로 대표되는 특징점으로 세분화해서 구별된다. 1880년에 이르러 지문인식 기술은 주로 범죄 수사 분야에 이용되었다. 1892년에, 갈턴은 지문의 지속성을 포함한 동시대 지문 과학, 지문의 유일성, 지문의 분류를 논의한 "Fingerprints" 라는 책을 발표했다. 신원 확인을 위하여 지문식별을 처음으로 사용하기 시작한 것은 1901년에 철도 노동자의 보수 관리의 편리를 위해 지문을 도입한 Edward Henry였다. 1960년대 후반에 사람의 지문을 전자적으로 기록할 수 있는 Live-Scan 시스템이 개발된 후 많은 수의 지문을 데이터베이스화 하는 것과 이를 자동 식별하여 검출하는 기술이 꾸준히 개발되었다[5-6].

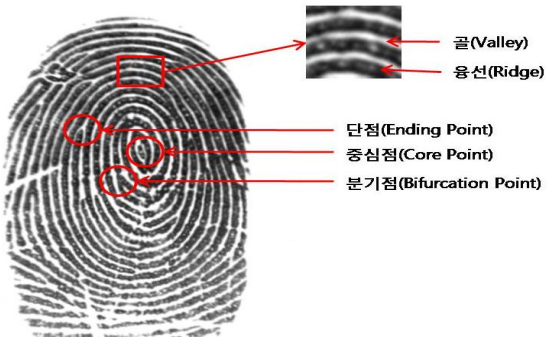


그림 2. 지문의 구조

일반적으로 사람의 지문은 그림 2와 같이 융선(Ridge)과 골(Valley)의 형태를 가지고 있다. 융선이 이루고 있는 형태에 따라 분기점(Bifurcation Point), 단점(Ending Point), 중심점(Core Point) 등과 같은 특징점(Minutiae)으로 구분된다.

지문인증 방법 중 가장 일반적인 방법은 특징점을 이용한 지문인증 방법이다. 특징점은 특징점의 좌표(x, y), 각도(θ), 그리고 종류(분기점, 단점)으로 표현할 수 있다. 지문인식은 그림 3과 같이 등록과 인식과정으로 구분된다. 오프라인에서의 지문등록은 전처리, 특징점 추출, 그리고 저장하는 순으로 진행된다. 그리고 온라인에서의 지문인식은 입력된 영상에서 추출된 특징점과 저장된 특징점을 비교하여 본인 여부를 확인한다.

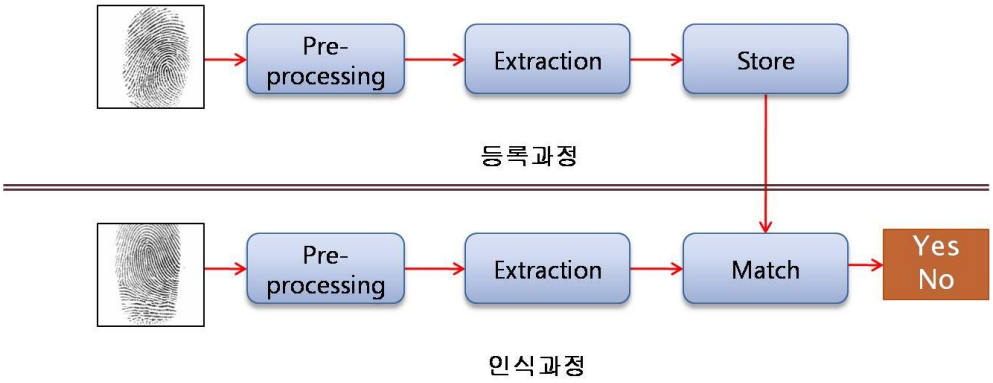


그림 3. 지문인식의 흐름도

제 1 절 등록 과정

1. 전처리 과정

등록과정은 전처리, 특징점 추출, 그리고 데이터베이스에 저장하는 단계로 구성되어 있다. 지문입력기를 통해 입력된 지문영상에서 특징점을 추출하는데 지문영상에서 특징점을 용이하게 추출하기 위해 그림 4와 같이 전처리 과정을 거친다. 지문의 입력과정에서 손가락에 묻은 땀, 먼지 등으로 발생하는 지문영상의 왜곡을 개선하고 특징점 추출에 유리한 지문 형태로 위한 전처리 과정이다.

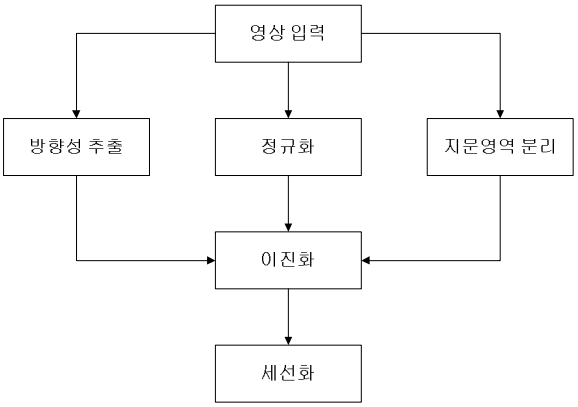


그림 4. 전처리 과정

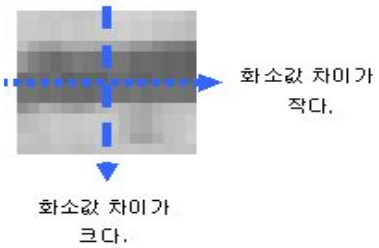
먼저 정규화 과정은 지문영상의 융선과 골의 구조는 바꾸지 않고 영상의 상태 변화를 최소로 조정하는 과정이다. 정규화가 완료된 지문영상은 그림 5와 같이 영상의 명암 분포가 일정구간으로 축소된다.

(a) 원본 지문영상 (b) 정규화된 지문영상

그림 5. 정규화된 지문영상

지문은 국부 영역에서 동일한 방향성을 가진다. 이런 국부 영역의 융선 흐름을 계산하는 것이 방향성 계산이다. 융선의 방향성을 결정하는 기본 원리는 그림 6의 (a)와 같이 지문의 융선 흐름 방향으로는 픽셀 값의 차이가 적고 융선의 흐름에 수직인 방향으로는 픽셀 값의 차가 크다는 것이다. 그림 6의 (c)는 8×8 크기의

방향 블록을 이용하여 지문영상의 방향성을 나타낸 것이다.



(a) 방향성 추출의 기본 원리

(b) 원본 영상

(c) 방향성 모습

그림 6. 지문의 방향성분 추출

지문 영역 분리(Segmentation)과정은 입력된 지문 영상에서 지문의 전경과 배경을 분리하는 것이다. 전경과 배경을 분리할 경우 정확한 비교영역에서 지문정합을 할 수 있고 불필요한 잡음을 제거할 수 있다.

지문영상은 일반적으로 256 레벨의 명도값을 갖는데, 이를 융선과 골의 두 상태로 변환하는 과정이 이진화이다. 이진화 방법은 여러 가지가 있지만 임계값에 의한 이진화 같은 경우 영상의 상태에 따라 값이 많은 오차를 가지게 되고, 검은 포인트들이 많이 생겨 세선화를 하는 과정에서 영상의 품질이 떨어질 뿐만 아니라 데이터의 손실도 많게 된다. 일반적으로 지문에서는 블록 방향성을 고려한 이진화 알고리즘이 널리 적용 된다. 그림 7의 (b)와 같이 지문영상의 이진화 영상을 구한다.

(a) 원본 지문영상 (b) 이진화 지문영상

그림 7. 지문영상의 이진화

세선화는 이진화된 영상에서 융선을 1픽셀 두께의 융선으로 표현하는 과정을 의미한다. 세선화 된 영상은 특징점 검출이 용이하므로 많은 추출 알고리즘에서 세선화 과정을 수행한다. 일반적으로 두꺼운 물체는 세선화 한 다음 골격선으로만 이를 표시하면 원래의 모습을 거의 찾아볼 수 없게 되나 지문 영상의 경우 폭이 좁고 길게 나타나는 형상의 집합이므로 1픽셀 두께의 영상으로 표현하기에 적합한 대상체이다. 세선화의 요건은 다음과 같다[7-9].

① 세선화 결과 골격선 폭은 1픽셀이어야 한다.
② 골격선의 위치는 선 도형의 중심에 위치해야 한다.
③ 골격선은 원래의 도형에 있어서의 연결성을 유지해야 한다.
④ 세선화 과정에서 골격선의 길이가 계속해서 줄어들어서는 안 된다.
⑤ 패턴 윤곽선의 작은 요철로 인한 잡가지의 모양이 골격선에 첨가 되지 않아야 한다.

지문 영상을 세선화 할 경우 그림 8의 (b)와 같은 형태로 표현된다.

(a) 원본 지문영상 (b) 세선화 지문영상

그림 8. 지문영상의 세선화

2. 특징점 추출

그림 9와 같이 세선화된 영상에서 융선이 끝나는 단점, 융선이 두 갈래이상 나누어지는 분기점과 각 특징점의 각도를 추출해 낸다.

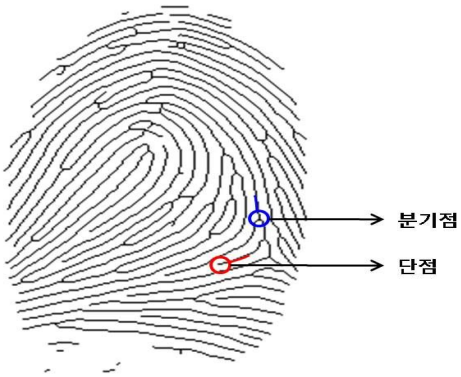
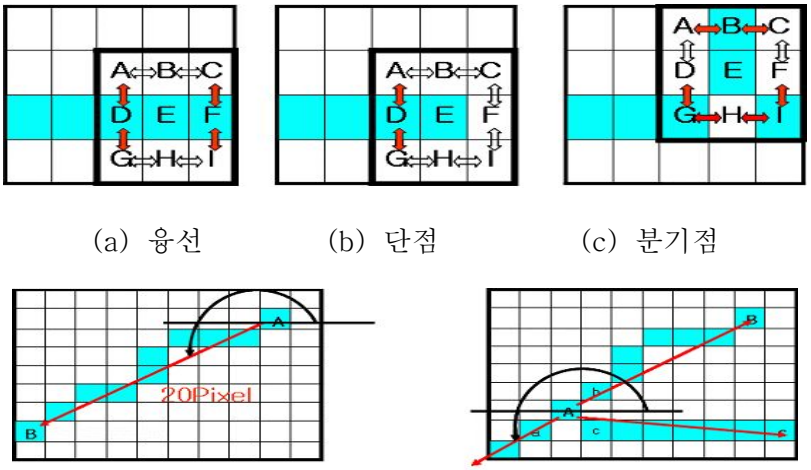


그림 9. 지문영상의 특징점

그림 10의 (a)는 융선의 연속된 선의 특징이다. 각 픽셀마다 검색을 하게 되는데 주위 픽셀 값의 변화가 4번인 경우는 연속된 선이고, 그림 10의 (b)와 같이 변화가 두 번 생기는 것이 단점이다. 단점의 각도는 그림 10의 (d)와 같이 단점의

좌표에서 20픽셀을 검색하게 된다. 이때 A 점에서 떨어진 B점까지의 각도를 구하게 된다. 그림 10의 (c)는 분기점을 나타낸 것으로 변화가 6번 일어나게 된다. 이때 분기점의 각도는 그림 10의 (e)와 같이 3개의 가지가 있기 때문에 각각 모두 20픽셀씩 탐색하게 된다. 그리고 각각의 각도를 구한 뒤 3개의 각도 중 차이가 적은 각도의 반대에 있는 가지를 기준가지로 잡고 분기점의 각도를 구하게 된다.



(a) 융선 (b) 단점 (c) 분기점

(d) 단점의 각도 (e) 분기점의 각도

그림 10. 특징점 타입과 각도 측정

추출된 특징점에는 실제 존재하지 않는 특징점인 의사 특징점이 포함되어 있다. 대표적인 의사 특징점은 지문과 배경이 만나는 경계 지점에 생기는 단점으로 이 단점은 실제 존재하는 단점이 아니라 융선이 더 이상 획득되지 않아서 생기는 단점들이다. 이 단점들은 지문배경 분리에서 만든 값을 이용하여 경계부분에서 추출된 특징점을 제거한 후 그림 11과 같이 최종 특징점을 추출한다[7-8].

그림 11. 최종 특징점

제 2 절 인식 과정

인식과정은 전처리, 특징점 추출, 그리고 정합 단계로 구성되어 있다. 등록 과정과 동일하게 지문영상의 전처리단계와 특징점 추출단계를 수행한다. 입력된 지문영상으로부터 추출된 등록 템플릿과 인증을 위해 입력된 지문 영상으로부터 추출되어 만들어진 템플릿 상의 유사도를 측정하는 과정으로 구성된다.

이렇게 템플릿을 비교하기 위해 두 템플릿이 가장 일치하는 형태로 중첩시키는 정렬과정(Alignment Stage)이 먼저 이루어지고 정렬된 두 지문 템플릿으로부터 대응하는 특징점과 기타의 정보를 찾는 정합과정(Matching Stage)을 수행하게 된다.

1. 정렬 과정

정렬 과정은 입력 템플릿과 이미 등록된 템플릿 사이의 동일한 국부영역을 찾아내어 두 지문의 특징점을 중첩될 수 있도록 만드는 과정이다. 입력 템플릿에서 몇 개의 특징점으로 구성된 국부 영역에 대한 형태적 동일성을 등록 템플릿에서 찾을 수 있다면 이 정보를 이용하여 두 지문을 정렬시킬 수 있는 보정값(Δx, Δy, $\Delta\ \theta$)을 얻을 수 있다. 국부 영역을 구성하는 특징점을 찾기 위해서는 두 개의 특징점을

이용하면 두 점을 잇는 하나의 선분을 설정할 수 있으며, 세 개의 특징점을 이용하면 면 정보를 구성할 수 있다. 이와 같이 계속적으로 확장할 수 있으나 보정값(Δx, Δy, $\Delta\theta$)을 얻어내기 위한 최소한의 정보는 두 개의 특징점이면 가능해진다. 그림 12은 클릭 구조를 이용한 정렬 과정으로 3개의 특징점을 이용한 특징점 구조의 그림이다. 3개의 특징점을 연결한 선들의 각과 선들의 길이는 지문의 회전과 이동에 무관한 특성을 가지게 되기 때문에 두 지문영상의 동일한 국부영역을 찾을 수 있다.

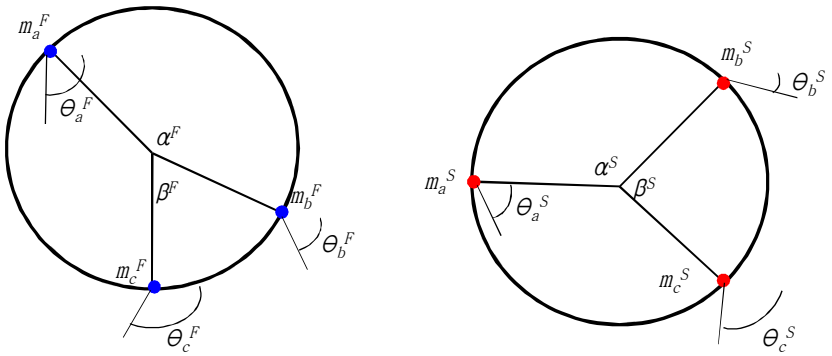
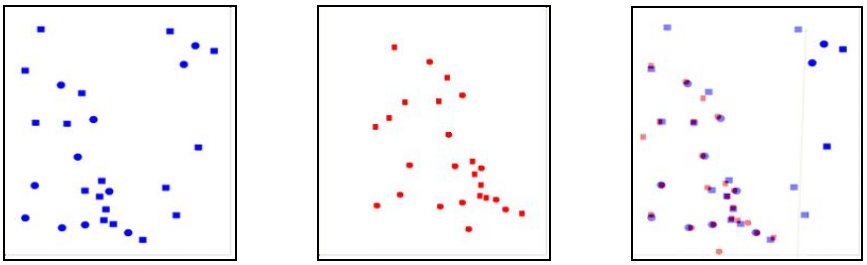


그림 12. 3개의 특징점을 이용한 특징점 구조

등록 특징점의 국부영역과 입력 특징점의 국부영역의 좌표와 회전의 차이값을 이용하여 두 특징점을 보정한다. 그림 13과 같은 형태로 두 영상을 보정할 수 있다.



(a) 등록 템플릿 (b) 입력 템플릿 (c) 중첩된 특징점 모습

그림 13. 특징점 보정(Δx : −65, Δy : 23, $\Delta\theta$: −2)

2. 정합 과정

정렬 과정을 통해 좌표 변환된 입력 템플릿은 등록 템플릿상의 특징점과 거리와 방향의 일치 정도를 판단하여 대응쌍 여부를 결정하게 된다. 이후 특징점 쌍 주변의 활용 가능한 정보를 이용하여 두 템플릿이 동일한 손가락으로부터 각기 다른 시점에 획득된 지문으로부터 생성된 것일 확률을 구하는 과정을 거쳐 지문 인증을 한다.

지문영상은 입력 과정에서 발생되는 비선형 일그러짐 때문에 동일한 위치에서 특징점이 추출되지 않는다. 그래서 특징점 정합을 수행할 때 그림 14와 같이 허용범위 안에 존재하는 특징점을 동일 특징점 후보를 선택한 후 두 특징점의 동일여부를 판단하게 된다.

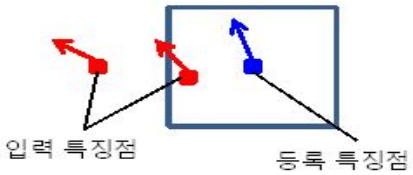


그림 14. 특징점 오차 허용범위

등록 템플릿에 존재하는 특징점을 기준으로 오차 허용 범위내에 존재하는 모든 입력 템플릿을 검색하여 대응 특징점 쌍을 찾아낸다. 그림 15와 같이 검색해 낸 대응 특징점 쌍들을 이용하여 두 지문영상의 유사도를 측정한다.

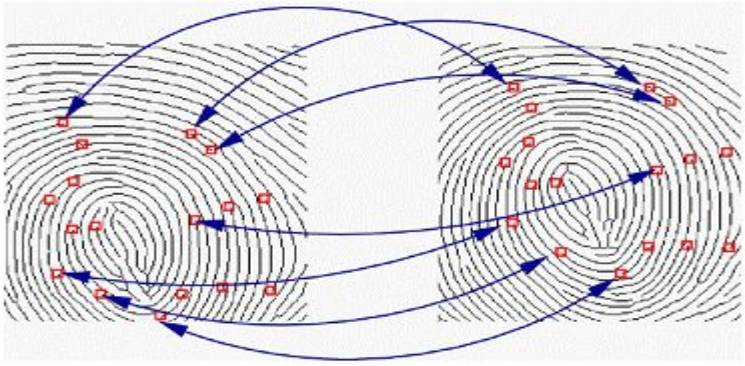


그림 15. 특징점 대조

대응 특징점 쌍들을 구한 후 각 대응 특징점의 좌표, 각도, 타입의 정보를 이용하여 두 특징점의 동일 특징점 여부를 측정한다. 가장 간단한 유사도 계산 방법은 식 1과 같이 동일 특징점 개수를 이용하여 두 지문 영상의 유사도 S를 계산하다. 이렇게 계산된 S를 이용하여 지문인증을 수행한다.

$$S = \frac{2 \times n^M}{n^S + n^F} \qquad (1)$$

n^M : 두 지문 사이의 동일 특징점 쌍의 개수

n^S : 입력 지문영상의 특징점 개수

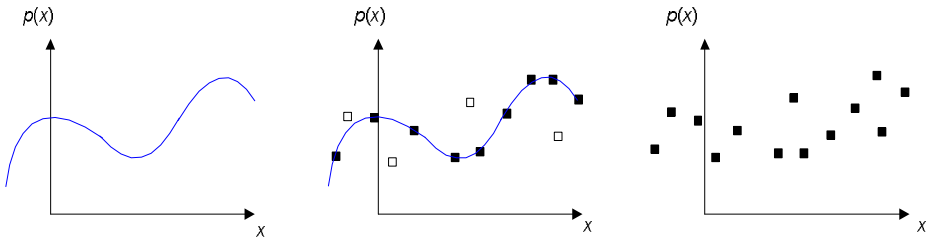
n^F : 등록 지문영상의 특징점 개수

제 3 장 지문 퍼지볼트

본 장에서는 퍼지볼트 이론과 기하학적 해싱을 이용하여 지문 템플릿을 보호하는 기하학적 해싱 기반 퍼지볼트 시스템을 분석한다.

제 1 절 퍼지볼트

Juels와 Sudan는 “퍼지볼트” 라는 암호이론을 통하여 중요한 정보를 보호할 수 있는 방법을 제안하였다. 이 방법은 두 집합 A와 B를 key로 이용하여 secret κ를 암호/복호화한다. 퍼지볼트는 인코딩(Encoding)과 디코딩(Decoding)과정으로 이루어진다. 설명하기 위해, 집합 A가 가지고 있는 secret κ를 lock하려는 쪽을 Alice라고 하고, 집합 B가 가지고 있는 κ을 unlock하려는 쪽을 Bob이라고 한다 [10-12].



(a) 다항식 생성 (b) 거짓 점 삽입 (c) 볼트 생성

그림 16. 퍼지볼트의 볼트 생성 과정

Alice는 t개의 점들로 구성된 집합 A로 비밀정보 κ을 은닉하는 것을 목표로 한다. Alice는 그림 16의 (a)와 같이 κ을 이용하여 단일 변수 x만 있는 k차 다항식 $p(x)$를 생성한다. 이때, $t > k$ 이다. 그리고 집합 A의 원소들을 다항식의 변수 x에 각각 대입하여 다항식 $p(x)$ 위에 존재하는 점들을 생성한다. 이것은 결국 사용자의 집합 A의 점들을 다항식 $p(x)$ 상에 있는 점들로 사영하는 것으로 간주할 수

있다. 결과적으로 (l_i , $p(l_i)$)로 구성된 집합을 생성한다. 그림 16의 (b)와 같이 집합 L로부터 생성된 점(검정색)들을 은닉하기 위해서 $p(x)$에 존재하지 않는 다수의 임의의 점(흰색)들을 생성하여 (l_i, $p(l_i)$)에 추가한 집합 R 또는 볼트(Vault)를 그림 17과 같이 생성한다. 여기서, 집합 A는 집합 R이라는 금고(Vault)내에 존재하는 다항식 $p(x)$ 즉, κ에 접근하기 위한 금고의 열쇠 역할을 한다. 그리고 $p(x)$와 임의의 점들(Chaff point)은 공격자로부터 다항식 $p(x)$를 은닉함으로써 이 알고리즘의 보안성을 제공한다.

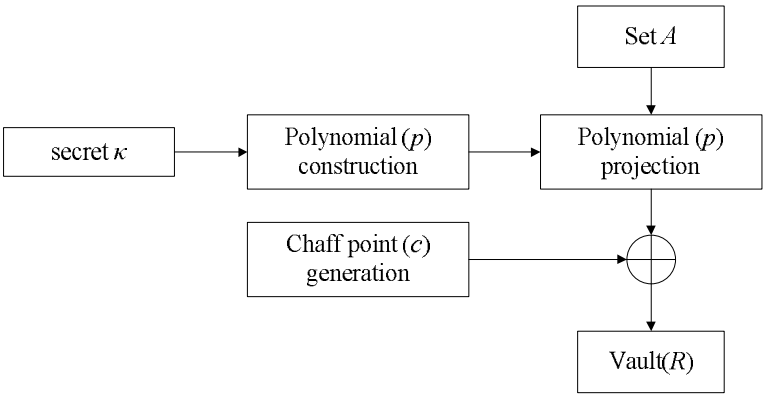


그림 17. 퍼지볼트의 인코딩 과정

이번에는 Bob이 그림 18과 같이 집합 B를 이용하여 금고에 있는 κ를 얻고자 한다. 만약 B의 원소들과 A의 원소들 사이에 일치하는 부분이 많다면, B는 R에서 다항식 $p(x)$ 위에 존재하는 대부분의 점들을 식별해 내게 된다. 그러나, 집합 B를 이용해 찾아낸 점들에는 다항식과 무관한 임의의 점들도 포함되어 있을 수 있다. 이는 랜덤로이즈(Random-noise)로 간주하여 오류정정부호(Error-correcting code)중에서 리드-솔로몬 부호(Reed Solomon code)를 사용하여 정정되고 Bob은 정확하게 κ를 얻어내게 된다. 반면에 B와 A가 거의 일치하지 않는다면 $p(x)$를 알아낼 수 없도록 방해하는 많은 점들(Chaff point) 때문에 Bob은 κ에 접근할 수 없다. 퍼지볼트 알고리즘은 집합 B가 집합 A와 항상 정확하게 일치하지 않더라도 오류정정부호를 사용하기 때문에 열쇠가 갖는 모호성과 암호화 구조가 요구하

는 정확성(Correctness)사이의 틈을 줄여줄 수 있다. 또한 집합 A와 비밀정보 κ는 다항식과 무관한 점들에 의해 동시에 보호된다. 결국 이 알고리즘은 바이오 정보와 결합되어 바이오 정보와 사용자의 중요한 정보를 보호하기 적합하다. 이러한 이유로 최근에 퍼지볼트와 바이오 정보를 결합하는 방법에 대한 연구가 활발하게 진행되고 있다.

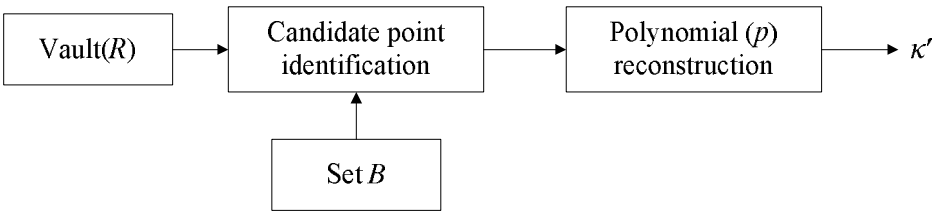


그림 18. 퍼지볼트의 디코딩 과정

Clancy [13] 등은 Jules와 Sudan의 퍼지볼트를 기반으로 지문 퍼지볼트(Fuzzy fingerprint vault)를 제안하였다. 그들이 사용한 지문의 특징은 지문영상에서 융선의 끝점과 분기점의 위치 값이다. 그들은 정규화된 특징점의 위치를 찾아내기 위해 동일인의 한 손가락에서 5장의 지문영상을 취득하여 가우시안 잡음 모델(Gaussian noise model)을 이용하였다. 그러나 그들의 시스템은 금고를 잠글 때 사용한 지문과 금고를 열기 위한 지문들을 모두 사전에 정합한다는 가정을 한다. 이는 실제로 지문인식 기반의 시스템을 구현하는 비현실적이다. 그리고 이들은 실제로 시스템을 구현한 것이 아니라 모의 실험을 통해 파라미터를 결정하고 시스템의 성능에 대한 결과를 얻었다. FRR(False Reject Rate)이 20~30%가 될 때 공격자가 2^{69} 번 시도해야 금고를 열수 있다고 한다. 즉, FAR(False Accept Rate)은 $2^{-69} \approx 1.7 \times 10^{-21}$이 된다.

이후에 Uludag, Pankanti와 Jain은 지문을 퍼지볼트에 적용하는 다른 방법을 제안하였다. 이들은 정규화된 특징점의 위치를 선택하기 위해 양자화 방법을 사용하였으며 Jules와 Sudan의 퍼지볼트에서 사용한 리드-솔로몬 부호대신 CRC 부호(Cyclic Redundancy Check Code)을 이용하여 오류를 정정하였다[14]. 이

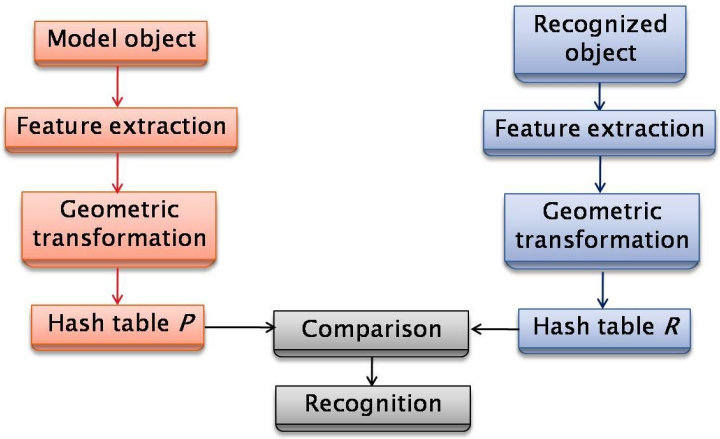
들은 이러한 결과를 바탕으로 다항식 위에 존재하는 점들 중 9개만 일치하면 다항식을 재생성하여 κ를 얻어내도록 설계하였다. 이 방법은 한 지문에서 추출된 여러 특징점들 중에서 템플릿의 특징점과 일치하는 9개의 점을 조합하는 연산을 통해 찾아내므로 시간이 오래 걸리는 단점이 있다. 그리고 동일하게 나오는 특징점의 개수에 따라 다항식의 차수가 결정되므로 파라미터 값을 선택하는데 한계가 있다.

제 2 절 기하학적 해싱 기법

컴퓨터 비전에서 객체는 객체의 특징들(Feature)의 집합으로 나타내는데, 객체의 특징들에는 점(Point), 선(Line), 코너(Corner), 에지(Edge) 등이 있다. 객체 인식이란 입력된 영상 장면에서 관심 있는 객체를 인식하는 것을 말한다. 즉, 데이터베이스에 관심 있는 객체인 모델의 특징을 저장한 후, 데이터베이스의 모델을 검색함으로써 입력 영상 장면에서 관심 있는 객체를 인식하는 것이다. 이와 같은 객체 인식 시스템을 모델 기반 인식 시스템이라 한다.

대부분의 모델 기반 인식 시스템은 입력 영상 객체의 특징과 데이터베이스에 저장된 하나의 모델의 특징과 같다는 가정에 기초하여, 가정을 검증하게 된다. 이와 같은 과정은 데이터베이스에 저장된 모든 모델에 대하여 전체적으로 수행된다. 따라서 처리 시간은 데이터베이스에 저장된 모델의 수에 비례하게 되고 모델의 수가 증가할수록 처리 시간은 오래 걸리게 된다. 이와 같은 문제점을 해결하기 위하여 Lamdan과 Wolfson은 색인(Index)에 기초한 기하학적 해싱 알고리즘을 객체 인식에 도입하였다[15].

기하학적 해싱 알고리즘은 객체를 나타내는 기하학적인 모양은 변화하지 않는다는 사항을 고려하여 색인 함수를 생성하고 이를 이용하여 데이터베이스를 효율적으로 검색하는 알고리즘이다. 모델 기반 인식 시스템에 기하학적 해싱 알고리즘을 이용하면, 모델의 개수가 증가하여도 색인 함수의 성능이 좋다면 처리 시간은 거의 영향을 받지 않는다.



(a) 전처리 과정 (b) 인식 과정

그림 19. 기하학적 해싱의 블록도

그림 19와 같이 기하하적 해싱 알고리즘은 전처리(Preprocessing)와 인식(Recognition) 과정으로 구성된다. 전처리 과정은 오프라인으로 단 한번만 수행된다. 우선, 객체의 점으로 표현된 특징점은 데이터베이스에 저장되기 위하여 기하학적 변환에 의해 많은 정보를 포함한 해시 테이블 형태로 저장된다. 전처리 과정은 다음의 절차를 따른다.

전처리 과정

① 모델에서 특징점을 추출한다.

② 기준이 되는 특징점을 선택한다.

③ 다른 특징점과 함께 회전, 이동의 기하학적 변환을 하여 기준점을 좌표축의 중심으로 이동시킨다.

④ 기준점과 다른 특징점의 정보를 포함한 해시 테이블(Hash table)을 생성한다.

⑤ 모든 특징점에 대하여 위 과정을 수행한다.

예를 들어, 그림 20과 같이 객체의 특징점 5개가 추출되었다면, 한 개의 특징점을 기준점으로 선택하여 쌍이 되는 다른 특징점과 함께 회전이나 이동의 기하학적 변환을 이용하여 좌표축의 중심으로 이동시킨다. 그리고 나머지 특징점들을 같은 방법으로 변환시키고, 변형된 좌표에는 해시 테이블의 bin이라고 정의된 방법으로 표시한다. 이렇게 5개의 모든 특징점에 대하여 기준점을 선택하고 나머지 특징점들을 같은 방법으로 변환시킨다.

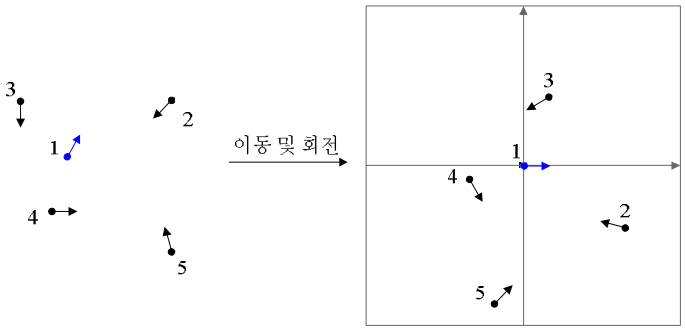


그림 20. 해시 테이블 생성(기준점 1)

온라인으로 수행되는 인식 과정은 다음의 절차를 따른다.

인식 과정

① 모델에서 관심 있는 특징점을 추출한다.

② 기준점을 선택하고, 남은 특징점과 함께 기하학적 변환을 수행한다.

③ 생성된 해시 테이블과 전처리 과정의 해시 테이블을 비교하여 bin의 수를 올려준다.

④ 다른 기준점을 선택하고 위와 같은 과정을 수행한다.

⑤ 최종적으로 가장 높은 점수를 가진 bin을 가장 유사한 객체라 인식

인식하려는 객체의 특징점 s개를 추출하고, 임의의 특징점을 기준점으로 선택

하여 쌍이 되는 다른 특징점과 함께 회전이나 이동의 기하학적 변환을 이용하여 좌표축의 중심으로 이동시킨다. 그리고 나머지 특징점들을 같은 방법으로 변환시킨다. 이렇게 생성된 테이블은 전처리 과정에서 생성된 해시 테이블과 비교되어 bin의 수를 올려준다. 다시 n개의 모든 특징에 대하여 기준점을 선택하고 위와 같이 나머지 특징점들을 같은 방법으로 변환시킨 후, 최종적으로 가장 높은 점수를 가진 bin이 가장 유사한 객체라고 인식한다[15-16].

제 3 절 기하학적 해싱 기반 지문 퍼지볼트

Clancy[13]과 Uludag[14]는 퍼지볼트를 이용하여 지문인식을 수행할 수 있는 지문 퍼지볼트 이론을 제안하였다. 지문 정렬은 지문 특징점들이 서로 이루는 구조적 정보를 이용해서 수행된다. 하지만 지문 퍼지볼트는 지문정보에 수백 개의 거짓 정보가 추가된다. 지문 템플릿으로부터 지문 특징점과 추가된 거짓 특징점을 구분하는 것이 어렵다. 따라서 정확한 구조적 정보를 이용할 수 없어 정렬과정을 수행하는 데 어려움이 있다. 지문인식의 중간단계인 정렬 과정이 수행되었다고 가정하거나, 수동으로 정렬하였기 때문에 현실적이지 못하며 자동화된 시스템을 구현할 수 없다는 문제점이 있다. 이러한 문제점을 해결하기 위해 Chung 등은 기하학적 해싱 기법을 적용하여 자동으로 정렬을 수행하였다[17-18].

지문 템플릿이 공격 당할 경우 사용자의 지문정보가 도용될 수 있는데 이 문제를 해결하기 위해 지문 퍼지볼트 시스템은 지문 템플릿에 다수의 거짓 특징점을 생성하여 삽입한다. 이렇게 지문 템플릿은 복원되지 않은 상태에서 사용자의 지문 인식을 수행할 수 있고, 보호된 지문 템플릿은 도난 당하여도 공격자는 거짓 특징점이 추가된 지문 템플릿으로부터 사용자의 지문 특징점을 분리할 수 없게 된다.

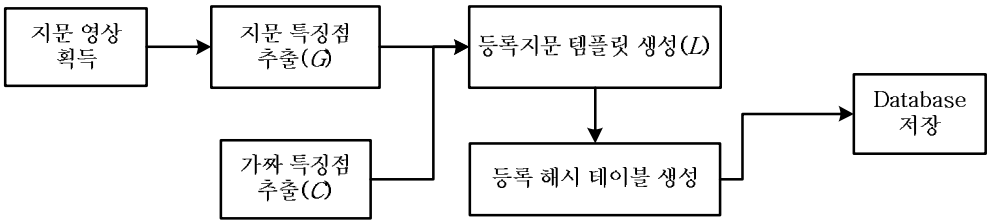
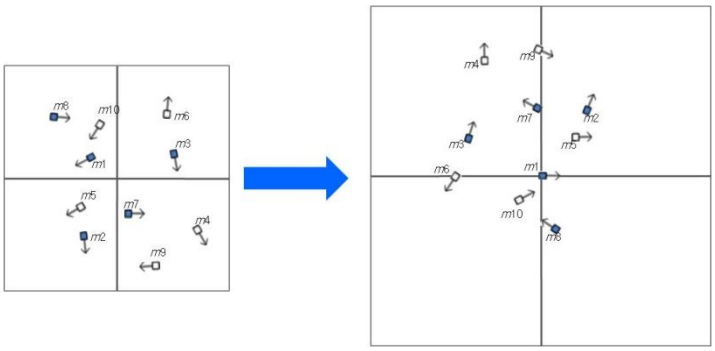


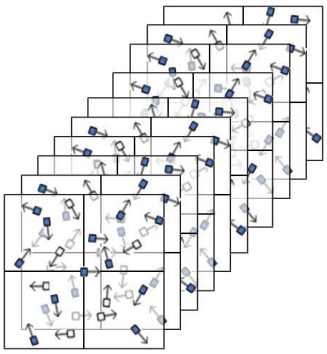
그림 21. 기하학적 해싱 기반 지문 퍼지볼트의 등록과정

기하학적 해싱 기법은 지문 템플릿을 구성하는 사용자의 특징점과 거짓 특징점 모두를 기준점으로 선택하여 자동 정렬하는 방법이다. 기하학적 해싱 기반 지문 퍼지볼트의 등록과정은 그림 21과 같다. 지문의 특징점을 추출하고 거짓 특징점을 추가하여 특징점의 기하학적인 특성을 고려한 등록 테이블을 생성한다. 거짓 특징점과 사용자의 지문 특징점이 포함된 등록 템플릿(라킹셋)을 $L = \{m_i \mid 1 \le i \le r\}$으로 표현할 수 있다. L에서 사용자의 지문 특징점과 거짓 특징점들의 집합은 각각 $G = \{m_i \mid 1 \le i \le n\}$, $C = \{m_i \mid n+1 \le i \le r\}$으로 표현한다. 등록 템플릿은 기하학적 해싱 기법을 통해 등록 해시 테이블을 생성한다. 사용자의 지문 특징점 G와 거짓 특징점 C의 모든 특징점에 대하여 기준점을 선택하고 선택된 기준점을 제외한 나머지 특징점들의 회전 및 이동 변환을 거쳐 해시 테이블이 생성된다. 해시 테이블은 기하학적 변환으로 생성된 변환 특징점으로 구성된다.

지문 템플릿이 그림 22의 (a)와 같이 지문 특징점(검정색, 5개)와 거짓 특징점(흰색, 5개) 총 10개의 특징점으로 구성되어 있다고 가정하면, m_1을 기준점으로 한 해시 테이블이 그림 22의 (b)와 같이 생성된다. m_1부터 m_{10}까지 기하학적 변환을 수행하면 그림 22의 (c)와 같이 등록 해시 테이블을 생성시킬 수 있다.



(a) 지문 템플릿 (b) 해시 테이블 생성(m_1 기준점)



(c) 등록 해시 테이블

그림 22. 등록 해시 테이블 생성 과정

인식 과정은 그림 23과 같이 특징점 정보 추출 단계, 테이블 생성 단계, 정합 단계, 그리고 인증 단계(후보자 목록 생성 단계)를 순차적으로 수행한다. 특징점 정보 추출 단계, 테이블 생성 단계는 등록 과정과 동일한 순서 및 방법으로 수행한다. 정합 단계는 변환 특징점 집합을 기본 단위로 하여 위치와 각도가 일치하는 특징점 쌍을 구하는 단계이다. 인식 테이블의 변환 특징점들 집합과 지문 데이터베이스에 저장된 등록 테이블의 변환 특징점들 집합 사이에 일치하는 변환 특징점들의 쌍의 수를 구함으로써 측정이 가능하다. 이렇게 만들어진 특징점 셋은 언라킹셋 U가 된다[19].

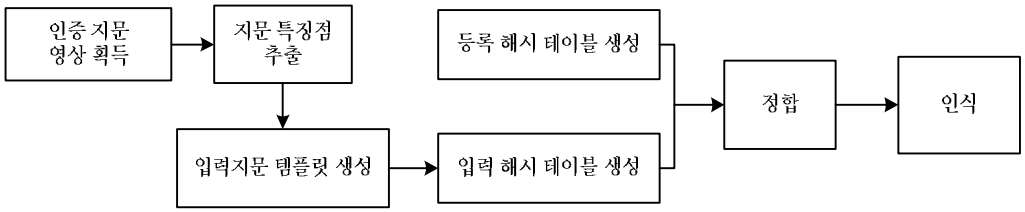


그림 23. 기하학적 해싱 기반 지문 퍼지볼트의 인식과정

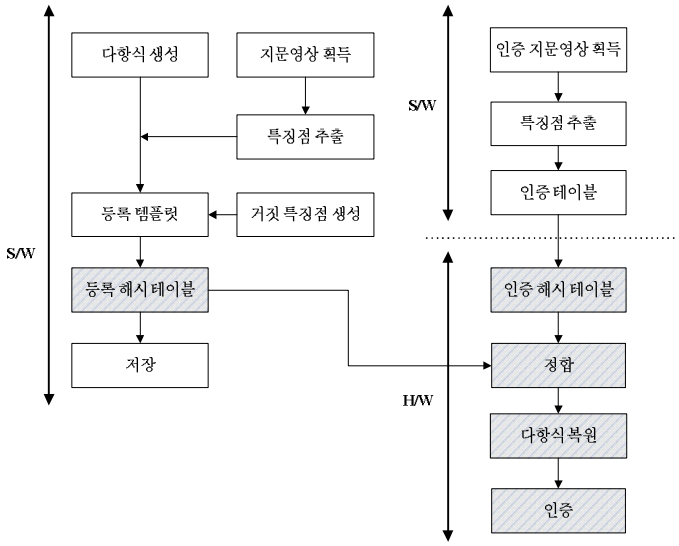
인증 단계는 인식 테이블의 모든 변환 특징점들 집합에 대하여 지문 데이터베이스에 저장된 등록 테이블의 모든 변환 특징점들 집합과 비교하여 유사도를 측정하고, 높은 유사도 순으로 후보자 목록을 구한다

제 4 장 지문 퍼지볼트 시스템의 하드웨어 구현

본 장에서는 기하학적 해싱 기반 지문 퍼지볼트 시스템의 수행시간 단축을 위해 지문 퍼지볼트 시스템의 하드웨어 구조에 대해 제안한다. 또한, 시스템의 고속 처리를 위해 병렬처리 기법 하드웨어 구조를 제안한다.

제 1 절 하드웨어 구조

지문 퍼지볼트의 하드웨어 시스템 구현을 위해 그림 24와 같이 소프트웨어와 하드웨어 모듈을 통합하여 구현한다. 지문 퍼지볼트 시스템의 등록 과정은 지문 특징점 추출, 거짓 특징점 생성, 지문 템플릿 생성, 지문 해시 테이블 생성, 그리고 지문 데이터베이스 저장 단계로 이루어진다. 인식 과정은 입력 지문의 특징점 추출, 지문 해시 테이블 생성, 정합, 다항식 복원, 그리고 인증 단계로 이루어진다.



(a) 등록과정 (b) 인식과정

그림 24. 지문 퍼지볼트 시스템의 소프트웨어와 하드웨어 통합 구현

인식 과정의 정합 단계, 다항식 복원단계, 그리고 인증 단계는 하드웨어로 구현한다. 정합 단계는 등록 과정에서 거짓 특징점의 수가 증가될수록 인식 과정에서의 정합단계의 연산량이 많아지므로 하드웨어로 구현하는 것이 바람직하다. 반면에, 등록 과정과 인식 과정의 입력 지문 해시 테이블 생성단계 등은 소프트웨어로 구현한다[20-22].

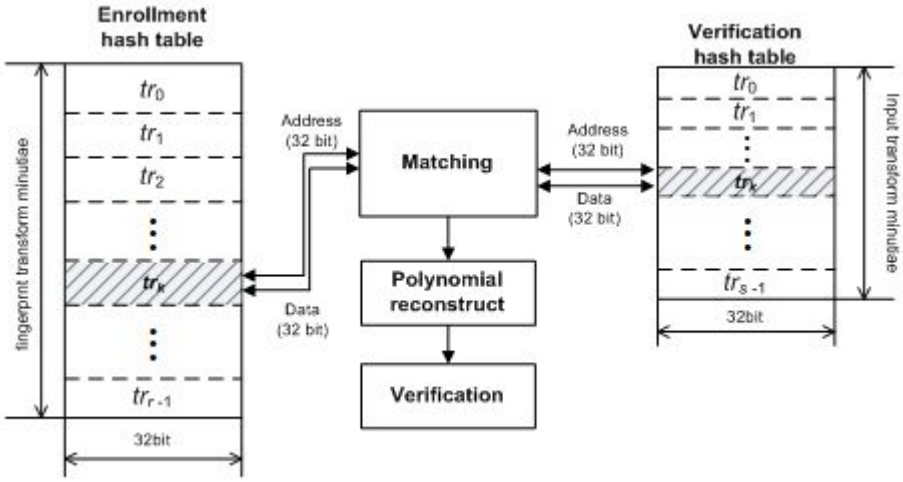


그림 25. 제안하는 하드웨어 모듈의 구조도

제안하는 하드웨어 모듈은 그림 25와 같이 등록 해시 테이블과 인증 해시 테이블을 각각 저장하는 메모리, 매칭모듈, 다항식 복원모듈, 그리고 인증모듈로 구성된다. 등록 해시 테이블을 $E = \{tr_i \mid 0 \le i \le r-1\}$으로 표현할 수 있다. r은 등록 지문 템플릿의 수이다. 여기서 tr_i는 지문 템플릿 중 m_i을 기준으로 선택하고 기하학적 해싱을 통해 생생된 해시 테이블이며 $r-1$개의 변환 특징점으로 이루어져 있다. 예를 들어, 5개의 지문 특징점과 10개의 거짓 특징점으로 구성된 지문 템플릿이 있을 때, 등록 지문 템플릿의 수는 15이다. 기하학적 해싱 기법에서 기준점을 15번 선택하기 때문에 등록 해시 테이블은 15개의 tr을 가진다. tr_1은 특징점 m_1을 기준점으로 하여 기하학적 해싱을 수행한 14개의 변환 특징점을 가진다[23-24].

인증 해시 테이블을 $V = \{tr_j \mid 0 \le j \le s-1\}$으로 표현할 수 있다. 여기서 s는 인증 지문 템플릿의 수이다. 즉, 인증 해시 테이블은 s개의 tr로 구성되며, 하나의 tr에는 $s-1$개의 변환 특징점을 가진다. 인증 지문의 템플릿이 12개라고 가정하면, 기준점을 12번 선택하므로 12개의 tr로 구성되며 하나의 tr에는 기준점을 제외한 11개의 변환 특징점이 있다. 변환 특징점은 그림 26과 같이 x축과 y축 각각 11비트, 각도 9비트, 그리고 종류 1비트 총 32비트로 이루어진다.

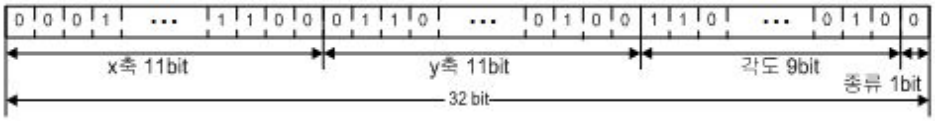


그림 26. 변환 특징점의 구조

해시 테이블을 저장하는 메모리의 크기는 각각의 총 변환 특징점 수이다. 등록 해시 테이블의 메모리 크기 p는 $r-1$개의 tr이 r개 있으므로 $r \times (r-1)$, 인증 해시 테이블의 메모리 크기 q는 $s-1$개의 tr이 s개 있으므로 $s \times (s-1)$이다.

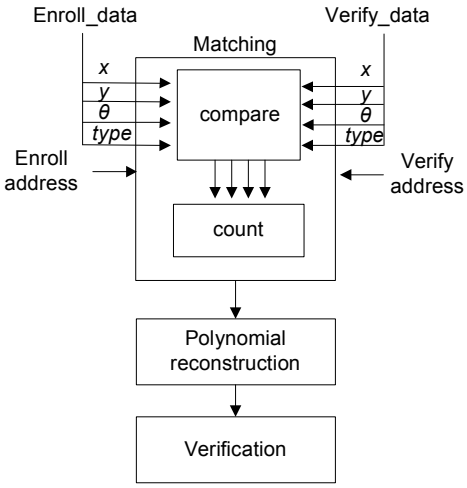


그림 27. 제안하는 하드웨어 모듈의 구조도

매칭모듈, 다항식 복원모듈, 그리고 인증모듈의 구조는 그림 27과 같다. 매칭모듈은 compare모듈과 count모듈로 구성된다. Compare모듈은 등록 지문 변환 특징점과 인증 지문 변환 특징점을 비교하는 모듈이고, count모듈은 일치된 변환 특징점의 수를 계산하는 모듈이다. 다항식 복원모듈은 일치된 변환 특징점을 이용하여 다항식의 계수를 복원하는 모듈이다. 인증모듈은 복원된 다항식의 계수와 등록 과정에서의 다항식의 계수를 비교하는 모듈이다.

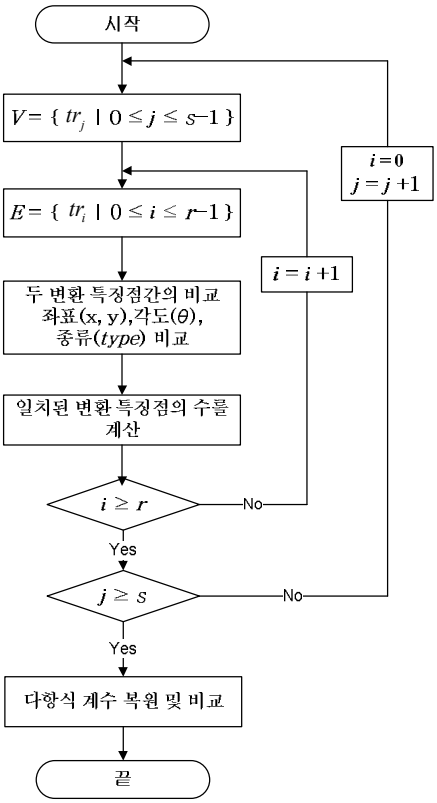


그림 28. 제안하는 하드웨어 모듈의 흐름도

제안하는 하드웨어 구조의 흐름도는 그림 28과 같다. 등록 해시 테이블 E의 크기는 p이며 r개의 tr 로, 인증 해시 테이블 V의 크기 q이며 s개의 tr 로 구성된다고 가정한다. 먼저, 인증 해시 테이블의 tr_0 이, 등록 해시 테이블의 tr_0 이 매칭모듈에 입력된다. 먼저, 매칭모듈의 compare모듈에서 인증 해시 테이블의 tr_0

과 등록 해시 테이블의 tr_0이 비교된다. 입력된 두 tr 의 변환 특징점들의 좌표, 각도, 종류를 비교하여 일치여부를 count모듈로 전송한다. count모듈은 일치된 변환 특징점의 수를 계산한다. 이 후, 등록 해시 테이블의 tr_1이 매칭모듈에 입력되어 인증 해시 테이블의 tr_0과 비교된다. 동일한 방법으로 등록 해시 테이블의 모든 tr 의 비교가 완료되면, 인증 해시 테이블의 tr_1이 매칭모듈에 입력된다. 이를 반복적으로 수행하여 인증 해시 테이블의 tr_{s-1}까지 비교한다.

다항식 복원모듈은 다항식의 계수를 복원한다. 등록 과정에서 생성된 다항식의 차수와 일치된 변환 특징점들을 이용하여 다항식의 계수를 복원시킨다. 먼저, 매칭된 변환 특징점과 등록과정에서의 생성된 볼트를 입력 받아 연립 방정식을 형성한다. 이 후에, Gauss-Jordan 소거법을 이용하여 연립 방정식의 해를 구해 복원된 다항식의 계수를 구한다. 인증모듈은 등록 과정의 다항식의 계수와 다항식 복원모듈에서 복원된 다항식의 계수를 비교한다

Procedure : 제안하는 하드웨어의 정합 방법

Step 1. 매칭모듈에 변환 특징점 전송.
- 등록 해시 테이블의 tr과 인증 해시 테이블의 tr을 매칭모듈에 전송

Step 2. 변환 특징점간의 비교.
- 두 변환 특징점의 좌표(x, y), 각도(θ), 종류($type$) 비교
- 일치되는 변환 특징점 수 계산

Step 3. 등록 변환 특징점의 전송
- 비교 수행 후, 등록 해시 테이블의 다음 tr을 매칭모듈에 전송
- Step2를 r번(등록 해시 테이블의tr 의 수) 반복수행

Step 4. 인증 변환 특징점의 전송
- 모든 등록 해시 테이블의 tr의 비교가 완료되면, 인증 해시 테이블의 다음을 매칭모듈에 전송.
- Step3을 s번(인증 해시 테이블의 tr의 수) 반복수행

Step 5. 다항식 복원
- 일치된 변환 특징점과 다항식의 차수를 이용하여 수행
- 다항식의 계수 복원

Step 6. 인증
- 등록 과정에서 생성된 다항식의 계수와 복원된 다항식 계수 비교

그림 29. 제안하는 하드웨어 모듈의 동작 과정

예를 들어, 등록 지문 템플릿의 수가 219개(19개의 지문 특징점과 200개의 거짓 특징점), 인증 지문 템플릿의 수가 10개라고 가정한다. 등록 해시 테이블의 tr 수는 219, 인증 해시 테이블은 10개의 tr 로 구성된다. 인증 해시 테이블의 tr_0 과 등록 해시 테이블의 tr_0 이 매칭모듈에 입력된다. 매칭모듈에서 등록 해시 테이블의 tr_0 과 인증 해시 테이블의 tr_0 이 비교된 후, 일치된 변환 특징점의 수를 계산한다.

이 후에, 등록 해시 테이블의 tr_1 이 매칭모듈에 입력되어 인증 해시 테이블의 tr_0 과 비교된다. 두 tr 의 일치된 변환 특징점을 계산한다. 동일한 방법으로 등록 해시 테이블의 tr_{218} 이 매칭모듈에 입력되어 인증 해시 테이블의 tr_0 와 비교된다. 이 후에, 인증 해시 테이블의 tr_1 이 매칭모듈에 입력되어 다시 등록 해시 테이블의 tr_0 이 입력되어 비교된다. 이를 반복적으로 수행하여 인증 해시 테이블의 tr_9 까지 비교된다. 매칭된 변환 특징점과 등록 과정에서의 다항식 차수를 이용하여 다항식의 계수를 복원시켜 등록 과정의 다항식 계수와 비교한다.

제 2 절 병렬처리 기법을 적용한 하드웨어 구조

본 논문에서는 제안한 하드웨어 구조의 고속처리를 위해서 병렬처리 기법을 적용한 하드웨어 구조를 제안한다. 소프트웨어와 하드웨어 모듈의 구분은 앞서 제안한 하드웨어 구조와 동일하다. 제안하는 병렬처리 기법 하드웨어 모듈은 등록 해시 테이블과 인증 해시 테이블을 각각 저장하는 메모리, 매칭모듈, 다항식 복원모듈, 그리고 인증모듈로 구성된다. 병렬처리를 위해 하드웨어 모듈의 매칭모듈의 수를 2^n $(1 \le n \le 6)$개로 구성한다.

매칭모듈의 구조는 그림 30과 같이 앞서 제안한 하드웨어 구조와 비슷하지만, 입력된 등록 지문 변환 특징점이 비교 후 다시 출력된다는 점이 다르다. 다항식 복원모듈과 인증모듈의 구조는 앞서 제안한 하드웨어 구조와 동일하다.

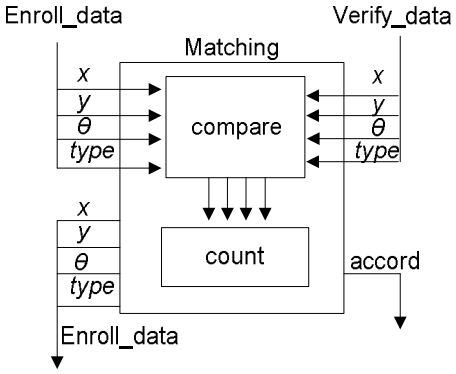


그림 30. 제안하는 병렬처리 기법 하드웨어의 매칭모듈 구조도

먼저, 2개의 매칭모듈로 구성된 하드웨어 구조는 그림 31과 같으며 흐름도는 그림 32와 같다. 등록 지문 템플릿의 수가 r개, 인증 지문 템플릿의 수가 s개일 때, 등록 해시 테이블 E의 변환 특징점의 수가 p개, 인증 해시 테이블 V의 변환 특징점이 q개라고 가정한다. 인증 지문 해시 테이블은 s개의 tr 로 구성되며 하나의 tr 에는 $s-1$개의 변환 특징점들이 있다.

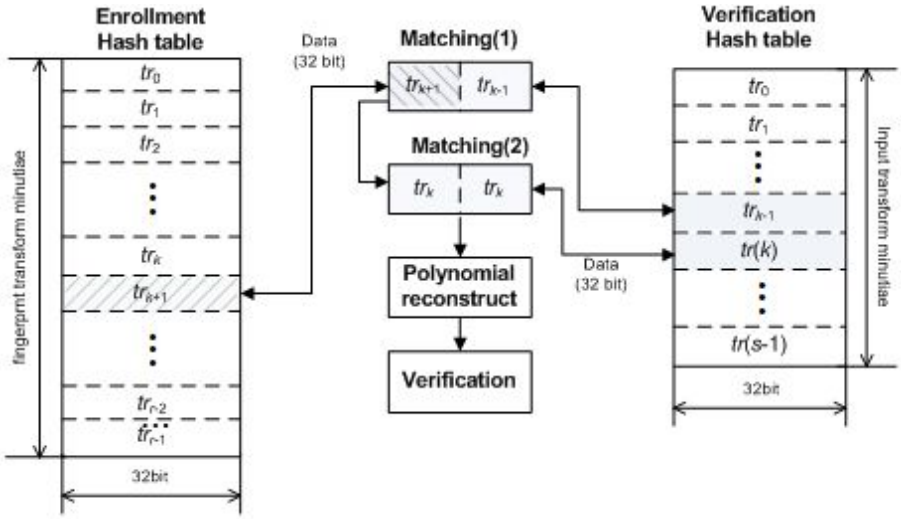


그림 31. 2개의 매칭모듈을 가진 하드웨어 모듈의 구조도

인증 해시 테이블의 tr_0 과 tr_1 이 각각 매칭모듈(1)과 매칭모듈(2)에 입력된다. 여기서, 매칭모듈(1)과 매칭모듈(2)는 그림 31의 Matching(1)과 Matching(2)를 의미한다. 등록 해시 테이블의 tr_0 이 매칭모듈(1)에 입력된다. 매칭모듈의 compare모듈에서 입력된 두 tr 의 변환 특징점들이 비교된다. 변환 특징점들의 일치여부를 count모듈에 전송하여 일치된 변환 특징점의 수를 계산한다.

이 후에, 등록 해시 테이블의 tr_0 은 매칭모듈(1)에서 매칭모듈(2)로 전송되고, 등록 해시 테이블의 tr_1 은 등록 해시 테이블에서 매칭모듈(1)로 전송된다. 매칭모듈(1)은 등록 해시 테이블의 tr_1 과 인증 해시 테이블의 tr_0 을, 매칭모듈(2)은 등록 해시 테이블의 tr_0 과 인증 해시 테이블의 tr_1 을 동시에 비교한다. 변환 특징점들의 일치여부를 판단한 후에 등록 해시 테이블의 tr 은 다음 매칭모듈로 전송된다. 즉, 등록 해시 테이블의 tr_1 은 매칭모듈(1)에서 매칭모듈(2)로, 등록 해시 테이블의 tr_2 는 등록 해시 테이블에서 매칭모듈(1)으로 전송된다.

위의 동일한 방법으로 등록 해시 테이블의 tr_{r-1} 이 매칭모듈(2)에서 인증 해시 테이블의 tr_1 과 비교가 이루어질 때까지 수행한다. 모든 등록 지문 변환 특징점이 비교가 완료되면, 인증 해시 테이블의 tr_2 와 tr_3 이 매칭모듈(1)과 매칭모듈(2)에 입력된다. 다시 등록 해시 테이블의 tr_0 이 매칭모듈(1)에 전송되면서 위의 과정을 수행한다. 인증 해시 테이블의 tr_{s-1} 이 매칭모듈(2)에 입력되어 등록 해시 테이블의 tr_{r-1} 과 비교될 때까지 반복한다.

예를 들어, 등록 지문 템플릿의 수가 213(13개의 지문 특징점과 200개의 거짓 특징점)개, 인증 지문 템플릿이 12개라고 가정한다. 등록 해시 테이블의 tr 수는 213개, 인증 해시 테이블은 12개 tr 로 구성된다.

인증 해시 테이블의 tr_0 과 tr_1 이 각각 매칭모듈(1)과 매칭모듈(2)에 입력된다. 등록 해시 테이블의 tr_0 이 매칭모듈(1)에 입력되어 인증 해시 테이블의 tr_0 과 비교된다. 일치된 변환 특징점의 수를 계산한 후에, 등록 해시 테이블의 tr_0 은 매칭모듈(1)에서 매칭모듈(2)로, 등록 해시 테이블의 tr_1 은 매칭모듈(1)로 전송된다. 변환 특징점의 일치여부를 판단한다. 동일한 방법으로 등록 해시 테이블의 tr_{212} 이 매칭모듈(2)에서 tr_1 과 비교가 이루어질 때까지 수행되며, 이 후에 인증 해시 테이

블의 tr_2와 tr_3이 매칭모듈(1)과 매칭모듈(2)에 입력된다. 인증 해시 테이블의 tr_{11}이 매칭모듈에 입력되어 등록 해시 테이블의 tr_{212}와 비교될 때까지 반복한다.

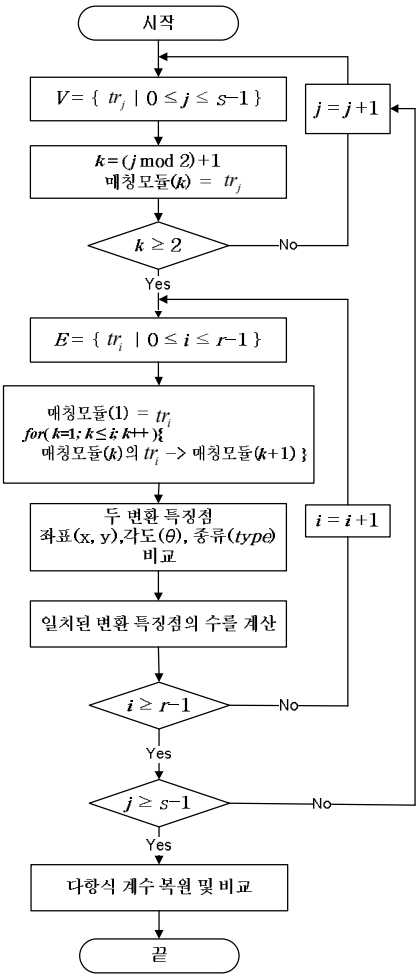


그림 32. 2개의 매칭모듈을 가진 하드웨어의 흐름도

4개의 매칭모듈로 구성된 하드웨어 구조는 그림 33과 같으며 흐름도는 그림 34와 같다. 이는 2개의 매칭모듈을 가진 하드웨어 모듈과 동작 과정이 비슷하지만, 매칭모듈이 4개로 구성되어 있어 4개의 tr 의 쌍이 동시에 비교된다는 점이 다르다. 동일하게 등록 지문의 템플릿의 수가 r개, 인증 지문의 템플릿의 수가 s개일 때, 등록 해시 테이블 E의 변환 특징점의 수가 p개, 인증 해시 테이블 V의 변환

특징점이 q개라고 가정한다.

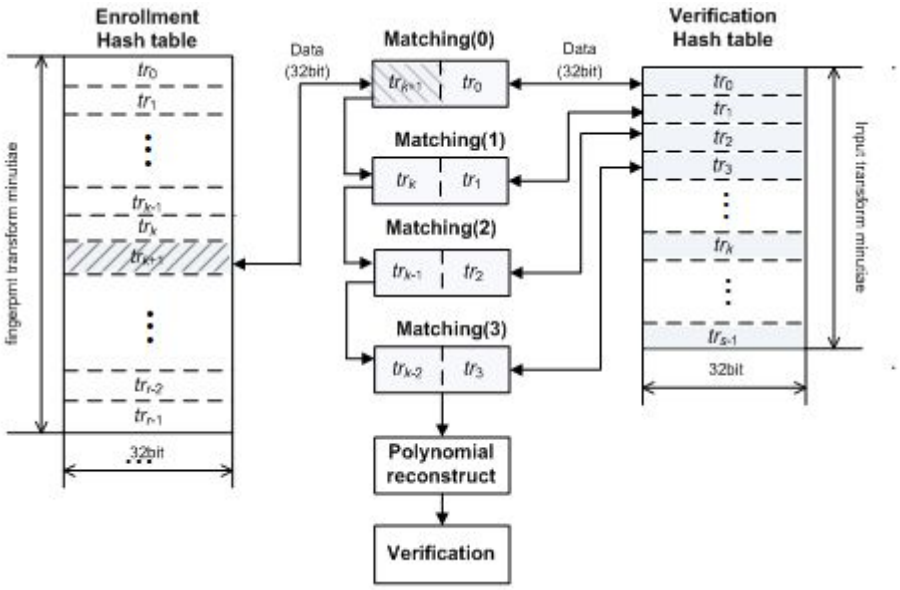


그림 33. 4개의 매칭모듈을 가진 하드웨어 모듈의 구조도

인증 해시 테이블의 tr_0, tr_1, tr_2 그리고 tr_3가 각각 매칭모듈(1), 매칭모듈(2), 매칭모듈(3), 그리고 매칭모듈(4)에 입력된다. 등록 해시 테이블의 tr_0가 매칭모듈(1)에 입력되어 인증 해시 테이블의 tr_0와 비교된다. 일치된 변환 특징점의 쌍의 수를 count모듈로 전송한다.

이 후, 등록 해시 테이블의 tr_0은 매칭모듈(1)에서 매칭모듈(2)로, 등록 해시 테이블의 tr_1은 등록 해시 테이블에서 매칭모듈(1)으로 전송된다. 매칭모듈(1)은 등록 해시 테이블의 tr_1과 인증 해시 테이블의 tr_0을, 매칭모듈(2)은 등록 해시 테이블의 tr_0와 인증 해시 테이블의 tr_1을 동시에 비교한다. 변환 특징점들의 일치여부를 판단한 후에 등록 지문 변환 특징점들은 다음 매칭모듈로 전송된다.

등록 해시 테이블의 tr_0은 매칭모듈(2)에서 매칭모듈(3)으로, 등록 해시 테이블의 tr_1은 매칭모듈(1)에서 매칭모듈(2)로, 등록 해시 테이블의 tr_2는 등록 해시 테이블에서 매칭모듈(1)로 전송된다. 매칭모듈(1)에서 등록 해시 테이블의 tr_2와 인

증 해시 테이블의 tr_0 을, 매칭모듈(2)에서 등록 해시 테이블의 tr_1 과 인증 해시 테이블의 tr_1 을, 매칭모듈(3)에서 등록 해시 테이블의 tr_0 와 인증 해시 테이블의 tr_2 을 동시에 비교한다. 변환 특징점들의 일치여부를 판단한 후 등록 해시 테이블의 tr 을 그 다음 매칭모듈로 전송된다.

등록 해시 테이블의 tr_0 은 매칭모듈(3)에서 매칭모듈(4)로, tr_1 은 매칭모듈(2)에서 매칭모듈(3)으로, tr_2 는 매칭모듈(1)에서 매칭모듈(2)로, tr_3 은 등록 해시 테이블에서 매칭모듈(1)로 전송된다. 매칭모듈(1)에서 등록 해시 테이블의 tr_3 과 인증 해시 테이블의 tr_0 을, 매칭모듈(2)에서 등록 해시 테이블의 tr_2 와 인증 해시 테이블의 tr_1 을, 매칭모듈(3)에서 등록 해시 테이블의 tr_1 과 인증 해시 테이블의 tr_2 를, 매칭모듈(4)에서 등록 해시 테이블의 tr_0 과 인증 해시 테이블의 tr_3 을 동시에 비교한다.

위의 동일한 방법으로 등록 해시 테이블의 tr_{r-1} 이 매칭모듈(4)에서 인증 해시 테이블의 tr_3 과 비교가 이루어질 때까지 수행한다. 모든 등록 해시 테이블의 tr 의 비교가 완료되면, 인증 해시 테이블의 tr_4 , tr_5 , tr_6 , 그리고 tr_7 이 각각 매칭모듈(1), 매칭모듈(2), 매칭모듈(3), 그리고 매칭모듈(4)에 입력된다. 다시 등록 해시 테이블의 tr_0 가 매칭모듈(1)에 전송되면서 위의 과정을 수행한다. 인증 해시 테이블의 tr_{s-1} 가 매칭모듈(4)에 입력되어 등록 해시 테이블의 tr_{r-1} 과 비교될 때까지 반복한다. 다항식 복원모듈과 인증모듈은 앞서 설명한 방법과 동일하다.

예를 들어, 등록 지문 템플릿의 수가 215(15개의 지문 특징점과 200개의 거짓 특징점)개, 인증 지문 템플릿이 10개라고 가정한다. 등록 해시 테이블의 tr 의 수는 215개, 인증 해시 테이블은 12개의 tr 로 구성된다.

인증 해시 테이블의 tr_0 , tr_1 , tr_2 과 tr_3 가 각각 매칭모듈(1), 매칭모듈(2), 매칭모듈(3), 그리고 매칭모듈(4)에 입력된다. 등록 해시 테이블의 tr_0 이 매칭모듈(1)에 입력되어 인증 해시 테이블의 tr_0 과 비교된다. 두 tr 의 일치여부를 판단한 후에 등록 해시 테이블의 tr_0 은 매칭모듈(1)에서 매칭모듈(2)로, 등록 해시 테이블의 tr_1 은 등록 해시 테이블에서 매칭모듈(1)로 전송된다. 매칭모듈(1)에서는 등록 해시 테이블의 tr_1 과 인증 해시 테이블의 tr_0 을, 매칭모듈(2)에서는 등록 해시 테

이블의 tr_0 와 인증 해시 테이블의 tr_1 이 동시에 비교한다. 일치여부를 판단하고 등록 해시 테이블의 tr 이 다음 매칭모듈로 전송된다.

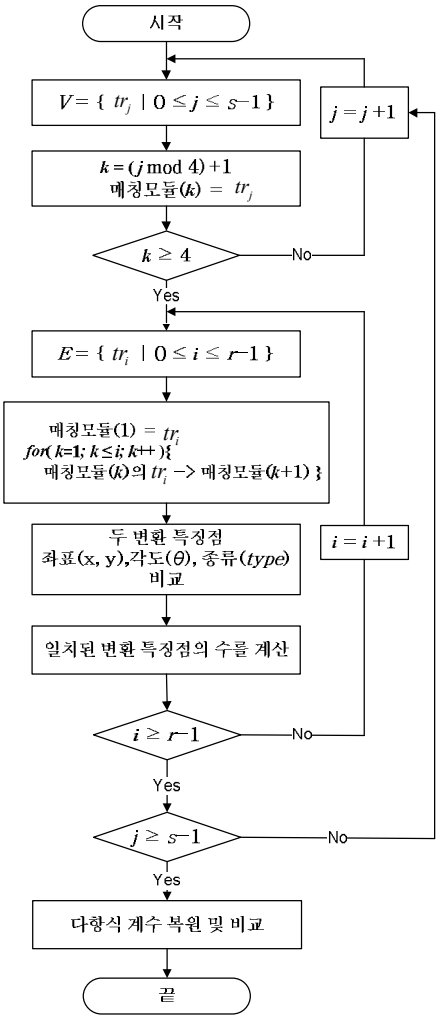


그림 34. 4개의 매칭모듈을 가진 하드웨어의 흐름도

등록 해시 테이블의 tr_0 은 매칭모듈(2)에서 매칭모듈(3)으로, tr_1 은 매칭모듈(1)에서 매칭모듈(2)로, tr_2 는 등록 해시 테이블에서 매칭모듈(1)로 전송된다. 동일한 방법으로 등록 해시 테이블의 tr_{214} 가 매칭모듈(4)에서 인증 해시 테이블의 tr_3 과 비교가 이루어질 때까지 수행되며 이 후에 인증 해시 테이블의 tr_4, tr_5, tr_6, 그리

고 tr_7이 매칭모듈(1), 매칭모듈(2), 매칭모듈(3), 그리고 매칭모듈(4)에 입력된다. 인증 해시 테이블의 tr_{11}이 매칭모듈(4)에 입력되어 등록 해시 테이블의 tr_{214}와 비교될 때까지 반복한다.

다음 8개, 16개, 32개, 64개의 매칭모듈로 구성된 하드웨어 모듈의 구조는 매칭모듈의 수가 다를 뿐 동일한 과정으로 수행된다. 64개의 매칭모듈을 가진 하드웨어 구조는 그림 35와 같다. 매칭모듈(1)부터 매칭모듈(64)로 구성되어 동시에 64쌍의 tr 들을 비교할 수 있다.

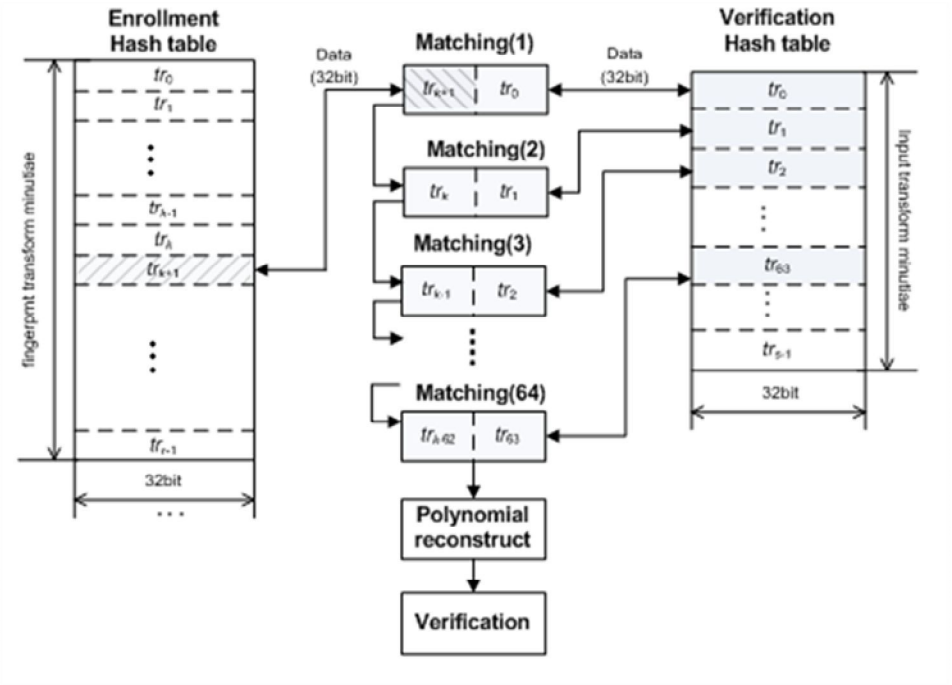


그림 35. 64개의 매칭모듈을 가진 하드웨어의 구조도

제안하는 병렬처리 하드웨어 모듈의 동작 과정을 요약하면 그림 36과 같이 표현할 수 있다.

Procedure : 제안하는 병렬처리 하드웨어의 정합 방법

Step 1. 인증 지문 변환 특징점 전송.
– 각각의 매칭모듈에 인증 해시 테이블의 tr 전송

Step 2. 등록 지문 변환 특징점 전송.
– 등록 해쉬 테이블에서 매칭모듈(1)로 tr 전송
– 이전의 매칭모듈에서 다음 매칭모듈로 전송

Step 3. 변환 특징점간의 비교.
– 두 변환 특징점의 좌표(x, y), 각도(θ), 종류($type$) 비교
– 일치되는 변환 특징점 수 계산

Step 4. 반복전송 I
– 등록 해시 테이블의 tr_{r-1}이 마지막 매칭모듈에 입력될 때까지
Step2와 Step3을 반복 실행

Step 5. 반복전송 II
– 인증 해시 테이블의 tr_{s-1}가 매칭모듈에 입력될 때까지
Step1, Step2, 그리고 Step3을 반복 실행

Step 6. 다항식 복원
– 일치된 변환 특징점과 다항식의 차수를 이용하여 수행
– 다항식의 계수 복원

Step 7. 인증
– 등록 과정에서 생성된 다항식의 계수와 복원된 다항식의 계수 비교

그림 36. 제안하는 병렬처리 하드웨어 동작 과정

제 3 절 성능 분석

본 논문에서 제안한 기하학적 해싱 기반 지문 퍼지볼트 시스템의 하드웨어 구조와 병렬처리 기법의 하드웨어 구조에 대한 실험을 하였다. 제안한 하드웨어 구조의 소프트웨어 모듈은 Visual C++ 6.0에서 C언어를 사용 구현하였다. PC의 주요환경은 다음과 같다.

① Intel Core 2 Duo E3700 2.67GHz

② Windows Vista 32bit Operation System

③ SDRAM 2.00GB

하드웨어 모듈의 구현은 그림 37과 같은 Spartan 3E starter board 개발 보드를 사용하였다. 개발 보드에는 32-bit RISC Processor와 DDR Interface을 구현할 수 있고, 50만 게이트의 용량을 가지고 있는 Xilinx XC3S500E FG320 디바이스가 내장되어 있다. Xilinx ISE 9.2 프로그램에서 VHDL언어를 사용하여 설계하였으며 하드웨어 시뮬레이션은 Modelsim XE 6.0에서 실험하였다.

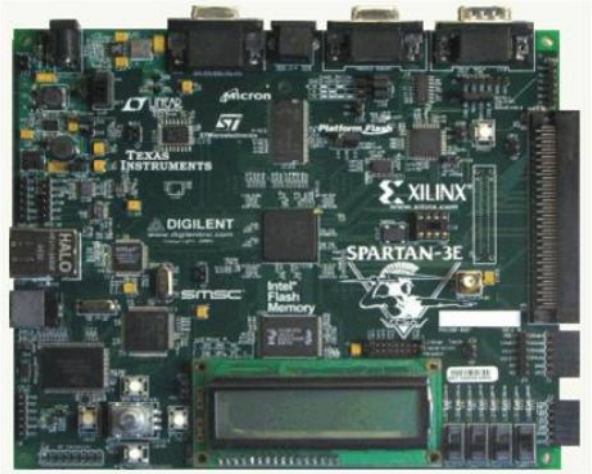


그림 37. Spartan 3E starter board

실험에 사용된 지문 특징점의 수는 36개에 거짓 특징점 200개를 추가하여 등록 템플릿을 생성하였다. 인증 지문 특징점은 34개였다. 그림 38은 제안한 하드웨어 구조의 매칭모듈 시뮬레이션 결과이다. 인증 지문 변환 특징점과 등록 지문 변환 특징점간의 일치 여부를 판단하고, 그 수를 계산한다.

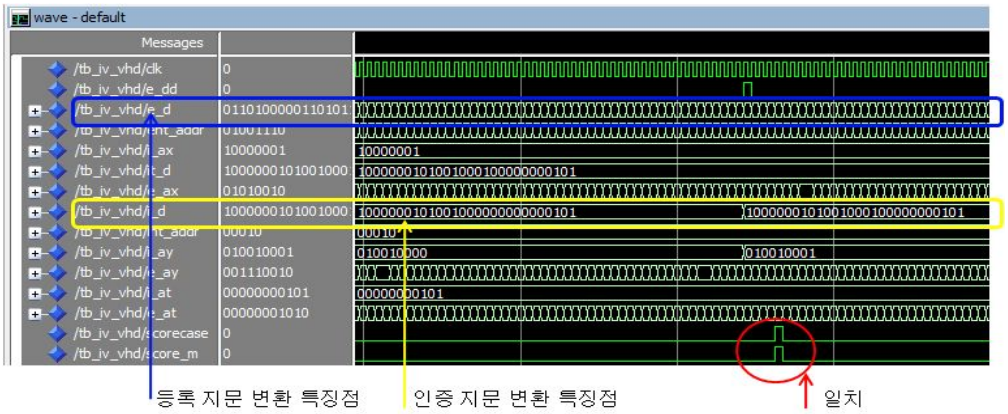


그림 38. 제안한 하드웨어 매칭모듈의 시뮬레이션 결과

실험결과, 그림 39는 제안한 하드웨어 구조의 회로 합성결과이다. 표 1은 개발보드 xilinx Spartan 3E starter board에서 제안한 하드웨어 구조의 주요 자원량을 나타내고 있다.

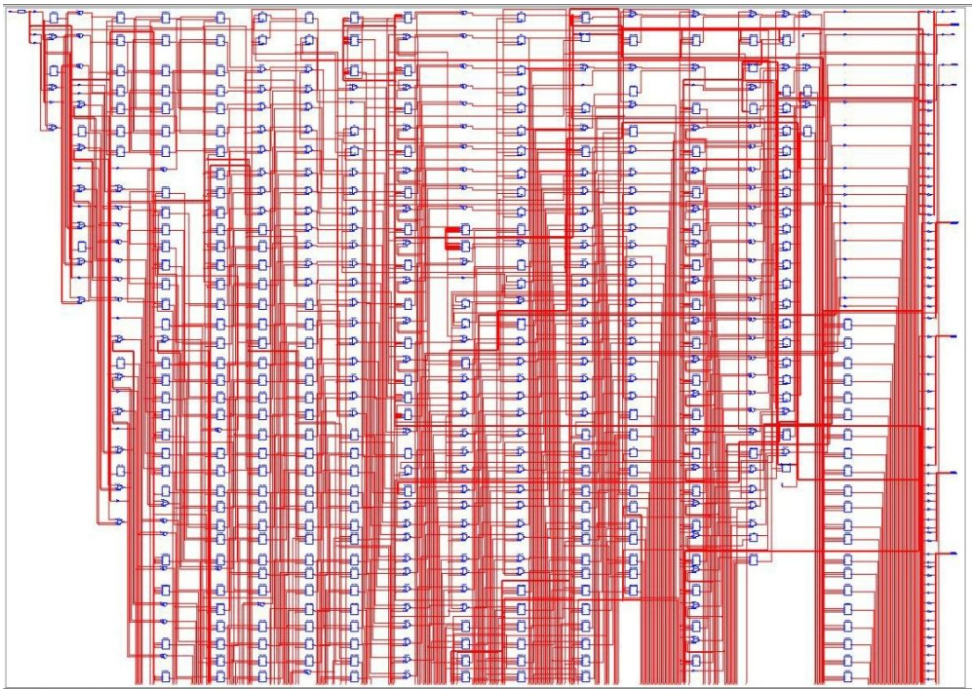
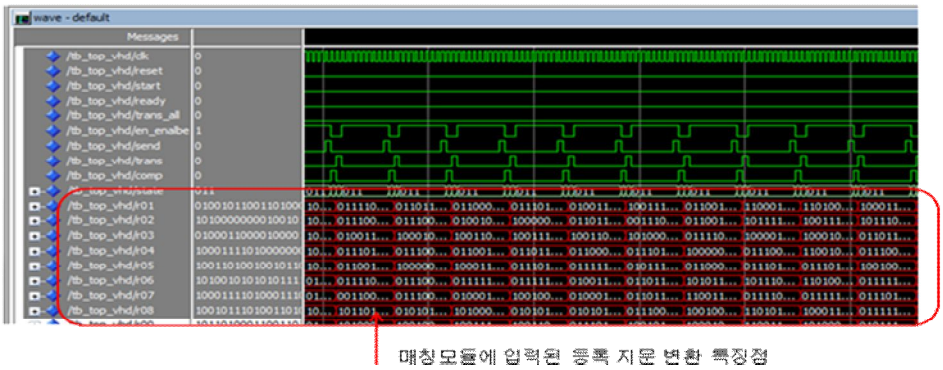


그림 39. 제안한 하드웨어 구조의 회로 합성결과

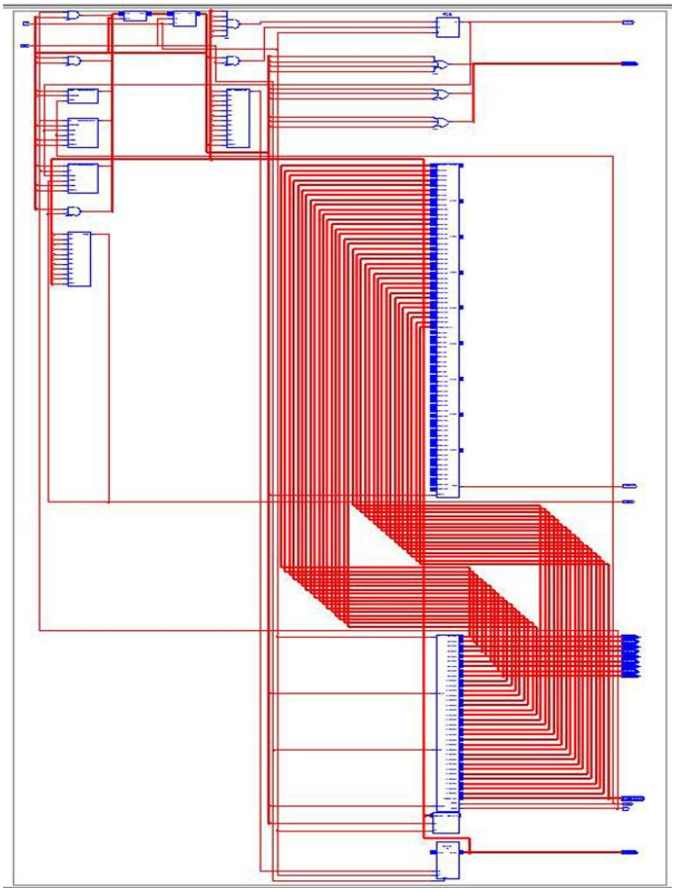
표 1. 제안한 하드웨어 모듈의 주요자원

	제안한 하드웨어 구조
Number of Slices	419 out of 4,656(9%)
Number of Slices Flip Flops	161 out of 9,312(2%)
Total Number of 4 input LUTs	668 out of 9,312(7%)

동일한 조건(지문 특징점 수, 추가된 거짓 특징점 수, 인증 지문 특징점 수)으로 제안하는 병렬처리 기법을 적용한 하드웨어 구조에 대해 실험하였다. 매칭모듈이 각각 2, 4, 8, 16, 32, 64개를 가지고 있을 때 병렬처리 기법의 하드웨어 구조의 성능에 대해 실험하였다. 2개의 매칭모듈을 가진 병렬처리 하드웨어 구조의 실험결과는 그림 40과 같으며 4개의 매칭모듈을 가진 하드웨어의 구조의 실험결과는 그림 41과 같다. 8개의 매칭모듈을 가진 하드웨어 구조의 실험결과는 그림 42, 16개의 매칭모듈을 가진 하드웨어 구조의 실험결과는 그림 43과 같다. 32개의 매칭모듈을 가진 하드웨어 구조의 실험결과는 그림 44와 같으며, 64개의 매칭모듈을 가진 하드웨어 구조의 실험결과는 그림 45와 같다.

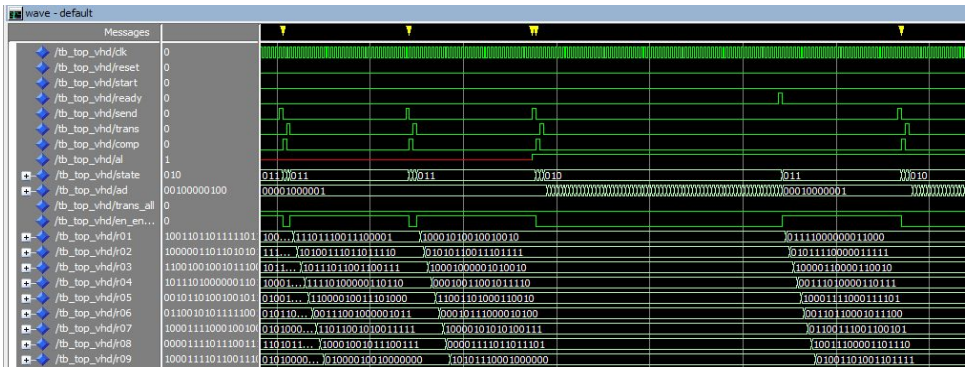


(a) 시뮬레이션 결과

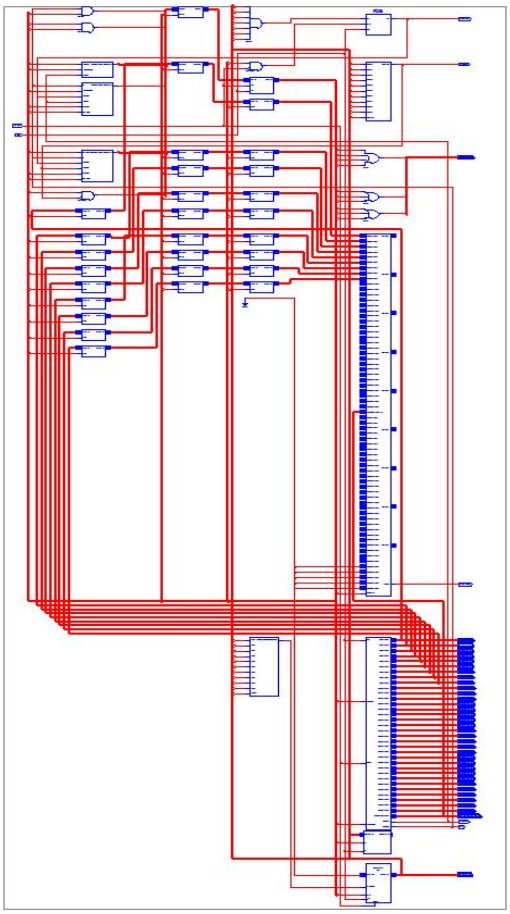


(b) 회로 합성결과

그림 40. 2개의 매칭모듈을 가진 하드웨어 구조의 실험결과

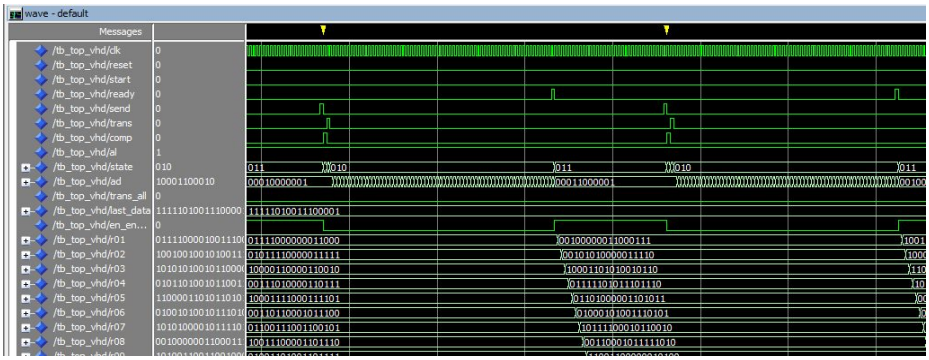


(a) 시뮬레이션 결과

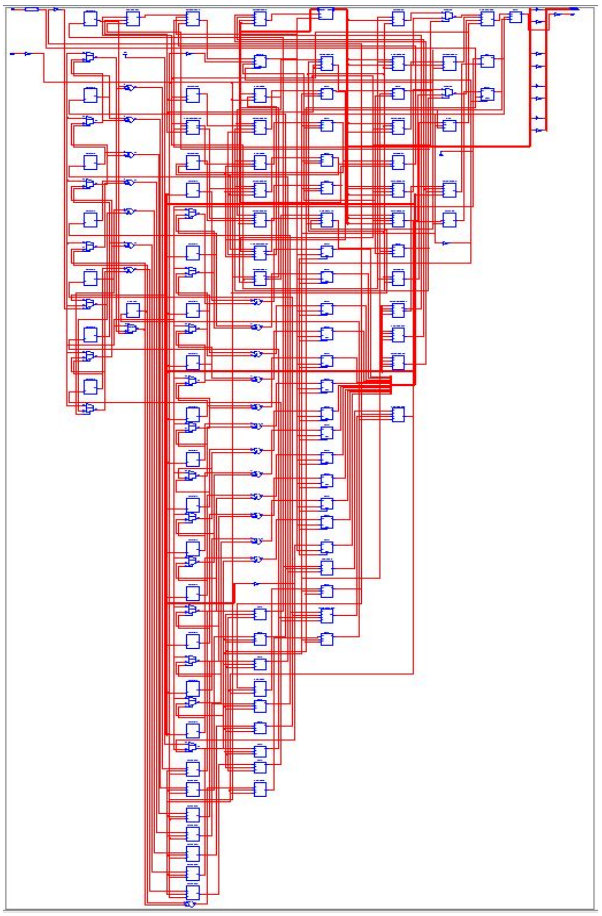


(b) 회로 합성결과

그림 41. 4개의 매칭모듈을 가진 하드웨어 구조의 실험결과

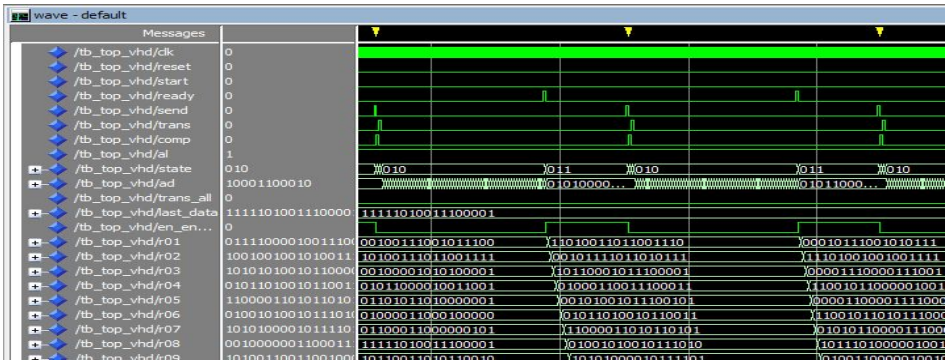


(a) 시뮬레이션 결과

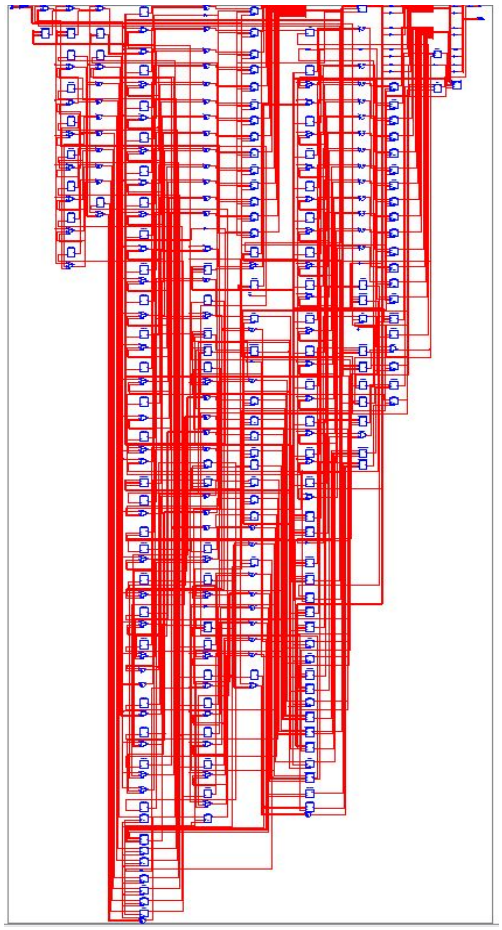


(b) 회로 합성결과

그림 42. 8개의 매칭모듈을 가진 하드웨어 구조의 실험결과

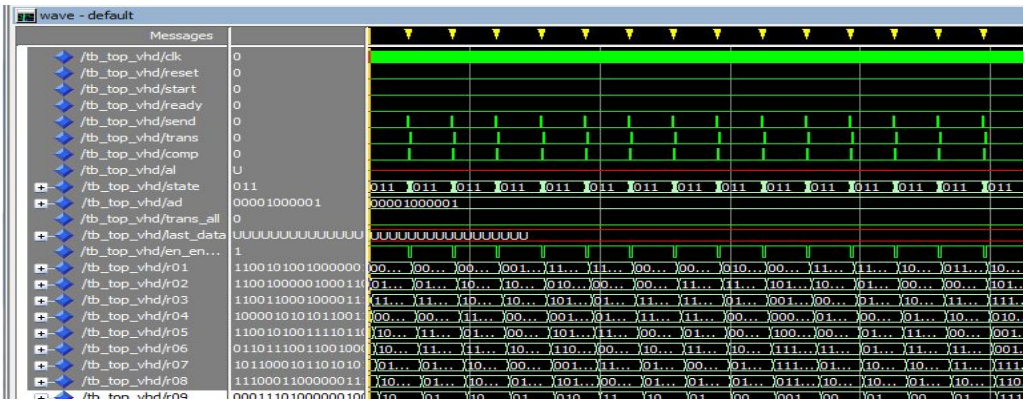


(a) 시뮬레이션 결과

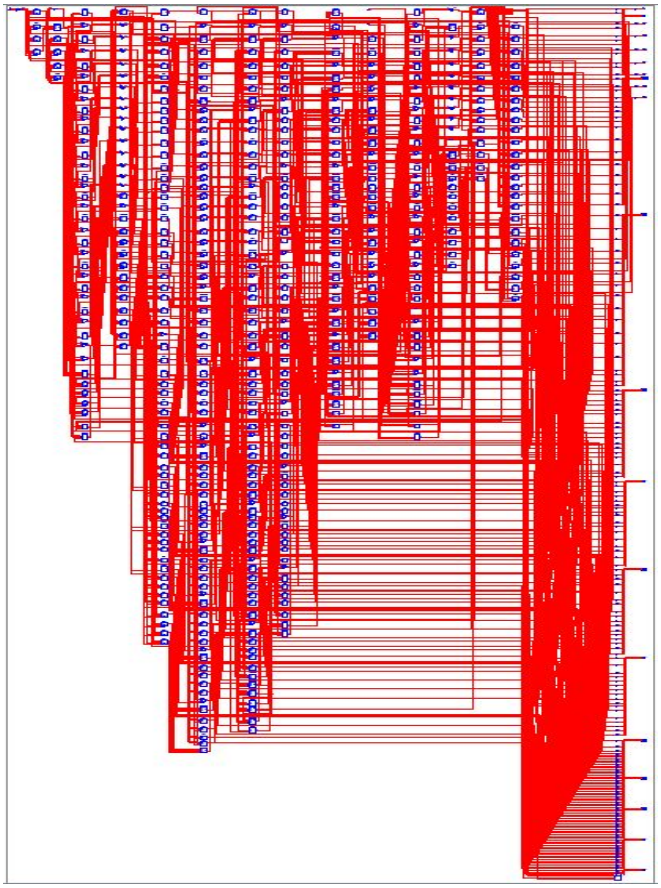


(b) 회로 합성결과

그림 43. 16개의 매칭모듈을 가진 하드웨어 구조의 실험결과

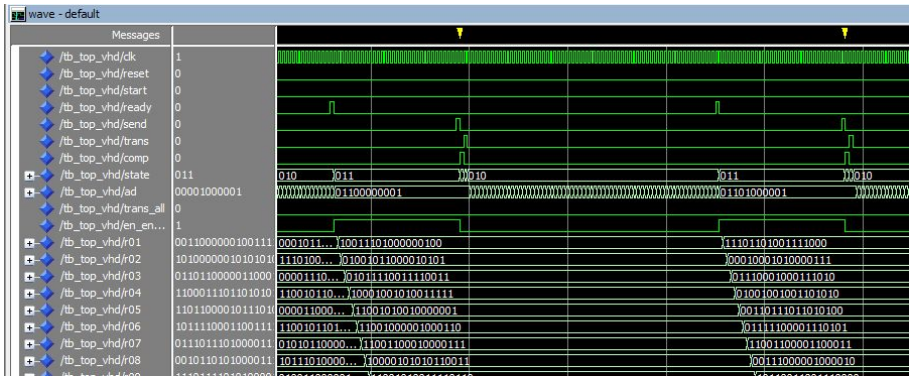


(a) 시뮬레이션 결과

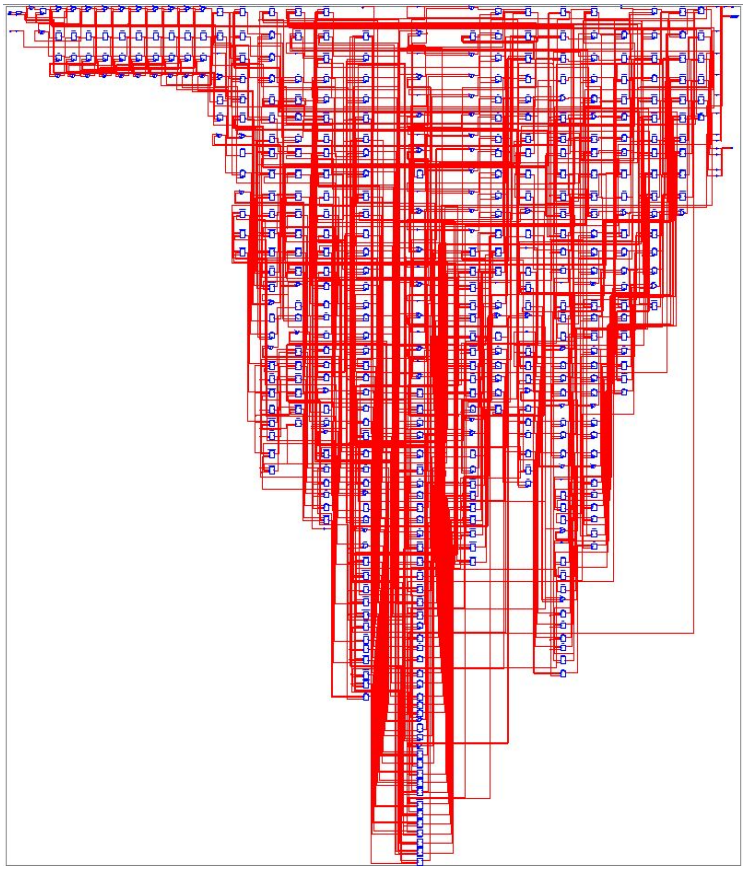


(b) 회로 합성결과

그림 44. 32개의 매칭모듈을 가진 하드웨어 구조의 실험결과



(a) 시뮬레이션 결과



(b) 회로 합성결과

그림 45. 64개의 매칭모듈을 가진 하드웨어 구조의 실험결과

표 2는 위의 시뮬레이션 결과 개발보드에서 사용된 자원량을 나타내고 있다. 2개의 매칭모듈을 가진 하드웨어 구조는 Number of Slices 10%, Number of Slices Flip Flop 2%, 그리고 Total Number of 4 input LUTs 8%를 사용한다. 64개의 매칭모듈을 가진 하드웨어 구조는 각각 65%, 30%, 그리고 65%를 사용하여 2개의 매칭모듈을 가진 하드웨어 구조보다 55%, 28%, 그리고 57%의 자원량을 더 사용하였다.

표 2. 매칭모듈의 수에 따른 병렬처리 기법 하드웨어 구조의 자원량

매칭모듈의 수	Number of Slices	Number of Slices Flip Flop	Total Number of 4 input LUTs
2개	496 out of 4,656 (10%)	246 out of 9,312 (2%)	762 out of 9,312 (8%)
4개	822 out of 4,656(17%)	618out of 9,312 (6%)	951 out of 9,312 (10%)
8개	1,046 out of 4,656(26%)	946 out of 9,312 (10%)	1,931 out of 9,312(20%)
16개	1,963 out of 4,656(42%)	1,161 out of 9,312(12%)	3,619out of 9,312(38%)
32개	2,793 out of 4,656(59%)	2,015 out of 9,312(21%)	4,962 out of 9,312(53%)
64개	3,048 out of 4,656(65%)	2,856 out of 9,312(30%)	6,081 out of 9,312(65%)

지문정보를 보호하기 위해 지문 퍼지볼트는 지문 템플릿에 다수의 거짓 특징점을 추가한다. 이에 제안한 하드웨어 모듈의 수행시간을 거짓 특징점의 수 100개, 200개, 300개, 400개에 따라 측정하였다. 표 3은 정합과정에서 필요한 cycle 수를 각각 소프트웨어, 1개, 2개, 4개, 8개, 16개, 32개, 64개의 매칭모듈을

가진 하드웨어 모듈에서 측정하였다. 그림 46은 개발보드의 클럭 주파수 50MHz를 동일하게 적용하여 계산된 시간이다. 표 3과 같이 거짓 특징점이 200개일 때, 1개의 매칭모듈을 가진 하드웨어 모듈과 소프트웨어는 약 0.89초, 64개의 매칭모듈을 가진 하드웨어 모듈과 소프트웨어는 약 1.12초 차이를 나타내고 있다. 이는 매칭모듈의 수가 많아질수록 시간 단축율이 커지는 것을 알 수 있다. 또한, 제안한 하드웨어 구조는 보안성을 향상시키기 위해 거짓 특징점 수를 증가시켜도, 실시간 처리가 가능함을 알 수 있다.

표 3. 거짓 특징점 수에 따른 정합과정의 cycle 수

거짓 특징점 / 모듈	100	200	300	400
S/W	30,128,840	56,376,516	75,087,761	110,009,410
1개의 매칭모듈	10,023,288,	12,027,946	18,763,596	26,461,481
2개의 매칭모듈	8,173,713	9,808,456	15,301,191	21,578,603
4개의 매칭모듈	4,086,907	4,904,288	7,650,689	10,789,434
8개의 매칭모듈	2,043,453	2,452,144	3,825,345	5,394,717
16개의 매칭모듈	1,021,714	1,226,057	1,912,649	2,697,325
32개의 매칭모듈	510,857	613,028	956,324	1,348,662
64개의 매칭모듈	480,807	576,968	900,070	1,239,330

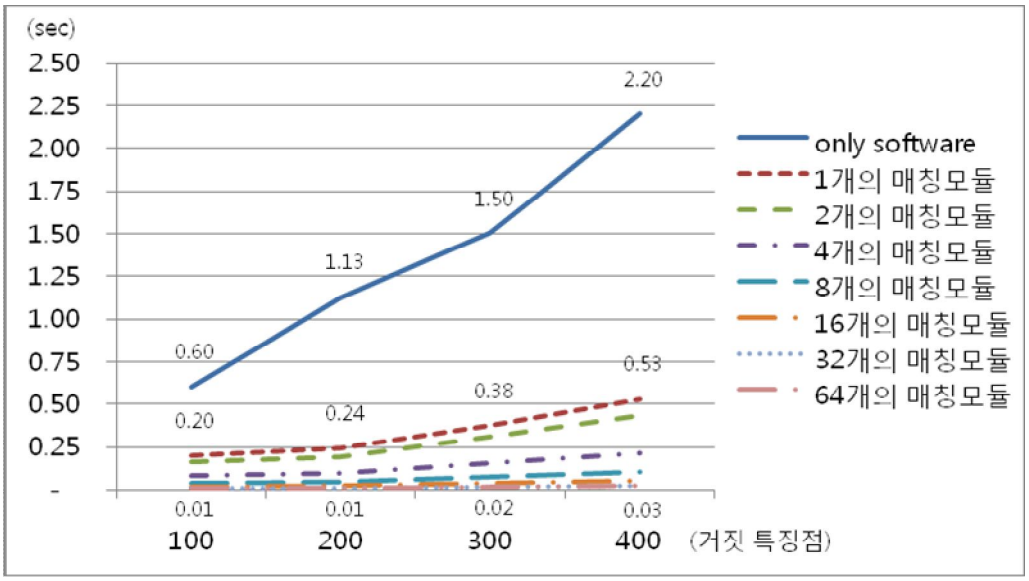


그림 46. 거짓 특징점 수에 따른 정합과정의 수행시간(sec)

제 5 장 결 론

지문정보를 이용한 사용자 인증 시스템은 편리함과 동시에 강력한 보안을 제공하나 지문정보가 도용 당하거나 유출되었을 때 심각한 문제를 발생시키게 된다. 최근 지문정보를 보호하기 위해 퍼지볼트 이론을 적용한 지문 퍼지볼트 연구가 진행되고 있다. 지문 퍼지볼트는 지문 템플릿에 거짓 특징점을 추가하여 지문 특징점을 보호한다. 하지만 지문 퍼지볼트는 지문의 기준점 부재에 따른 정렬 과정이 생략되어 지문 인식 시스템을 구현하는데 비현실적이다. 이 문제를 해결하기 위해 지문 퍼지볼트 시스템에 기하학적 해싱 기법을 적용하는 방법이 제안되었다. 기하학적 해싱을 통해 생성된 등록 해시 테이블과 인증 해시 테이블을 비교하여 정합과정을 수행함으로써 지문 정렬문제를 해결하였다. 하지만, 지문 특징점을 보호하기 위해 다수의 거짓 특징점이 추가될수록 지문 템플릿의 수가 증가하게 되고 그에 따라 생성되는 해시 테이블의 수도 증가하여 정합 수행시간이 길어지게 된다.

본 논문에서는 지문 퍼지볼트 시스템의 수행시간을 단축시키기 위한 하드웨어 구조를 제안하였다. 제안한 하드웨어 구조의 정합모듈은 입력 지문 해시 테이블의 하나의 변환 특징점마다 등록 지문 해시 테이블의 모든 변환 특징점을 정합하게 구현하였다. 또한, 시스템의 고속 처리를 위하여 병렬처리 기법을 적용한 하드웨어 구조의 정합 방법은 2^n개의 매칭모듈을 구성하여 동시에 정합시켰다.

실험결과, 36개의 지문 특징점, 200개의 거짓 특징점, 그리고 34개의 인증 지문 특징점에 대해 정합과정의 수행시간은 소프트웨어로만 구현했을 때 1.13초, 제안한 하드웨어 구조는 0.24초, 64개의 매칭모듈을 가진 병렬처리 하드웨어 구조는 0.01초가 소요되었다.

지문정보를 이용한 인식시스템의 상용화를 위해서는 대형이 아닌 소형으로, 저전력으로, 저가로 구현되어야 한다. 그러므로 제안한 지문 퍼지볼트 시스템의 하드웨어 구조는 지문정보를 보호하면서 지문 인식을 수행할 수 있으며, SOC(System on Chip)화하여 상용화가 가능할 것이다.

참고문헌

[1] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*, Springer, 2003.

[2] W. Stallings, *Cryptography and Network Security*, Pearson Ed. Inc., 2003.

[3] 바이오인식 위키백과사전, http://en.wikipedia.org/wiki/Biometrics

[4] S. Prabhakar, et al., "Biometric recognition: security and privacy concerns," *IEEE Security and Privacy*, pp. 33-42, 2003.

[5] The History of Fingerprints, http://onin.com/fp/fphistory.html

[6] 김학일, 안도성, 류춘우, 지문인식, 한국정보통신교육원, 2002.

[7] 채승훈, "하이브리드 지문인증 시스템에 관한 연구," 조선대학교 석사학위 논문, 2009.

[8] R. Bolle, J. Connell, and N. Ratha, "Biometric perils and patches," *Pattern Recognition*, vol. 32, pp. 2727-2738, 2002.

[9] B. Schneier, "The uses and abuses of biometrics," *Communications of the ACM*, vol. 42, no. 8, pp. 136, 1999.

[10] S. Prabhakar, S Pankanti, and A. Jain, "Biometric recognition: security and privacy concerns," *IEEE Security and Privacy*, pp. 33-42, 2003.

[11] A. Juels and M. Sudan, "A fuzzy vault scheme," *Proc. of Symp. on Information Theory*, pp. 408, 2002.

[12] A. Juels and M, Wattenberg, "A fuzzy commitment scheme," *Proc. of ACM Conf. on Computer and Comm. Security*, pp. 28-36, 1999.

[13] T. Clancy, N. Kiyavash, and D. Lin, "Secure smartcard-based fingerprint authentication," *Proc. of ACM SIGMM Multim., Biom. Met. & App.*, pp. 45-52, 2003.

[14] U. Uludag, S. Pankanti, and A. Jain, "Fuzzy vault for fingerprints," *LNCS*

3546, pp. 310-319, 2005.

[15] H. Wolfson, and I. Rigoutsos, “Geometric hashing: an overview,” *IEEE Computational Science and Engineering*, vol. 4, pp. 10–21, 1997.

[16] Y. Lamdan, H. Wolfson, “Geometric hashing: A general and efficient model-based recognition scheme,” *Proc. of 2nd Int. Conf. Computer Vision*, pp. 238-249, 1998.

[17] Y. Chung, D. Moon, S. Lee, S. Jung, T. Kim, and D. Ahn, “Automatic alignment of fingerprint features for fuzzy fingerprint vault.,” *LNCS*, vol. 3822, pp. 358–369, 2005.

[18] 이성주, 문대성, 김학재, 정용화, 이옥연, “3차원 기하학적 해싱을 이용한 퍼지볼트에서의 지문 정합,” 한국정보보호학회 논문지, 제 18권, 1호, pp. 11-21, 2008,

[19] S. Lee, D. Moon, H. Choi, Y. Chung, "Memory-efficient fuzzy fingerprint vault based on the geometric hashing," *Proc. 2008 Int. Conf. on Information Security and Assurance,* pp. 312-315, 2008.

[20] S. B. Pan, Y. H. Gil, D. Moon, Y. Chung, and C. H. Park, "A memory-efficient fingerprint verification algorithm using a multi-resolution accumulator array,” *ETRI Journal*, vol. 22, no. 3, pp. 179-186, 2003.

[21] 김기철, 김민, 정용화, 반성범, “실시간 지문 식별을 위한 하드웨어 구현,” 한국정보보호학회 논문지, 제 14권, 6호, pp. 79-89, 2004.

[22] 문대성, 최우용, 문기영, “다중 다항식을 이용한 지문 퍼지볼트,” 한국정보보호학회 논문지, 제 19권, 1호, pp. 125- 133, 2009.

[23] 임성진, 채승훈, 반성범, “지문 퍼지볼트 시스템의 VLSI 구조,” 2009 정보통신분야학회 합동학술대회, pp. 97-100, 2009.

[24] S. Lim, S. Chae, D. Moon, Y. Chung, N. Lee, and S. Pan, “VLSI architecture for the fuzzy fingerprint vault with automatic alignment module,” *Proc. Int. Conf. FGCN*, pp. 470-476, 2009.

감사의 글

이 글을 마지막으로 석사과정 동안 저에게 많은 도움을 주셨던 분들께 감사의 말씀을 드리고자 합니다.

우선, 저를 이끌어 주시고 본 논문의 연구계획에서부터 완성에 이르기까지 지도하여 주신 반성범 교수님께 깊은 감사를 드립니다. 그리고 논문심사과정을 통해 본 논문을 지도하여 주시고 심사하여 주신 고낙용 교수님과 곽근창 교수님께 감사를 드립니다.

또한, 대학원 생활 전반에 대해 도움을 준 지성이형, 승훈이형 그리고 논문작성에 바쁜 저에게 많은 도움을 준 해민이, 즐거운 연구실 생활을 하게 해 준 아라, 민구, 그리고 경훈이 형에게 고마움을 전합니다. 아울러 옆에서 항상 파이팅을 외쳐 준 사랑스런 올챙이에게도 고맙다는 말을 전합니다.

마지막으로 항상 곁에서 아낌없는 성원과 격려해 주신 아버지, 어머니, 그리고 동생 지은이에게 이 논문을 드립니다.

2009년 11월

임성진 올림